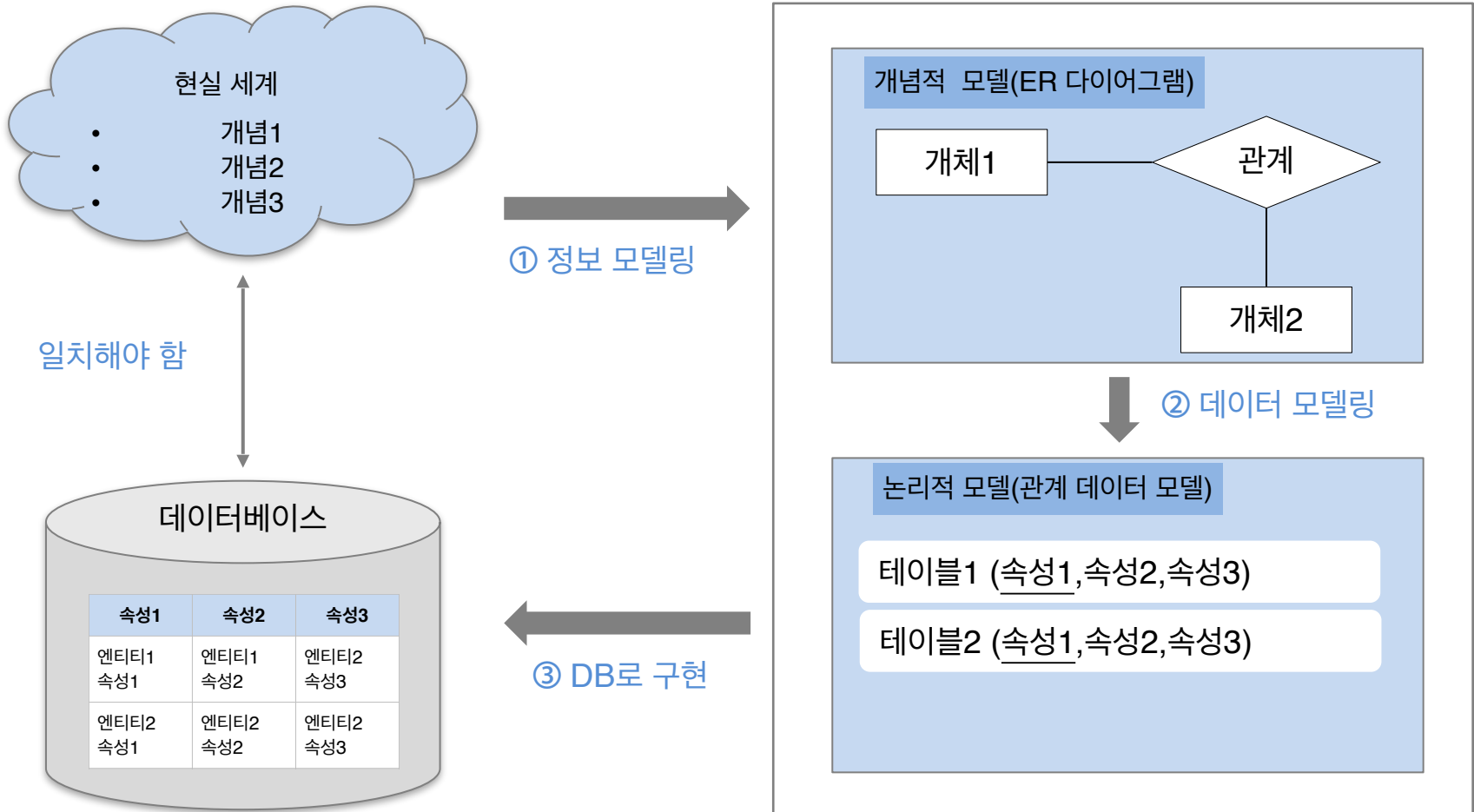


오라클 11g XE 데이터베이스 모델링



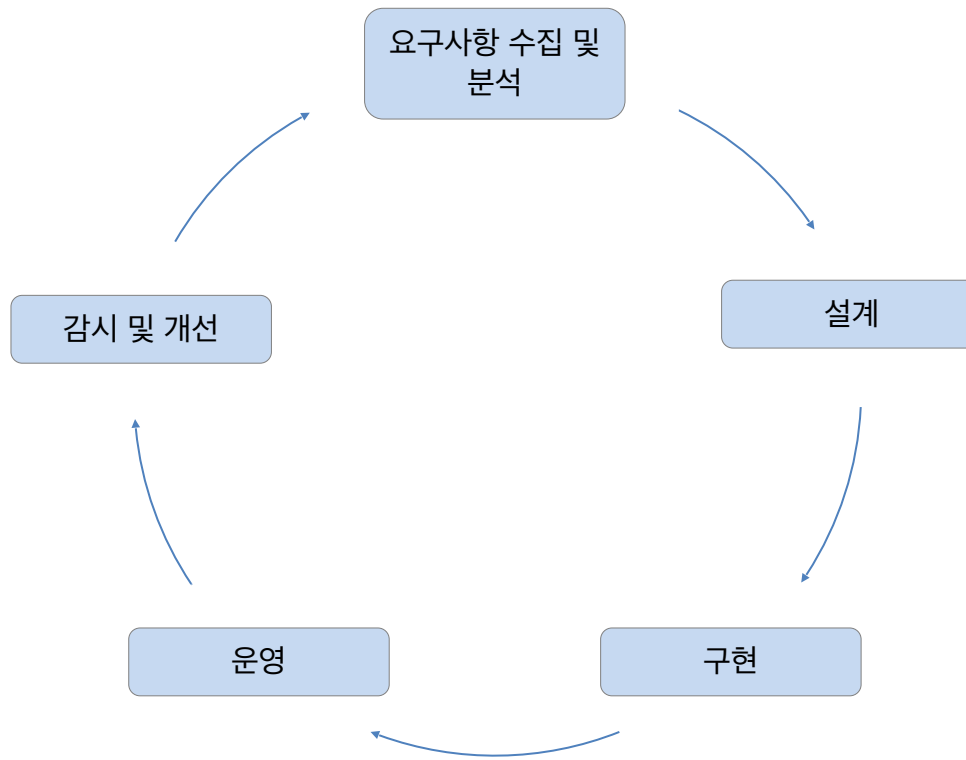
훈련교사 : 전 은 석



데이터 모델링의 개념

- 데이터베이스 생명주기(database life cycle)란?

데이터베이스의 생성과 운영에 관련된 특징



데이터베이스 생명주기

① 요구사항 수집 및 분석

사용자들의 요구사항을 듣고 분석하여 데이터베이스 구축의 범위를 정하는 단계

② 설계

분석된 요구사항을 기초로 주요 개념과 업무 프로세스 등을 식별하고(개념적 설계), 사용하는 DBMS의 종류에 맞게 변환(논리적 설계)한 후, 데이터베이스 스키마를 도출(물리적 설계)함.

③ 구현

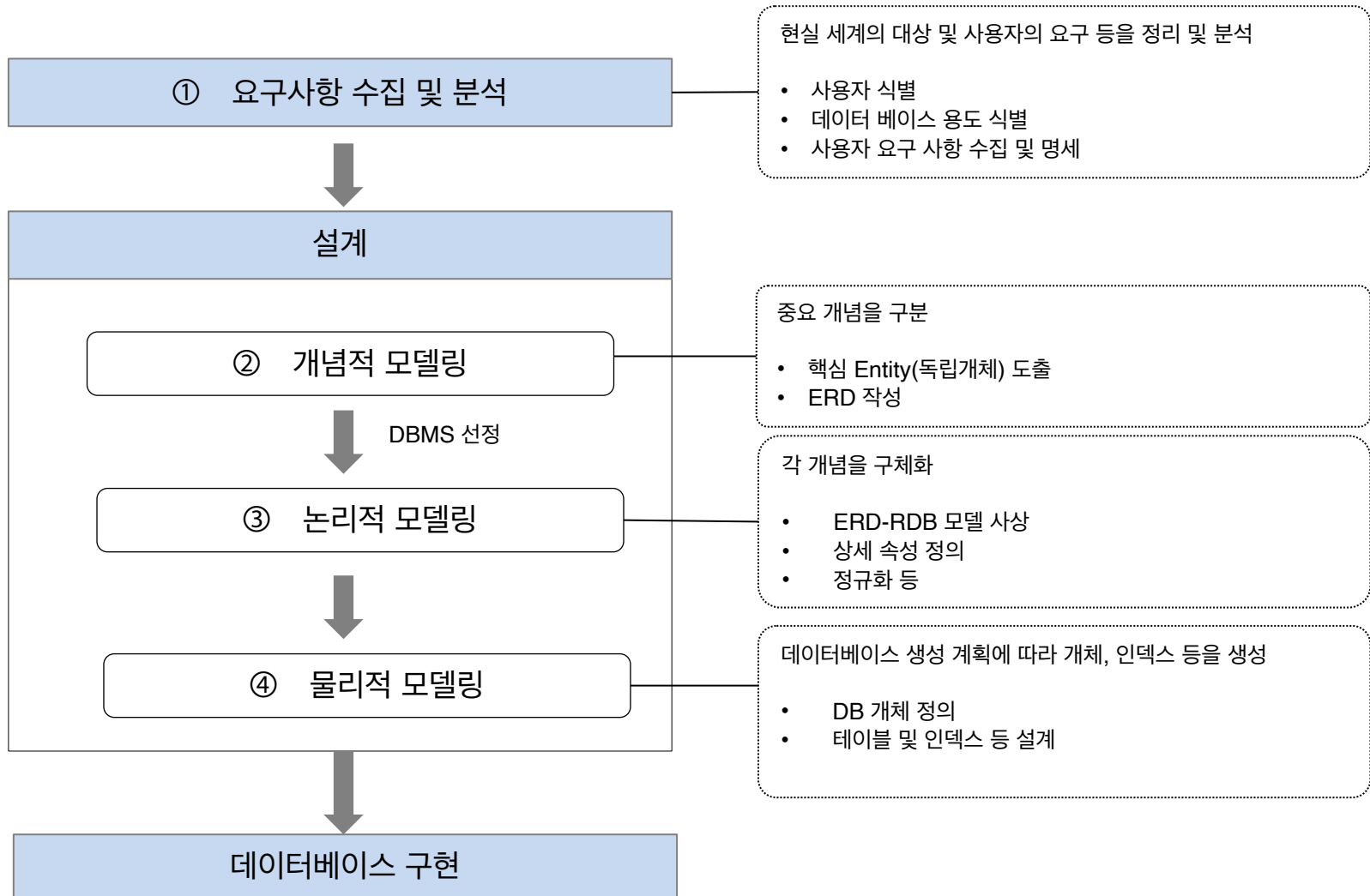
설계 단계에서 생성한 스키마를 실제 DBMS에 적용하여 테이블 및 관련 객체(뷰, 인덱스 등)를 만듦.

④ 운영

구현된 데이터베이스를 기반으로 소프트웨어를 구축하여 서비스를 제공함.

⑤ 감시 및 개선

데이터베이스 운영에 따른 시스템의 문제를 관찰하고 데이터베이스 자체의 문제점을 파악하여 개선함.

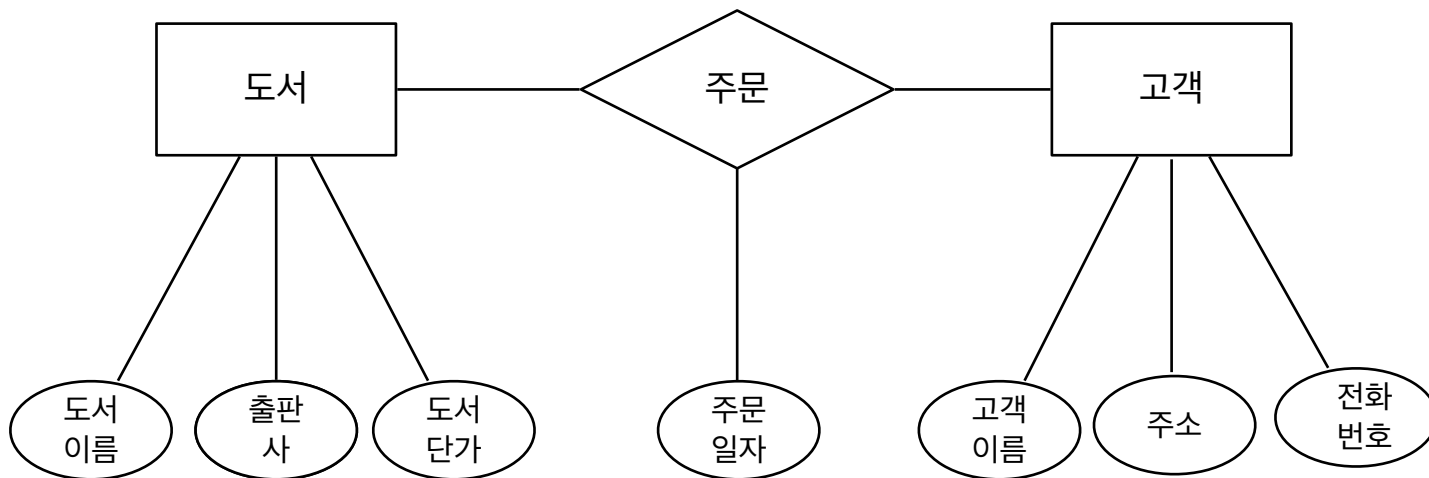


데이터 모델링 과정

■ 요구사항 수집 방법

1. 실제 문서를 수집하고 분석함.
2. 담당자와의 인터뷰나 설문조사를 통해 요구사항을 직접 수렴함.
3. 비슷한 업무를 처리하는 기존의 데이터베이스를 분석함.
4. 각 업무와 연관된 모든 부분을 살펴봄.

- 개념적 모델링(conceptual modeling) : 요구사항을 수집하고 분석한 결과를 토대로 업무의 핵심적인 개념을 구분하고 전체적인 뼈대를 만드는 과정
- 개체(entity)를 추출하고 각 개체들 간의 관계를 정의하여 ER 다이어그램(ERD, Entity Relationship Diagram)을 만드는 과정까지를 말함.



개념적 모델링의 예

■ 논리적 모델링(logical modeling)이란?

- 개념적 모델링에서 만든 ER 다이어그램을 사용하려는 DBMS에 맞게 사상(매핑, mapping)하여 실제 데이터베이스로 구현하기 위한 모델을 만드는 과정

논리적 모델링의 예



도서 (도서번호, 도서이름, 출판사이름, 도서단가)

고객 (고객번호, 고객이름, 주소, 전화번호)

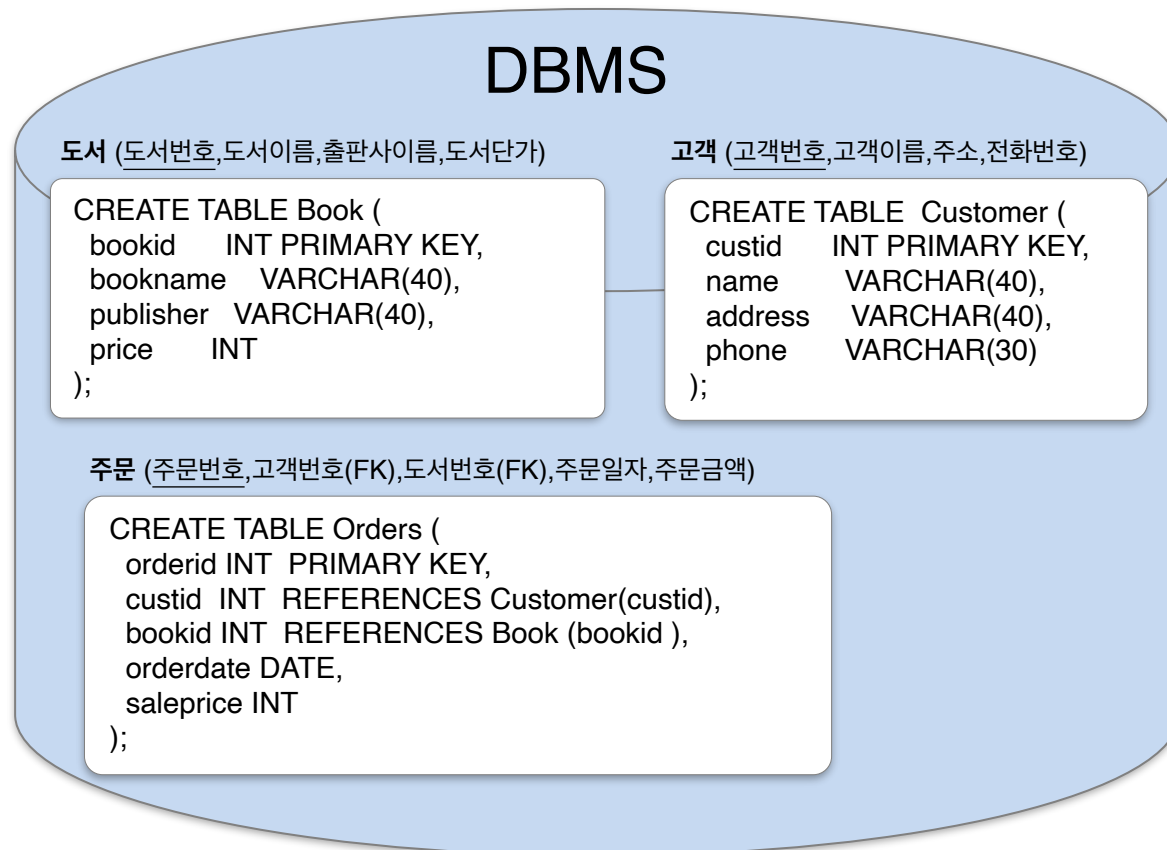
주문 (주문번호, 고객번호(FK), 도서번호(FK), 주문일자, 주문금액)

■ 논리적 모델링 과정

1. 개념적 모델링에서 추출하지 않았던 상세 속성들을 모두 추출함.
2. 정규화 수행
3. 데이터 표준화 수행

■ 물리적 모델링(physical modeling)이란?

- 작성된 논리적 모델을 실제 컴퓨터의 저장 장치에 저장하기 위한 물리적 구조를 정의하고 구현하는 과정
- DBMS의 특성에 맞게 저장 구조를 정의해야 데이터베이스가 최적의 성능을 낼 수 있음.



- 물리적 모델링 시 트랜잭션, 저장 공간 설계 측면에서 고려할 사항
 1. 응답시간을 최소화해야 한다.
 2. 얼마나 많은 트랜잭션을 동시에 발생시킬 수 있는지 검토해야 한다.
 3. 데이터가 저장될 공간을 효율적으로 배치해야 한다.

❖ 정규화의 목적

데이터베이스의 이상현상 제거

■ 잘못 설계된 데이터베이스가 어떤 이상현상(anomaly)을 일으키는지 알아보기

학생번호	학생이름	학과	주소	강좌이름	강의실
501	박지성	컴퓨터과	영국 맨체스타	데이터베이스	공학관 110
401	김연아	체육학과	대한민국 서울	데이터베이스	공학관 110
402	장미란	체육학과	대한민국 강원도	스포츠경영학	체육관 103
502	추신수	컴퓨터과	미국 클리블랜드	자료구조	공학관 111
501	박지성	컴퓨터과	영국 맨체스타	자료구조	공학관 111

학생수강 테이블

- **삭제이상(deletion anomaly)** : 튜플 삭제 시 같이 저장된 다른 정보까지 연쇄적으로 삭제되는 현상
→ 연쇄삭제(triggered deletion) 문제 발생
- **삽입이상(insertion anomaly)** : 튜플 삽입 시 특정 속성에 해당하는 값이 없어 NULL 값을 입력해야 하는 현상
→ NULL 값 문제 발생
- **수정이상(update anomaly)** : 튜플 수정 시 중복된 데이터의 일부만 수정되어 데이터의 불일치 문제가 일어나는 현상 → 불일치 (inconsistency) 문제 발생

학생번호	학생이름	학과	주소	강좌이름	강의실
501	박지성	컴퓨터과	영국 맨체스타	데이터베이스	공학관 110
401	김연아	체육학과	대한민국 서울	데이터베이스	공학관 110
402	장미란	체육학과	대한민국 강원도	스포츠경영학	체육관 103
502	추신수	컴퓨터과	미국 클리블랜드	자료구조	공학관 111
501	박지성	컴퓨터과	영국 맨체스타	자료구조	공학관 111

403	박세리	체육학과	대한민국 대전	NULL	NULL
-----	-----	------	---------	------	------

DELETE

- 장미란 학생 삭제
- 연쇄삭제 문제

UPDATE

- 박지성 학생 주소 변경
- 불일치 문제

INSERT

- 박세리 학생 삽입
- NULL 값 문제

데이터 조작과 이상현상

■ Summer 테이블을 생성하고 데이터를 삽입하는 SQL 문

```
DROP TABLE Summer; /* 기존 테이블이 있으면 삭제하고 새로 생성하기 위한 준비 */
```

```
CREATE TABLE Summer
```

```
(  sid      NUMBER,  
   class    VARCHAR2(20),  
   price    NUMBER  
);
```

```
INSERT INTO Summer VALUES (100, 'FORTRAN', 20000);
```

```
INSERT INTO Summer VALUES (150, 'PASCAL', 15000);
```

```
INSERT INTO Summer VALUES (200, 'C', 10000);
```

```
INSERT INTO Summer VALUES (250, 'FORTRAN', 20000);
```

```
/* 생성된 Summer 테이블 확인 */
```

```
SELECT      *  
FROM        Summer;
```

SID	CLASS	PRICE
100	FORTRAN	20000
150	PASCAL	15000
200	C	10000
250	FORTRAN	20000

■ 각 질의에 대한 SQL문을 직접 실습해보기

Summer 테이블을 이용하여 처리하는 질의와 SQL 문

질의	SQL 문
계절학기를 듣는 학생의 학번과 수강하는 과목은?	SELECT sid, class FROM Summer;
C 강좌의 수강료는?	SELECT price FROM Summer WHERE class='C';
수강료가 가장 비싼 과목은?	SELECT DISTINCT class FROM Summer WHERE price = (SELECT max(price) FROM Summer);
계절학기를 듣는 학생 수와 수강료 총액은?	SELECT COUNT(*), SUM(price) FROM Summer;

■ 삭제이상

질의] 200번 학생의 계절학기 수강신청을 취소하시오.

```
/* C 강좌 수강료 조회 */  
SELECT price "C 수강료"  
FROM Summer  
WHERE class='C';
```

C 수강료
10000

```
/* 200번 학생의 수강신청 취소 */  
DELETE FROM Summer  
WHERE sid=200;
```

```
/* C 강좌 수강료 다시 조회 */ => C 수강료 조회 불가능!!  
SELECT price "C 수강료"  
FROM Summer  
WHERE class='C';
```

C 수강료

```
/* 다음 실습을 위해 200번 학생 자료 다시 입력 */  
INSERT INTO Summer VALUES (200, 'C', 10000);
```


■ 삽입이상

질의] 계절학기에 새로운 자바 강좌를 개설하시오.

/* 자바 강좌 삽입 */ => NULL을 삽입해야 한다. NULL 값은 문제가 있을 수 있다.

INSERT INTO Summer VALUES (NULL, 'JAVA', 25000);

/* Summer 테이블 조회 */

SELECT *
FROM Summer;

SID	CLASS	PRICE
100	FORTTRAN	20000
150	PASCAL	15000
250	FORTTRAN	20000
200	C	10000
(null)	JAVA	25000

/* NULL 값이 있는 경우 주의할 질의 : 튜플은 다섯 개지만 수강학생은 총 네 명임 */

SELECT COUNT(*) "수강인원"
FROM Summer;

수강인원
5

SELECT COUNT(sid) "수강인원"
FROM Summer;

수강인원
4

SELECT count(*) "수강인원"
FROM Summer
WHERE sid IS NOT NULL;

수강인원
4

■ 수정이상

질의] FORTRAN 강좌의 수강료를 20,000원에서 15,000원으로 수정하시오.

/* FORTRAN 강좌 수강료 수정 */

```
UPDATE Summer
SET price=15000
WHERE class='FORTRAN';
```

```
SELECT *
FROM Summer;
```

SID	CLASS	PRICE
100	FORTRAN	15000
150	PASCAL	15000
250	FORTRAN	15000
200	C	10000
(null)	JAVA	25000

```
SELECT DISTINCT price "FORTRAN 수강료"
FROM Summer
WHERE class='FORTRAN';
```

FORTRAN 수강료
15000

/* 다음 실습을 위해 FORTRAN 강좌의 수강료를 다시 20,000원으로 복구 */

```
UPDATE Summer
SET price=20000
WHERE class='FORTRAN';
```

/* 만약 UPDATE 문을 다음과 같이 작성하면 데이터 불일치 문제가 발생함 */

```
UPDATE Summer
SET price=15000
WHERE class='FORTRAN' AND sid=100;
```

/* Summer 테이블을 조회하면 FORTRAN 강좌의 수강료가 한 건만 수정되었음 */

```
SELECT *  
FROM Summer;
```

SID	CLASS	PRICE
100	FORTRAN	15000
150	PASCAL	15000
250	FORTRAN	20000
200	C	10000
(null)	JAVA	25000

/* FORTRAN 수강료를 조회하면 두 건이 나옴(데이터 불일치 문제 발생) */

```
SELECT price "FORTRAN 수강료"  
FROM Summer  
WHERE class='FORTRAN';
```

FORTRAN 수강료
15000
20000

/* 다음 실습을 위해 FORTRAN 강좌의 수강료를 다시 20,000원으로 복구 */

```
UPDATE Summer  
SET price=20000  
WHERE class='FORTRAN';
```

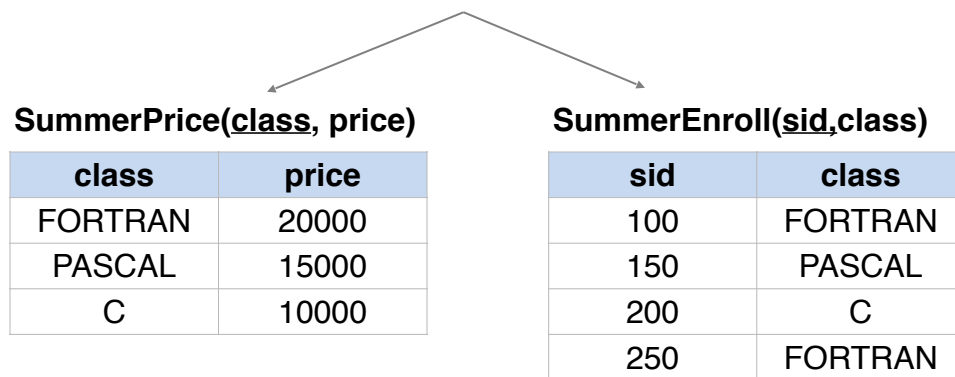
/* 다음 실습을 위해 sid가 NULL인 튜플 삭제 */

```
DELETE FROM Summer  
WHERE sid IS NULL;
```

- 테이블의 구조를 수정하여 이상현상이 발생하지 않는 사례

Summer(sid, class, price)

sid	class	price
100	FORTTRAN	20000
150	PASCAL	15000
200	C	10000
250	FORTTRAN	20000



Summer 테이블의 분리

■ SummerPrice 테이블과 SummerEnroll 테이블을 생성하는 SQL 문

```
/* 기존 테이블이 있으면 삭제하고 새로 생성하기 위한 준비 */
```

```
DROP TABLE SummerPrice;  
DROP TABLE SummerEnroll;
```

```
/* SummerPrice 테이블 생성 */
```

```
CREATE TABLE SummerPrice  
( class VARCHAR2(20),  
  price NUMBER  
);
```

```
INSERT INTO SummerPrice VALUES ('FORTRAN', 20000);  
INSERT INTO SummerPrice VALUES ('PASCAL', 15000);  
INSERT INTO SummerPrice VALUES ('C', 10000);
```

```
SELECT * FROM SummerPrice;
```

CLASS	PRICE
FORTRAN	20000
PASCAL	15000
C	10000

```
/* SummerEnroll 테이블 생성 */
```

```
CREATE TABLE SummerEnroll  
( sid NUMBER,  
  class VARCHAR2(20)  
);
```

```
INSERT INTO SummerEnroll VALUES (100, 'FORTRAN');  
INSERT INTO SummerEnroll VALUES (150, 'PASCAL');  
INSERT INTO SummerEnroll VALUES (200, 'C');  
INSERT INTO SummerEnroll VALUES (250, 'FORTRAN');
```

```
SELECT * FROM SummerEnroll;
```

SID	CLASS
100	FORTRAN
150	PASCAL
200	C
250	FORTRAN

■ 각 질의에 대한 SQL문 실습하기

SummerPrice 테이블과 SummerEnroll 테이블을 이용하여 처리하는 질의와 SQL 문

질의	SQL 문
계절학기를 듣는 학생의 학번과 수강하는 과목은?	SELECT sid, class FROM SummerEnroll;;
C 강좌의 수강료는?	SELECT price FROM SummerPrice WHERE class='C';
수강료가 가장 비싼 과목은?	SELECT DISTINCT class FROM SummerPrice WHERE price = (SELECT max(price) FROM SummerPrice);
계절학기를 듣는 학생 수와 수강료 총액은?	SELECT COUNT(*), SUM(price) FROM SummerPrice, SummerEnroll WHERE SummerPrice.class=SummerEnroll.class;

■ 삭제이상 없음

질의] 200번 학생의 계절학기 수강신청을 취소하시오.

/* C 강좌 수강료 조회 */

```
SELECT price "C 수강료"  
FROM SummerPrice  
WHERE class='C';
```

C 수강료
10000

```
DELETE  
FROM SummerEnroll  
WHERE sid=200;
```

```
SELECT *  
FROM SummerEnroll;
```

SID	CLASS
100	FORTRAN
150	PASCAL
250	FORTRAN

/* C 강좌의 수강료가 존재하는지 확인 */ => 삭제이상 없음!!

```
SELECT price "C 수강료"  
FROM SummerPrice  
WHERE class='C';
```

C 수강료
10000

- 삽입이상 없음

질의] 계절학기에 새로운 자바 강좌를 개설하시오.

/* 자바 강좌 삽입, NULL 값을 입력할 필요 없음 */

INSERT INTO SummerPrice VALUES ('JAVA', 25000);

SELECT *
FROM SummerPrice;

CLASS	PRICE
FORTRAN	20000
PASCAL	15000
C	10000
JAVA	25000

/* 수강신청 정보 확인 */

SELECT *
FROM SummerEnroll;

SID	CLASS
100	FORTRAN
150	PASCAL
250	FORTRAN

- 수정이상 없음

질의] FORTRAN 강좌의 수강료를 20,000원에서 15,000원으로 수정하시오.

```
UPDATE SummerPrice
SET price=15000
WHERE class='FORTRAN';
```

```
SELECT price "FORTRAN 수강료"
FROM SummerPrice
WHERE class='FORTRAN';
```

FORTRAN 수강료
15000

❖ 함수 종속성의 개념

- 학생수강성적 릴레이션의 각 속성 사이에는 의존성이 존재함.
- 어떤 속성 A의 값을 알면 다른 속성 B의 값이 유일하게 정해지는 의존 관계를 ‘속성 B는 속성 A에 종속한다(dependent)’ 혹은 ‘속성 A는 속성 B를 결정한다(determine)’라고 함.
- ‘ $A \rightarrow B$ ’로 표기하며, A를 B의 결정자라고 함.

학생수강성적

학생번호	학생이름	주소	학과	학과사무실	강좌이름	강의실	성적
501	박지성	영국 맨체스타	컴퓨터과	공학관101	데이터베이스	공학관 110	3.5
401	김연아	대한민국 서울	체육학과	체육관101	데이터베이스	공학관 110	4.0
402	장미란	대한민국 강원도	체육학과	체육관101	스포츠경영학	체육관 103	3.5
502	추신수	미국 클리블랜드	컴퓨터과	공학관101	자료구조	공학관 111	4.0
501	박지성	영국 맨체스타	컴퓨터과	공학관101	자료구조	공학관 111	3.5

학생수강성적 릴레이션

- 학생수강성적 릴레이션에서 종속관계에 있는 예

학생번호 → 학생이름

학생번호 → 주소

강좌이름 → 강의실

학과 → 학과사무실

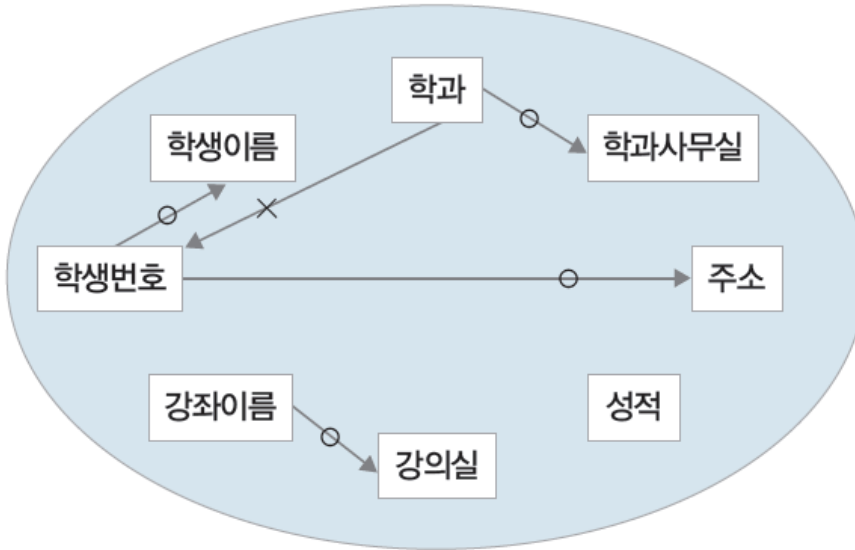
- 종속하지 않는 예

학생이름 → 강좌이름

학과 → 학생번호

- 종속하는 것처럼 보이지만 주의 깊게 보면 그렇지 않은 예

학생이름 → 학과



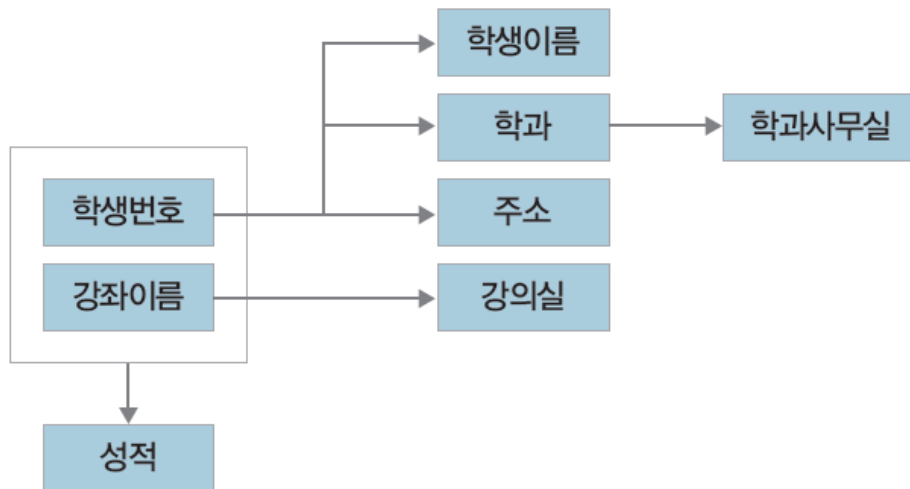
학생수강성적 릴레이션의 종속관계

함수 종속성(FD, Functional Dependency)

릴레이션 R과 R에 속하는 속성의 집합 X, Y가 있을 때, X 각각의 값이 Y의 값 한 개와 대응이 될 때 'X는 Y를 함수적으로 결정한다'라고 하고 $X \rightarrow Y$ 로 표기함. 이때 X를 결정자(determinant)라고 하고, Y를 종속 속성(dependent attribute)이라고 함. 함수 종속성은 보통 릴레이션 설계 때 속성의 의미로부터 정해짐.

■ 함수 종속성 다이어그램(functional dependency diagram)은 함수 종속성을 나타내는 표기법

- 릴레이션의 속성 : 직사각형
- 속성 간의 함수 종속성 : 화살표
- 복합 속성 : 직사각형으로 묶어서 그림



학생수강성적 릴레이션의 함수 종속성 다이어그램

함수 종속성 규칙(functional dependency rule)

X, Y, Z가 릴레이션 R에 포함된 속성의 집합이라고 할 때,

함수 종속성에 관한 다음과 같은 규칙이 성립

부분집합(Subset) 규칙 : if $Y \subseteq X$, then $X \rightarrow Y$

증가(Augmentation) 규칙 : If $X \rightarrow Y$, then $XZ \rightarrow YZ$

이행(Transitivity) 규칙 : If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$

위 세 가지 규칙으로부터 부가적으로 다음의 규칙을 얻을 수 있음

결합(Union) 규칙 : If $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$

분해(Decomposition) 규칙 : If $X \rightarrow YZ$, then $X \rightarrow Y$ and $X \rightarrow Z$

유사이행(Pseudotransitivity) 규칙 : If $X \rightarrow Y$ and $WY \rightarrow Z$, then $WX \rightarrow Z$

학생수강성적 릴레이션에 함수 종속성 규칙을 적용한 예

적용 규칙	사례	설명
부분집합 규칙 $\text{if } Y \subseteq X, \text{ then } X \rightarrow Y$	$(\text{학과, 주소}) \rightarrow \text{학과}$	학과는 (학과, 주소)의 부분집합 속성이므로, ‘(학과, 주소) \rightarrow 학과’ 성립
증가 규칙 $\text{If } X \rightarrow Y, \text{ then } XZ \rightarrow YZ$	$(\text{학생번호, 강좌이름}) \rightarrow$ (학생이름, 강좌이름)	‘학생번호 \rightarrow 학생이름’이므로 강좌이름을 추가하여, ‘(학생번호, 강좌이름) \rightarrow (학생이름, 강좌이름)’ 성립
이행 규칙 $\text{: If } X \rightarrow Y \text{ and } Y \rightarrow Z, \text{ then } X \rightarrow Z$	$\text{학생번호} \rightarrow \text{학과사무실}$	‘학생번호 \rightarrow 학과’, ‘학과 \rightarrow 학과사무실’이므로 이행 규칙을 적용하여, ‘학생번호 \rightarrow 학과사무실’ 성립
결합 규칙 $\text{If } X \rightarrow Y \text{ and } X \rightarrow Z, \text{ then } X \rightarrow YZ$	$\text{학생번호} \rightarrow (\text{학생이름, 주소})$	‘학생번호 \rightarrow 학생이름’, ‘학생번호 \rightarrow 주소’이므로 결합 규칙을 적용하여, ‘학생번호 \rightarrow (학생이름, 주소)’ 성립
분해 규칙 $\text{If } X \rightarrow YZ, \text{ then } X \rightarrow Y \text{ and } X \rightarrow Z$	$\text{학생번호} \rightarrow \text{학생이름},$ $\text{학생번호} \rightarrow \text{주소}$	‘학생번호 \rightarrow (학생이름, 주소)’이므로 분해하여, ‘학생번호 \rightarrow 학생이름’, ‘학생번호 \rightarrow 주소’ 성립
유사이행 규칙 $\text{If } X \rightarrow Y \text{ and } WY \rightarrow Z, \text{ then } WX \rightarrow Z$	$(\text{강좌이름, 학생이름}) \rightarrow \text{성적}$	‘학생이름 \rightarrow 학생번호’(학생이름이 같은 경우가 없다고 가정한다), ‘(강좌이름, 학생번호) \rightarrow 성적’이므로 유사이행 규칙을 적용하여, ‘(강좌이름, 학생이름) \rightarrow 성적’ 성립

- 릴레이션의 함수 종속성을 파악하기 위해서는 우선 기본키를 찾아야 함.
- 기본키가 함수 종속성에서 어떤 역할을 하는지 알면 이상현상을 제거하는 정규화 과정을 쉽게 이해할 수 있음

함수 종속성과 기본키

릴레이션 $R(K, A_1, A_2, A_3, \dots, A_n)$ 에서 K 가 기본키면, $K \rightarrow R$ 이 성립.

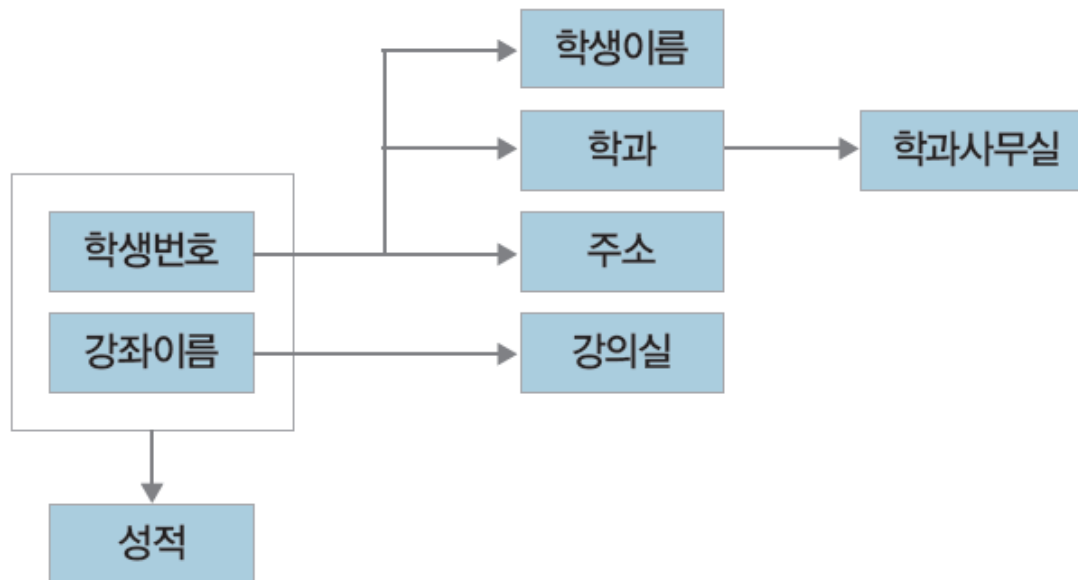
즉 기본키는 릴레이션의 모든 속성에 대해 결정자(determinant)임.

예) 이름이 같은 학생이 없다고 가정하면, ‘이름 \rightarrow 학과, 이름 \rightarrow 주소, 이름 \rightarrow 취득학점’이므로
‘이름 \rightarrow 이름, 학과, 주소, 취득학점’이 성립한다. 즉 이름 속성이 학생 릴레이션의 전체를 결정함.

이름	학과	주소	취득학점
박지성	컴퓨터과	영국 맨체스터	92
김연아	체육학과	대한민국 서울	95
장미란	체육학과	대한민국 강원도	98
추신수	컴퓨터과	미국 클리블랜드	99

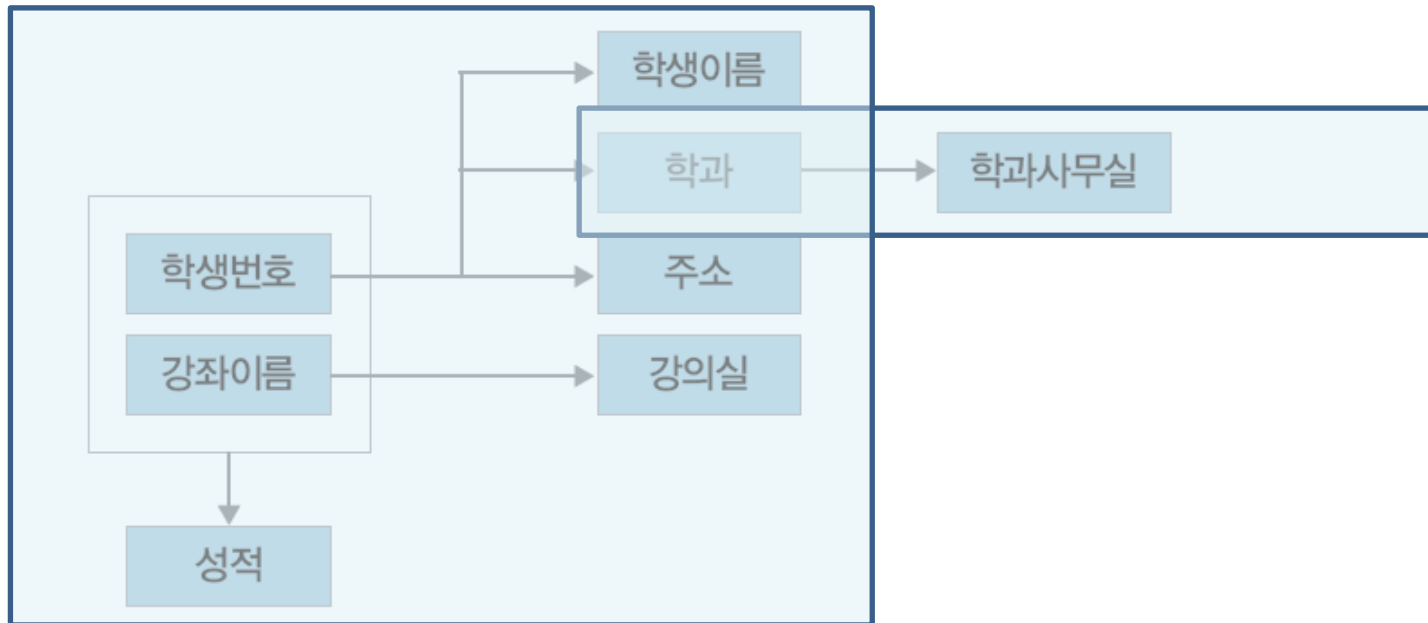
학생 릴레이션

- 이상현상은 한 개의 릴레이션에 두 개 이상의 정보가 포함되어 있을 때 나타남.
기본키가 아니면서 결정자인 속성이 있을 때 발생한다.
- 학생수강성적 릴레이션의 경우 학생 정보(학생번호, 학생이름, 주소, 학과)와 강좌 정보(강좌이름, 강의실)가 한 릴레이션에 포함되어서 이상현상이 나타남.
(학과, 학생번호, 강좌이름은 기본키가 아니면서 결정자인 예이다)



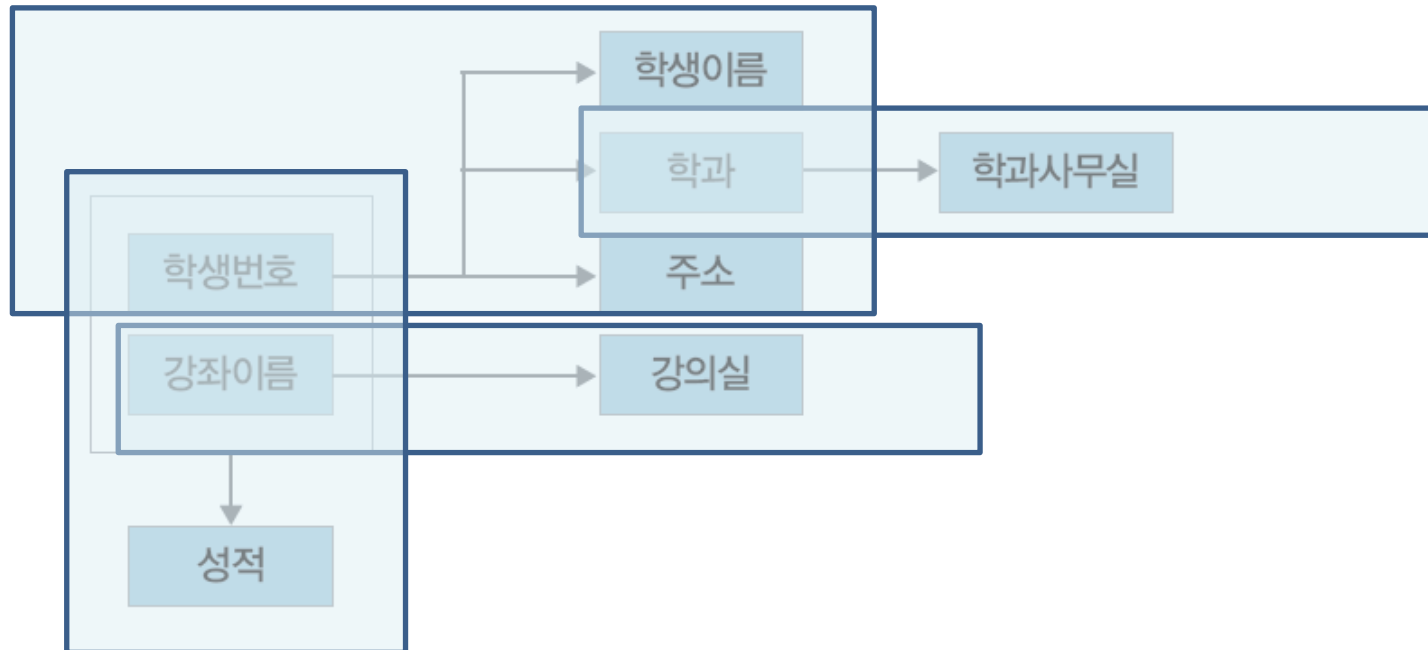
학생수강성적 릴레이션의 함수 종속성 다이어그램

- 이상현상을 없애려면 릴레이션을 분해한다.
- (학과, 학과사무실) 속성을 학생수강성적 릴레이션에서 분리하는 예



학생수강성적 릴레이션의 함수 종속성 다이어그램

- 릴레이션의 분해(계속....)



학생수강성적 릴레이션의 함수 종속성 다이어그램

- **학생수강성적 릴레이션에서 부분 릴레이션을 분해하기**

- 분해할 때 부분 릴레이션의 결정자는 원래 릴레이션에 남겨두어야 함. 그래야 분해된 부분 릴레이션이 원래 릴레이션과 관계를 형성할 수 있음.

- **1단계 : 학생수강성적 릴레이션에서 (강좌이름, 강의실)을 분리**

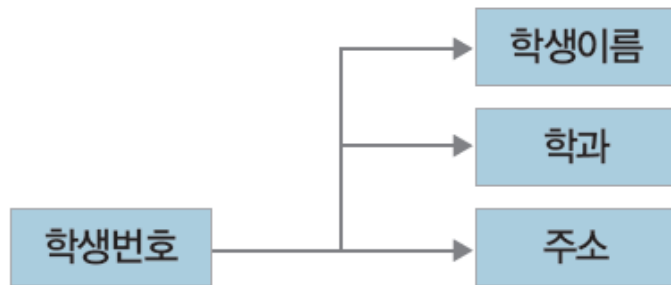
학생수강성적1(학생번호, 학생이름, 학과, 주소, 강좌이름, 성적, 학과사무실)
강의실(강좌이름, 강의실)

- **2단계 : 학생수강성적1 릴레이션에서 (학생번호, 강좌이름, 성적)을 분리**

학생학과(학생번호, 학생이름, 학과, 주소, 학과사무실)
학생성적(학생번호, 강좌이름, 성적)
강의실(강좌이름, 강의실)

- **3단계 : 학생학과 릴레이션에서 (학과, 학과사무실)을 분리**

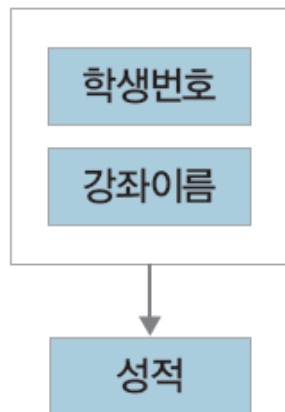
학생(학생번호, 학생이름, 학과, 주소)
학과(학과, 학과사무실)
학생성적(학생번호, 강좌이름, 성적)
강의실(강좌이름, 강의실)



(a) 학생



(b) 학과



(c) 학생성적



(d) 강의실

학생수강성적 릴레이션을 분해한 결과

- 정규화 과정
- 무손실 분해
- 정규화 정리

- 이상현상이 발생하는 릴레이션을 분해하여 이상현상을 없애는 과정
- 이상현상이 있는 릴레이션은 이상현상을 일으키는 함수 종속성의 유형에 따라 등급을 구분 가능.
- 릴레이션은 정규형 개념으로 구분하며, 정규형이 높을수록 이상현상은 줄어듦.



오토바이
1등급



자동차
2등급



기차
3등급



비행기
4등급

▲ 이동수단의 유형에 따른 안전도 등급의 구분 : 등급이 높을수록 빠르고 안전하다

<div>R1(...)</div> <div>R2(...)</div> <div>R3(...)</div>	<div>R4(...)</div> <div>R5(...)</div>	<div>R6(...)</div>	<div>R7(...)</div> <div>R8(...)</div>
제 1정규형	제 2정규형	제 3정규형	... 정규형

▲ 함수 종속성의 유형에 따른 등급의 구분 : 정규형이 높을수록 이상현상은 줄어든다

이동수단과 릴레이션의 등급 구분

- 릴레이션 R의 모든 속성 값이 원자값을 가지면 제 1정규형이라고 함.
- 제 1정규형으로 변환

고객취미들(이름, 취미들) 릴레이션을 고객취미(이름, 취미) 릴레이션으로 바꾸어 저장하면 제 1정규형을 만족한다.

고객취미들(이름, 취미들)

이름	취미들
김연아	인터넷
추신수	영화, 음악
박세리	음악, 쇼핑
장미란	음악
박지성	게임



고객취미(이름, 취미)

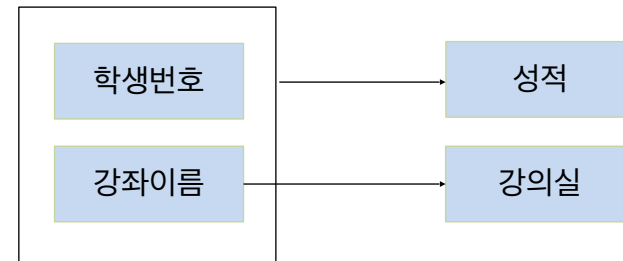
이름	취미
김연아	인터넷
추신수	영화
추신수	음악
박세리	음악
박세리	쇼핑
장미란	음악
박지성	게임

속성 값이 원자값을 갖도록 분해

- 릴레이션 R이 제 1정규형이고 기본키가 아닌 속성이 기본키에 완전 함수 종속일 때 제 2정규형이라고 함.
- 완전 함수 종속(full functional dependency) : A와 B가 릴레이션 R의 속성이고 $A \rightarrow B$ 종속성이 성립할 때, B가 A의 속성 전체에 함수 종속하고 부분 집합 속성에 함수 종속하지 않을 경우 완전 함수 종속라고 함.

수강강좌

학생번호	강좌이름	강의실	성적
501	데이터베이스	공학관 110	3.5
401	데이터베이스	공학관 110	4.0
402	스포츠경영학	체육관 103	3.5
502	자료구조	공학관 111	4.0
501	자료구조	공학관 111	3.5



수강강좌 릴레이션

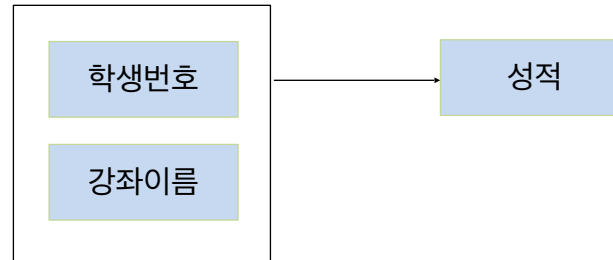
* 후보키는 무엇인가?

■ 제 2정규형으로 변환

수강강좌 릴레이션에서 이상현상을 일으키는(강좌이름, 강의실)을 분해함.

수강

학생번호	강좌이름	성적
501	데이터베이스	3.5
401	데이터베이스	4.0
402	스포츠경영학	3.5
502	자료구조	4.0
501	자료구조	3.5



강의실

강좌이름	강의실
데이터베이스	공학관 110
스포츠경영학	체육관 103
자료구조	공학관 111

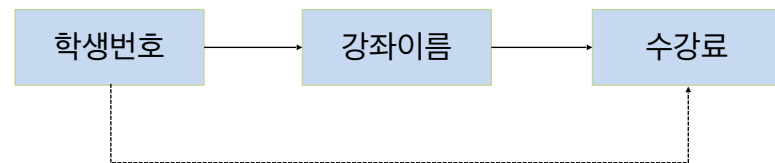


수강강좌 릴레이션을 수강, 강의실 릴레이션으로 분해

- 릴레이션 R이 제 2정규형이고 기본키가 아닌 속성이 기본키에 비이행적non-transitive으로 종속할 때(직접 종속) 제 3정규형이라고 함.
- 이행적 종속이란 $A \rightarrow B$, $B \rightarrow C$ 가 성립할 때 $A \rightarrow C$ 가 성립되는 함수 종속성

계절학기

학생번호	강좌이름	수강료
501	데이터베이스	20000
401	데이터베이스	20000
402	스포츠경영학	15000
502	자료구조	25000



계절학기 릴레이션

- * 계절학기 강좌는 학생은 한 강좌만 신청할 수 있다고 가정한다.
- * 후보키는 무엇인가?

■ 제 3정규형으로 변환

계절학기 릴레이션에서 이상현상을 일으키는 (강좌이름, 수강료)를 분해함

계절수강

학생번호	강좌이름
501	데이터베이스
401	데이터베이스
402	스포츠경영학
502	자료구조



수강료

강좌이름	수강료
데이터베이스	20000
스포츠경영학	15000
자료구조	25000

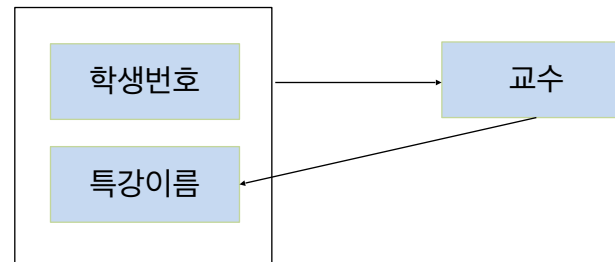


계절학기 릴레이션을 계절수강, 수강료 릴레이션으로 분해

- 릴레이션 R에서 함수 종속성 $X \rightarrow Y$ 가 성립할 때 모든 결정자 X가 후보키이면 BCNF 정규형이라고 함.

특강수강

학생번호	특강이름	교수
501	소셜네트워크	김교수
401	소셜네트워크	김교수
402	인간과 동물	승교수
502	창업전략	박교수
501	창업전략	홍교수



특강수강 릴레이션

- * 교수는 1개의 특강만을 담당한다.
- * 학생은 같은 이름의 특강을 1개만 신청할 수 있다
- * 후보키는 무엇인가?

■ BCNF 정규형으로 변환

특강수강 릴레이션에서 이상현상을 일으키는 (교수, 특강이름)을 분해함.

특강신청

학생번호	교수
501	김교수
401	김교수
402	승교수
502	박교수
501	홍교수



특강교수

특강이름	교수
소셜네트워크	김교수
인간과 동물	승교수
창업전략	박교수
창업전략	홍교수

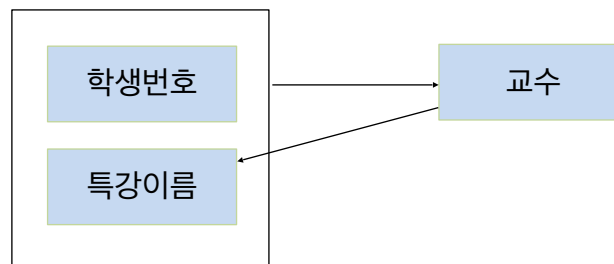


특강수강 릴레이션을 특강신청, 특강교수 릴레이션으로 분해

- 릴레이션 R을 릴레이션 R1과 R2로 분해할 때, $R1 \bowtie R2 = R$ 이면 무손실 분해(lossless-join decomposition)라고 함.
- $R1 \cap R2 \rightarrow R1$ 혹은 $R1 \cap R2 \rightarrow R2$ 중 하나를 만족해야 함.

특강수강

학생번호	특강이름	교수
501	소셜네트워크	김교수
401	소셜네트워크	김교수
402	인간과 동물	송교수
502	창업전략	박교수
501	창업전략	홍교수



특강수강 릴레이션

특강수강 릴레이션의 분해 – 2가지 방법 비교

구분	릴레이션 분해	무손실 분해 여부
[분해1]	특강수강(학생번호, 특강이름, 교수) → R1(학생번호, 교수), R2(교수, 특강이름)	R1과 R2의 공통 속성은 교수이며, 교수는 R2의 키 → 무손실 분해 규칙을 만족
[분해2]	특강수강(학생번호, 특강이름, 교수) → R3(학생번호, 특강이름), R4(교수, 특강이름)	R3과 R4의 공통 속성은 특강이름이지만, 특강이름은 R3나 R4의 키가 아님. → 무손실 분해 규칙을 만족하지 않음

[분해1]의 경우 R1, R2를 다시 조인하면 원래 릴레이션이 됨.

[분해2]의 경우 R3, R4 릴레이션

R3

학생번호	특강이름
501	소셜네트워크
401	소셜네트워크
402	인간과 동물
502	창업전략
501	창업전략

R4

특강이름	교수
소셜네트워크	김교수
인간과 동물	승교수
창업전략	박교수
창업전략	홍교수

특강수강 릴레이션을 R3, R4 릴레이션으로 분해

R3, R4 릴레이션을 다시 조인하면 의미없는 투플이 생김.

-> 무손실 분해 조건을 만족하지 못하고 손실(loss) 분해되었기 때문.

특강수강

학생번호	특강이름	교수
501	소셜네트워크	김교수
401	소셜네트워크	김교수
402	인간과 동물	승교수
502	창업전략	박교수
501	창업전략	홍교수

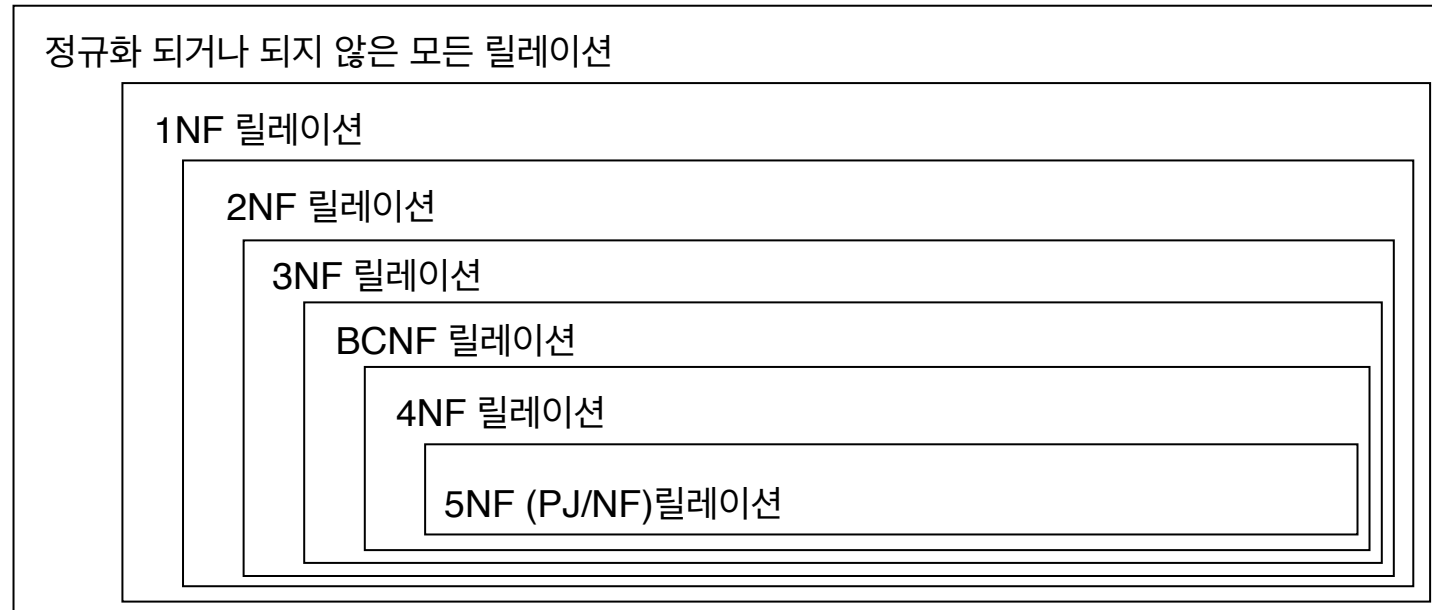
≠

R3 ⋈ R4

학생번호	특강이름	교수
501	소셜네트워크	김교수
401	소셜네트워크	김교수
402	인간과 동물	승교수
502	창업전략	박교수
502	창업전략	홍교수
501	창업전략	박교수
501	창업전략	홍교수

특강수강 릴레이션과 R3 ⋈ R4 릴레이션의 비교

- 대부분의 릴레이션은 BCNF까지 정규화하면 실제적인 이상현상이 없어지기 때문에 보통 BCNF까지 정규화를 진행함.



정규형의 포함 관계

❖ 무결성 제약조건

- 데이터 무결성(integrity, 無缺性)은 데이터베이스에 저장된 데이터의 일관성과 정확성을 지키는 것을 말함.

- 도메인 무결성 제약조건

도메인 제약(domain constraint)이라고도 하며, 릴레이션 내의 튜플들이 각 속성의 도메인에 지정된 값만을 가져야 한다는 조건임. SQL 문에서 데이터 형식(type), 널(null/not null), 기본 값(default), 체크(check) 등을 사용하여 지정할 수 있음.

- 개체 무결성 제약조건

기본키 제약(primary key constraint)이라고도 함. 릴레이션은 기본키를 지정하고 그에 따른 무결성 원칙 즉, 기본키는 NULL 값을 가져서는 안 되며 릴레이션 내에 오직 하나의 값만 존재해야 한다는 조건임.

- 참조 무결성 제약조건

외래키 제약(foreign key constraint)이라고도 하며, 릴레이션 간의 참조 관계를 선언하는 제약조건임. 자식 릴레이션의 외래키는 부모 릴레이션의 기본키와 도메인이 동일해야 하며, 자식 릴레이션의 값이 변경될 때 부모 릴레이션의 제약을 받는다는 것임.

제약조건의 정리

구분	도메인	키	
	도메인 무결성 제약조건	개체 무결성 제약조건	참조 무결성 제약조건
제약 대상	속성	투플	속성과 투플
같은 용어	도메인 제약 (Domain Constraint)	기본키 제약 (Primary Key Constraint)	외래키 제약 (Foreign Key Constraint)
해당되는 키	-	기본키	외래키
NULL 값 허용 여부	허용	불가	허용
릴레이션 내 제약조건의 개수	속성의 개수와 동일	1개	0~여러 개
기타	<ul style="list-style-type: none"> 투플 삽입, 수정 시 제약사항 우선 확인 	<ul style="list-style-type: none"> 투플 삽입/수정 시 제약사항 우선 확인 	<ul style="list-style-type: none"> 투플 삽입/수정 시 제약사항 우선 확인 부모 릴레이션의 투플 수정/삭제 시 제약사항 우선 확인

- 삽입 : 기본키 값이 같으면 삽입이 금지됨.
- 수정 : 기본키 값이 같거나 NULL로도 수정이 금지됨.
- 삭제 : 특별한 확인이 필요하지 않으며 즉시 수행함.

학번	이름	학과코드
501	박지성	1001
401	김연아	2001
402	장미란	2001
502	추신수	1001

학생 릴레이션

(501, 남슬찬, 1001)

➡ 삽입 거부

학번	이름	학과코드
501	박지성	1001
401	김연아	2001
402	장미란	2001
502	추신수	1001

(NULL, 남슬찬, 1001)

➡ 삽입 거부

학번	이름	학과코드
501	박지성	1001
401	김연아	2001
402	장미란	2001
502	추신수	1001

개체 무결성 제약조건의 수행 예(기본키 충돌 및 NULL 값 삽입)

■ 삽입

- 학과(부모 릴레이션) : 튜플 삽입한 후 수행하면 정상적으로 진행된다.
- 학생(자식 릴레이션) : 참조받는 테이블에 외래키 값이 없으므로 삽입이 금지된다.

학생(자식 릴레이션)

학번	이름	학과코드
501	박지성	1001
401	김연아	2001
402	장미란	2001
502	추신수	1001

학과(부모 릴레이션)

학과코드	학과명
1001	컴퓨터학과
2001	체육학과

참조

학생관리 데이터베이스

■ 삭제

- 학과(부모 릴레이션) : 참조하는 테이블을 같이 삭제할 수 있어서 금지하거나 다른 추가 작업이 필요함.
- 학생(자식 릴레이션) : 바로 삭제 가능함.

※ 부모 릴레이션에서 튜플을 삭제할 경우 참조 무결성 조건을 수행하기 위한 고려사항

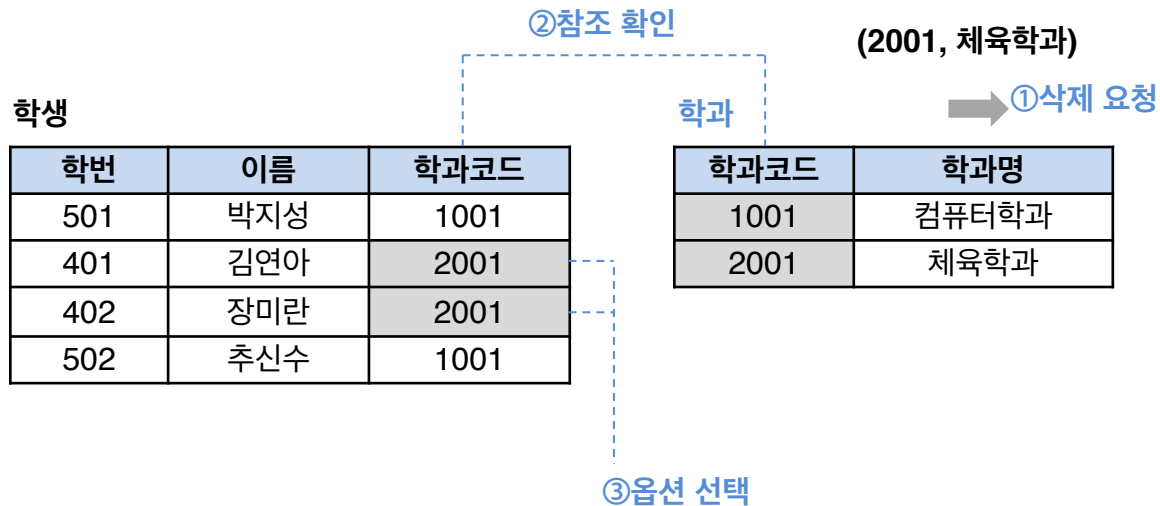
- ① 즉시 작업을 중지
- ② 자식 릴레이션의 관련 튜플을 삭제
- ③ 초기에 설정된 다른 어떤 값으로 변경
- ④ NULL 값으로 설정

■ 수정

- 삭제와 삽입 명령이 연속해서 수행됨.
- 부모 릴레이션의 수정이 일어날 경우 삭제 옵션에 따라 처리된 후 문제가 없으면 다시 삽입 제약조건에 따라 처리됨.

참조 무결성 제약조건의 옵션(부모 릴레이션에서 튜플을 삭제할 경우)

명령어	의미	예
RESTRICTED	자식 릴레이션에서 참조하고 있을 경우 부모 릴레이션의 삭제 작업을 거부함	학과 릴레이션의 튜플 삭제 거부
CASCADE	자식 릴레이션의 관련 튜플을 같이 삭제 처리함	학생 릴레이션의 관련 튜플을 삭제
DEFAULT	자식 릴레이션의 관련 튜플을 미리 설정해둔 값으로 변경함	학생 릴레이션의 학과가 다른 학과로 자동 배정
NULL	자식 릴레이션의 관련 튜플을 NULL 값으로 설정함 (NULL 값을 허가한 경우)	학과 릴레이션의 학과가 NULL 값으로 변경



- ① RESTRICTED : 요청한 삭제 작업중지(에러 처리)
- ② CASCADE : 학생 릴레이션의 해당 튜플을 같이 연쇄적으로 삭제(CASCADE)
- ③ 기본값으로 변경(미리 설정한 값, DEFAULT)
- ④ NULL 값으로 설정

참조 무결성 제약조건에서 부모 릴레이션의 튜플을 삭제할 경우

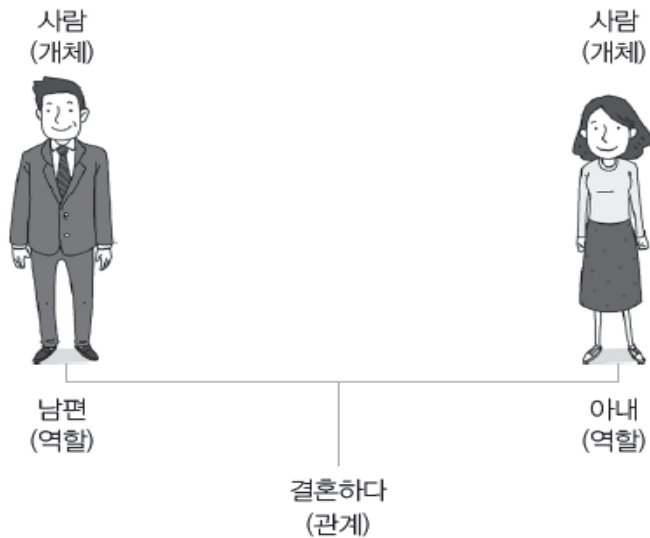
- 개체와 개체 타입
- 속성
- 관계와 관계 타입
- 약한 개체 타입과 식별자
- IE 표기법

■ ER(Entity Relationship) 모델

- 세상의 사물을 개체(entity)와 개체 간의 관계(relationship)로 표현함.

■ 개체

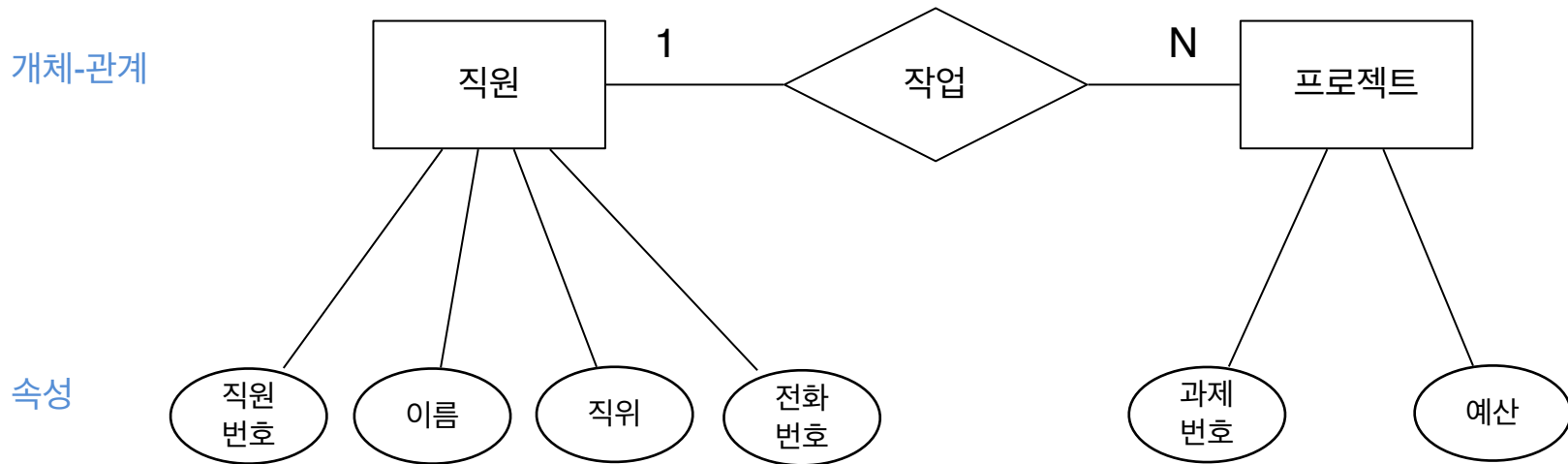
- 독립적인 의미를 지니고 있는 유무형의 사람 또는 사물
- 개체의 특성을 나타내는 속성(attribute)에 의해 식별됨. 개체끼리 서로 관계를 가짐.



ER 모델의 기본 개념

■ ER 다이어그램이란?

- ER 모델은 개체와 개체 간의 관계를 표준화된 그림으로 나타냄.

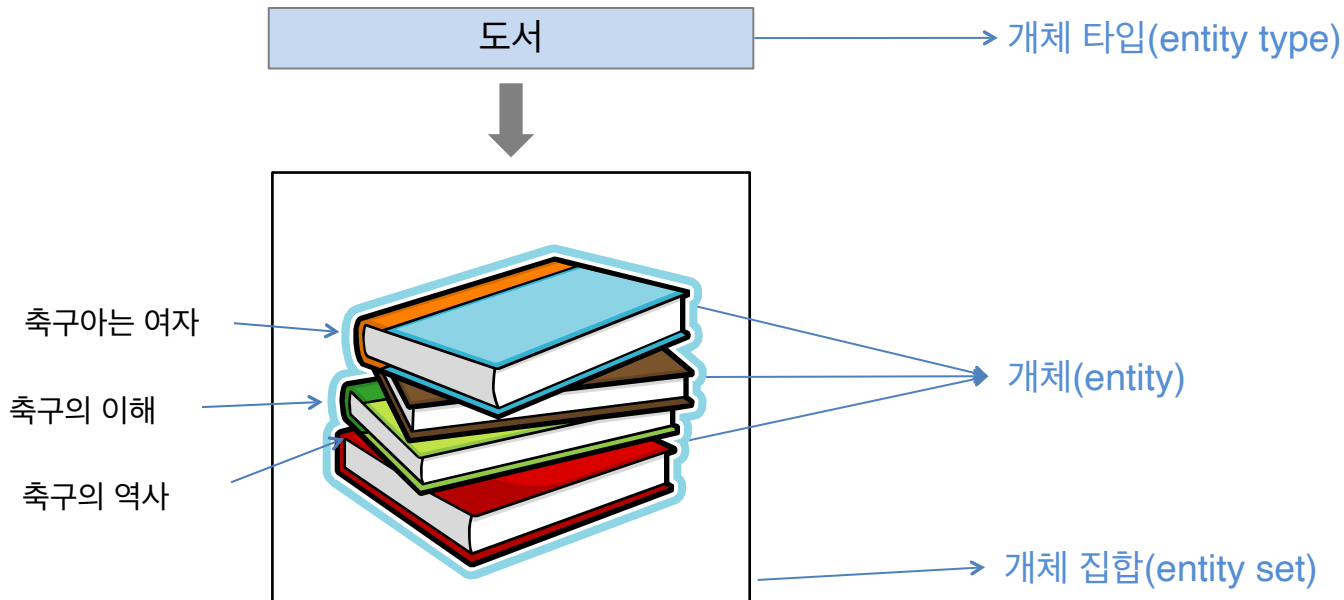


ER 다이어그램

■ 개체(entity)란?


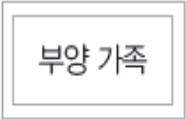
- 사람, 사물, 장소, 개념, 사건과 같이 유무형의 정보를 가지고 있는 독립적인 실체.
- 데이터베이스에서 주로 다루는 개체는 낱개로 구성된 것, 낱개가 각각 데이터 값을 가지는 것, 데이터 값이 변하는 것 등이 있음.
- 비슷한 속성의 개체 타입(entity type)을 구성하며, 개체 집합(entity set)으로 묶임.

개체, 개체 타입, 개체 집합



- ER 다이어그램상에서 개체 타입은 직사각형으로 나타냄.

개체 타입의 ER 다이어그램 표현

기호	의미
 직원	강한 개체 타입(보통 개체 타입이라고 하면 강한 개체 타입을 말한다)
 부양 가족	약한 개체 타입

- 개체 타입의 유형
 - 강한 개체(strong entity) : 다른 개체의 도움 없이 독자적으로 존재할 수 있는 개체
 - 약한 개체(weak entity) : 독자적으로는 존재할 수 없고 반드시 상위 개체 타입을 가짐.

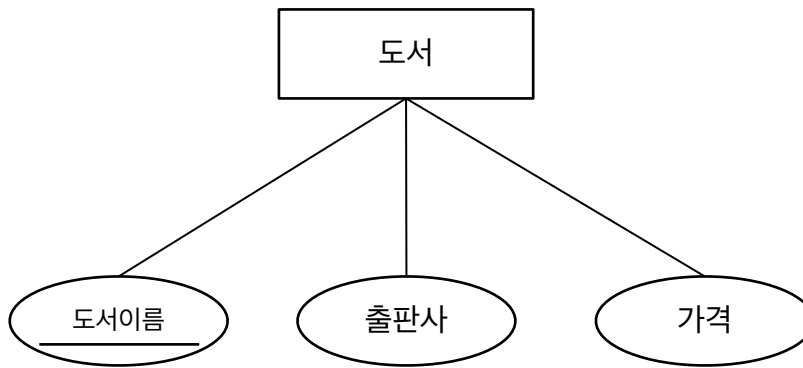
■ 속성(attribute) : 개체가 가진 성질

개체 타입	속성
도서	도서이름, 출판사, 도서단가

표 6-2 개체 타입과 속성

■ 속성의 ER 다이어그램 표현

- 속성은 기본적으로 타원으로 표현. 개체 타입을 나타내는 직사각형과 실선으로 연결됨.
- 속성의 이름은 타원의 중앙에 표기함.
- 속성이 개체를 유일하게 식별할 수 있는 키일 경우 속성 이름에 밑줄을 그음.

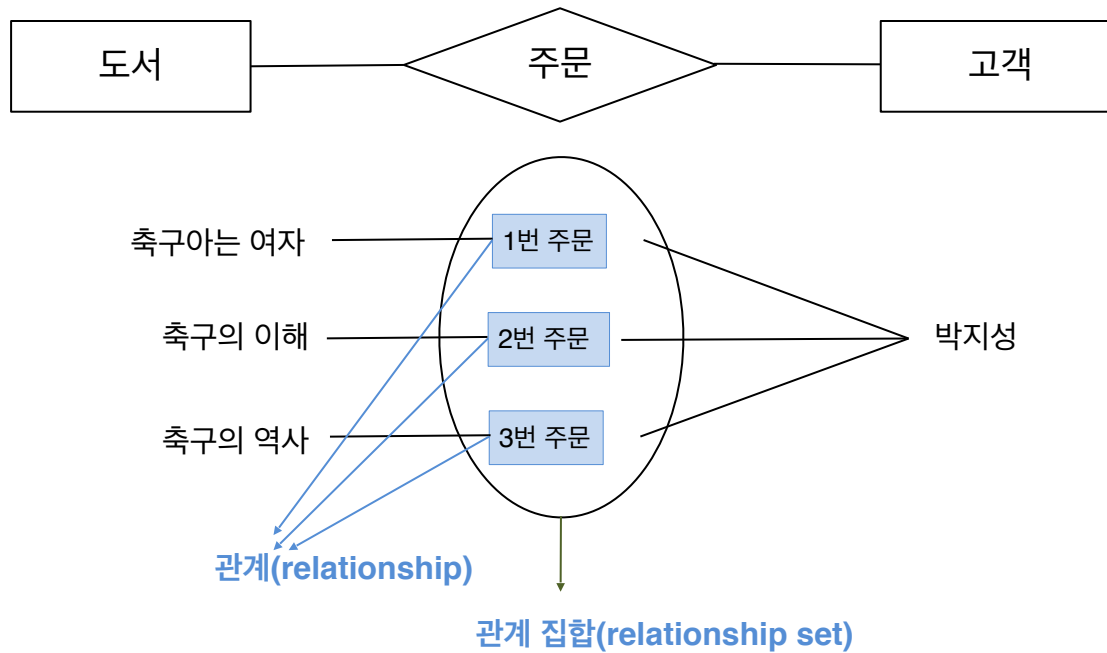


도서 개체 타입

속성의 ER 다이어그램 표현


기호	의미	설명
	속성	<ul style="list-style-type: none"> 일반적인 속성을 나타냄 속성의 이름은 타원 중앙에 표시
	키(key) 속성	<ul style="list-style-type: none"> 속성이 개체를 유일하게 식별할 수 있는 키일 경우 속성 이름에 밑줄을 그음
	약한 개체의 식별자	<ul style="list-style-type: none"> 약한 개체는 키를 갖지 못하고 대신 식별자를 가짐 식별자의 아래에 점선을 그음
	다중값 속성	<ul style="list-style-type: none"> 취미(수영, 자전거)와 같이 여러 개의 값을 갖는 속성 이중 타원으로 표현
	유도 속성	<ul style="list-style-type: none"> 나이와 같이 출생년도로 유도가 가능한 속성 점선 타원으로 표현
	복합 속성	<ul style="list-style-type: none"> 주소(시, 동, 번지)와 같이 여러 속성으로 구성된 속성 큰 타원 아래 작은 타원으로 연결

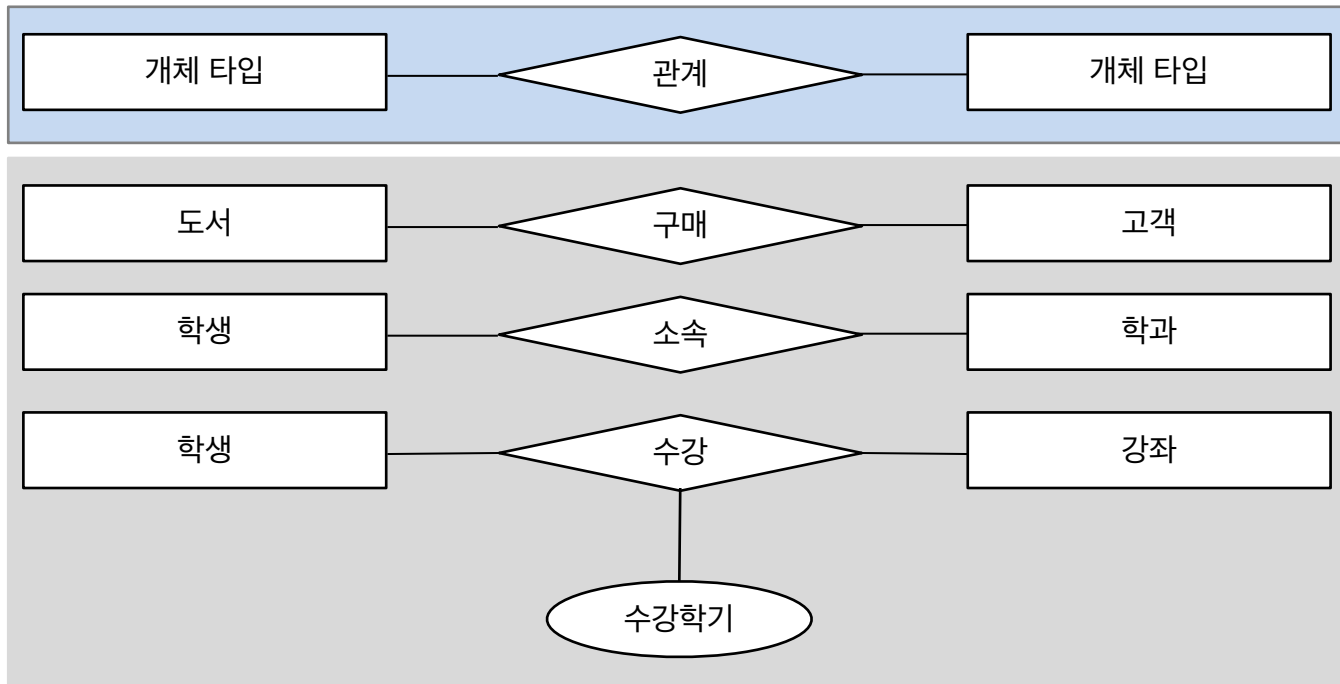
- 관계(relationship) : 개체 사이의 연관성을 나타내는 개념.
- 관계 타입(relationship type) : 개체 타입과 개체 타입 간의 연결 가능한 관계를 정의한 것이며, 관계 집합(relationship set)은 관계로 연결된 집합을 의미함.



관계, 관계 타입, 관계 집합

관계 타입의 ER 다이어그램 표현

기호	의미
	관계 타입


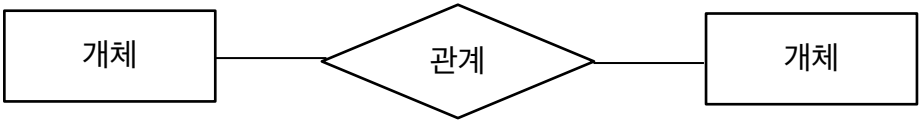
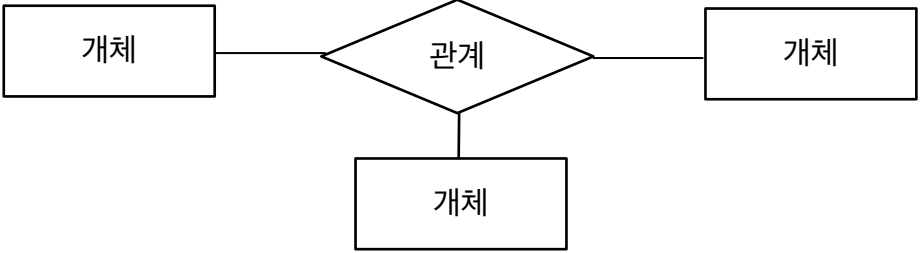


관계의 예

■ 차수에 따른 유형

관계 집합에 참가하는 개체 타입의 수를 관계 타입의 차수(degree)라고 함.

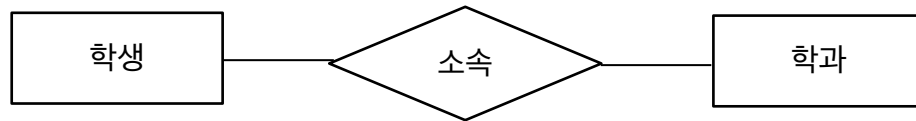
차수에 따른 관계 타입의 유형

기호	의미	설명
	1진 관계	한 개의 개체가 자기 자신과 관계를 맺음
	2진 관계	두 개의 개체가 관계를 맺음
	3진 관계	세 개의 개체가 관계를 맺음

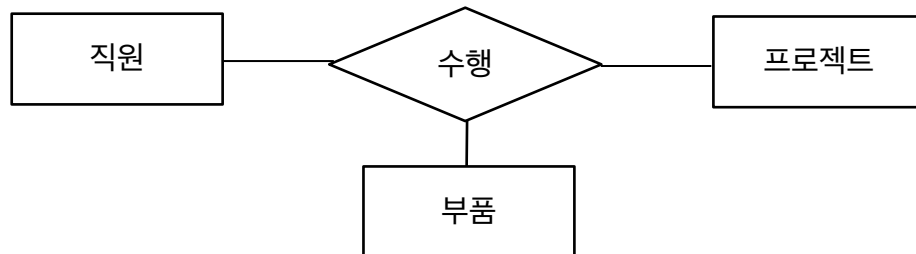
① 1진 관계(recursive relationship) : 한 개의 개체가 자기 자신과 관계를 맺는 경우



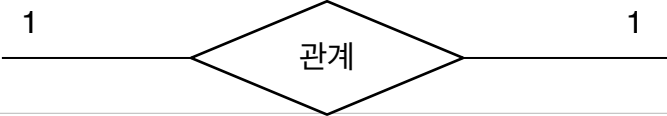
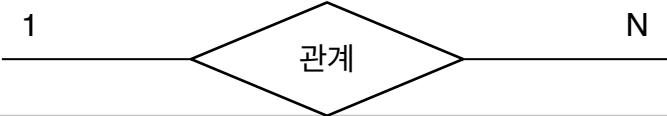
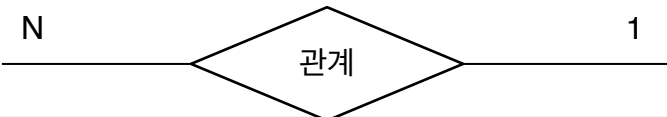
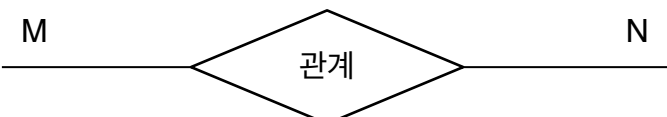
② 2진 관계(binary relationship) : 두 개의 개체가 관계를 맺는 경우



③ 3진 관계(ternary relationship) : 세 개의 개체가 관계를 맺는 경우



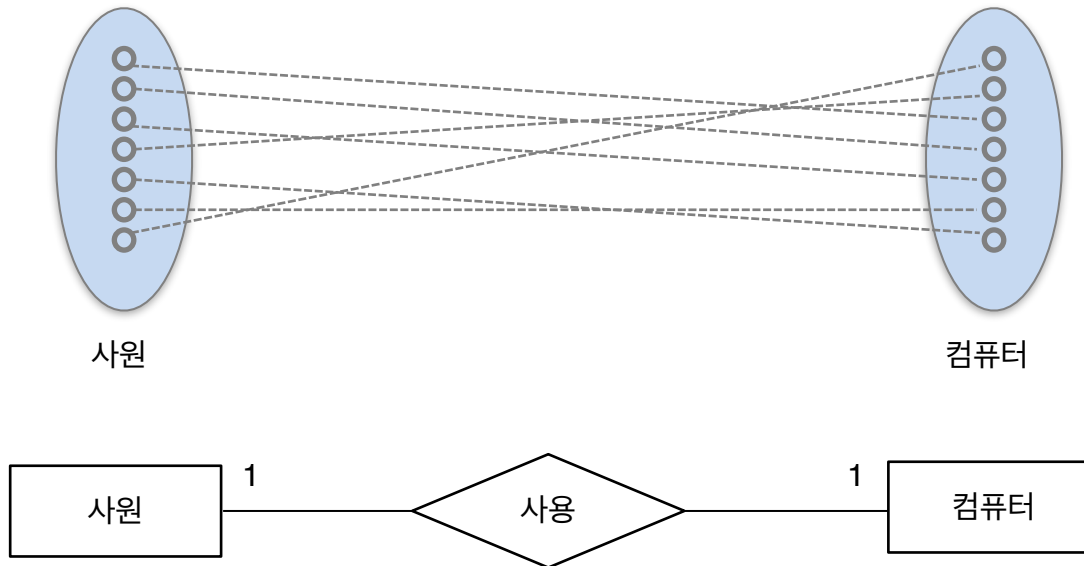
- 관계 대응수(cardinality) : 두 개체 타입의 관계에 실제로 참여하는 개별 개체 수

기호	의미	설명
	일대일 관계	하나의 개체가 하나의 개체에 대응
	일대다 관계	하나의 개체가 여러 개체에 대응
	다대일 관계	여러 개체가 하나의 개체에 대응
	다대다 관계	여러 개체가 여러 개체에 대응

관계 대응수에 따른 관계 타입의 유형

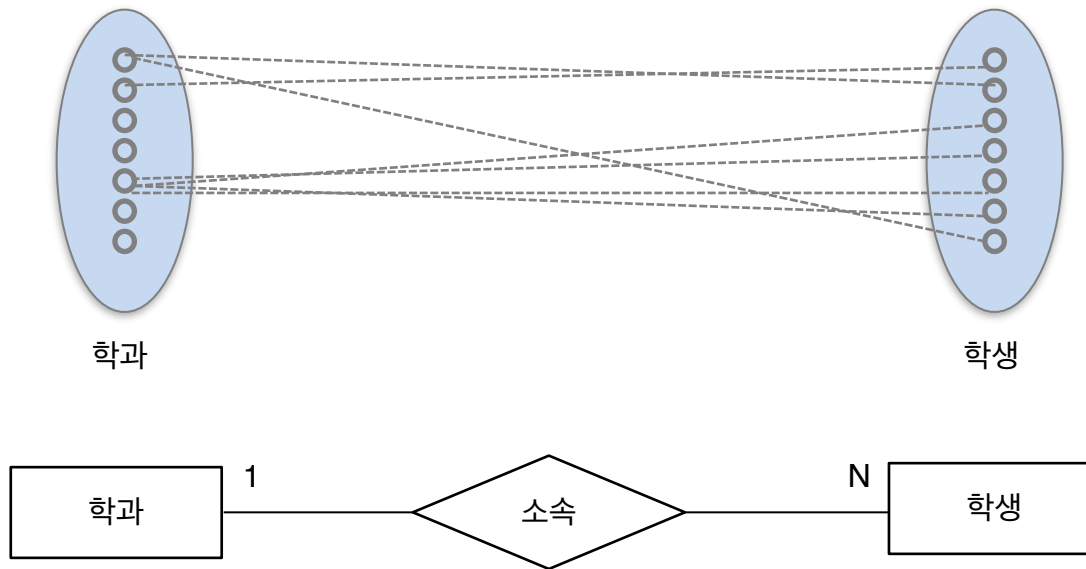
① 일대일(1:1)관계

좌측 개체 타입에 포함된 개체가 우측 개체 타입에 포함된 개체와 일대일로 대응하는 관계



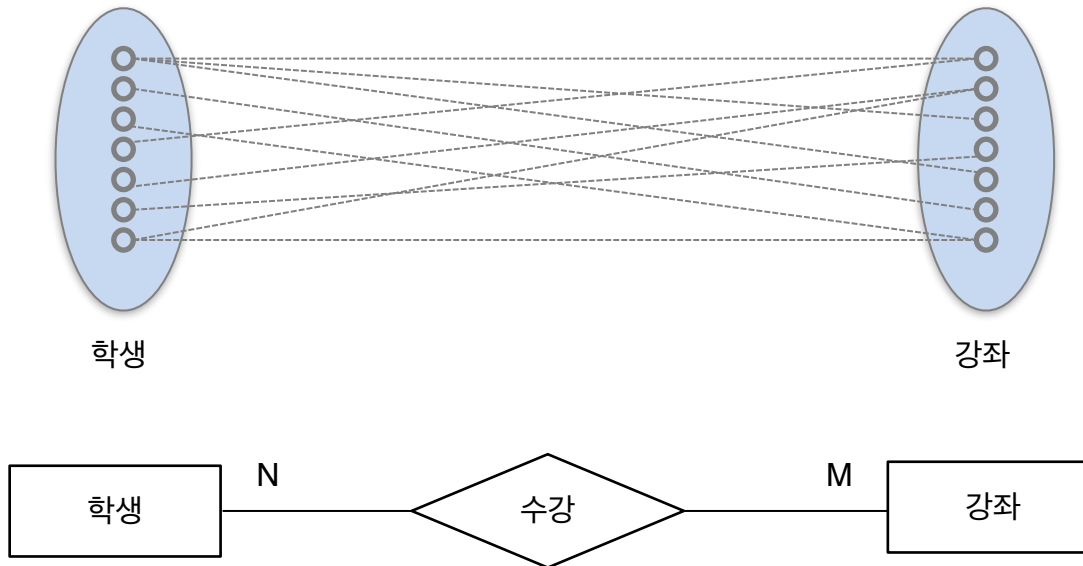
② 일대다(1:N), 다대일(N:1) 관계

실제 일상생활에서 가장 많이 볼 수 있는 관계로, 한쪽 개체 타입의 개체 하나가 다른 쪽 개체 타입의 여러 개체와 관계를 맺음.



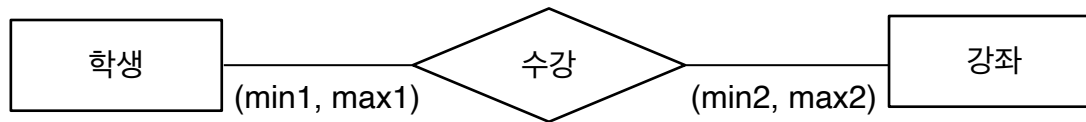
③ 다대다(N:M) 관계

각 개체 타입의 개체들이 서로 임의의 개수의 개체들과 서로 복합적인 관계를 맺고 있는 관계를 말함.



■ 관계 대응수의 최솟값과 최댓값

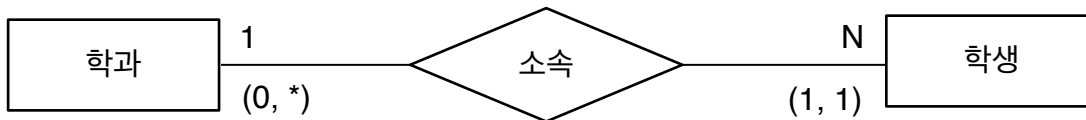
- 관계 대응수 1:1, 1:N, M:N에서 1, N, M은 각 개체가 관계에 참여하는 최댓값을 의미함.
- 관계에 참여하는 개체의 최솟값을 표시하지 않는다는 단점을 보완하기 위해 다이어그램에서는 대응수 외에 최솟값과 최댓값을 관계실선 위에 (최솟값, 최댓값)으로 표기함.



관계 대응수의 최솟값과 최댓값의 표기


관계 대응수에 따른 관계 타입의 유형

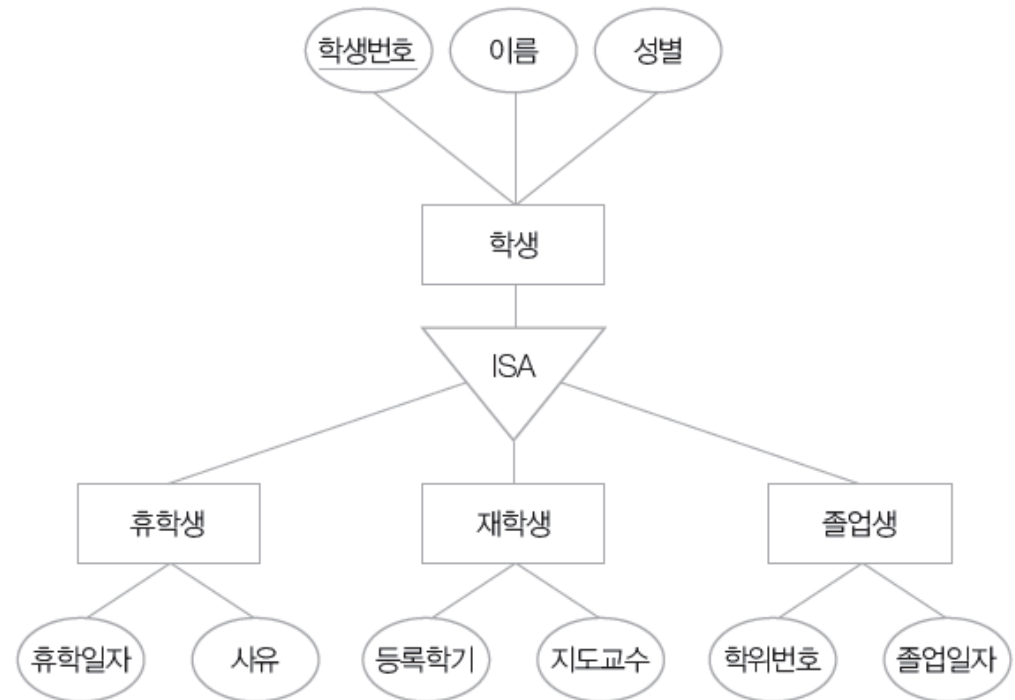
관계	(min1,max1)	(min2,max2)
1:1	(0, 1)	(0, 1)
1:N	(0, *)	(0, 1)
M:N	(0, *)	(0, *)



- 상위 개체 타입의 특성에 따라 하위 개체 타입이 결정되는 형태

ISA 관계 (ISA => is-a)

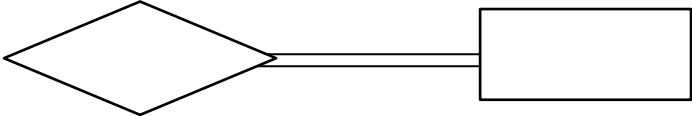
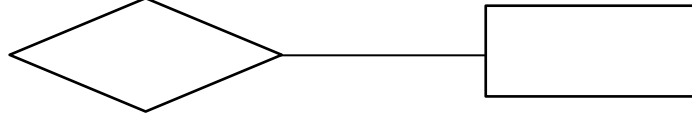
기호	의미
	개체 간 상하관계

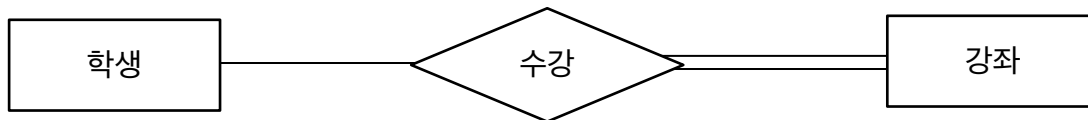


ISA 관계의 예

- 개체 집합 내 모든 개체가 관계에 참여하는지 유무에 따라 전체 참여와 부분 참여로 구분 가능.
- 전체 참여는 개체 집합의 모든 개체가, 부분 참여는 일부만 참여함.
- 전체 참여를 (최솟값, 최댓값)으로 표현할 경우 최솟값이 1 이상으로 모두 참여한다는 뜻이고, 부분 참여는 최솟값이 0 이상이다.

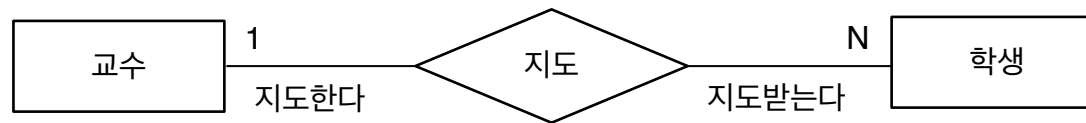
관계의 참여 제약 조건

기호	의미
	전체 참여
	부분 참여



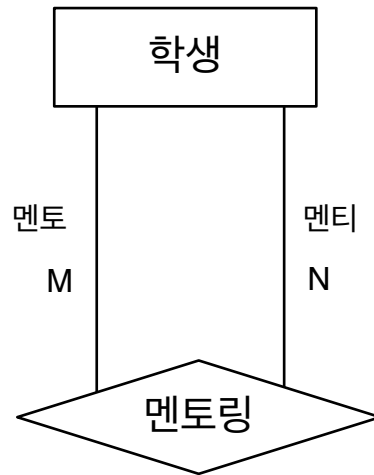
부분 참여와 전체 참여의 예

- 개체 타입 간의 관계를 표현할 때 각 개체들은 고유한 역할(role)을 담당함.

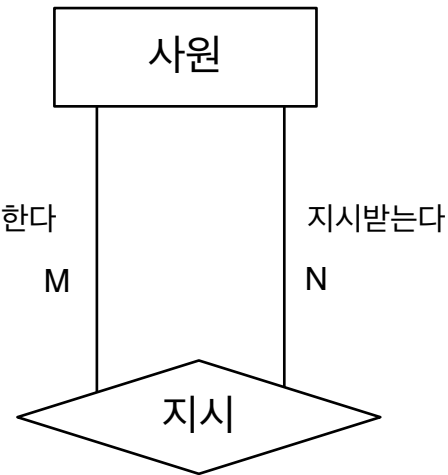


역할의 예

- 순환적 관계(recursive relationship) : 하나의 개체 타입이 동일한 개체 타입(자기 자신)과 순환적으로 관계를 가지는 형태.



(a) 학생의 멘토링 관계

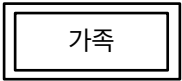
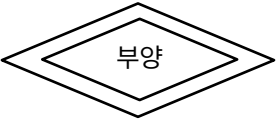
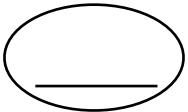
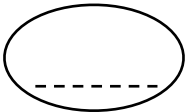


(b) 사원의 지시 관계

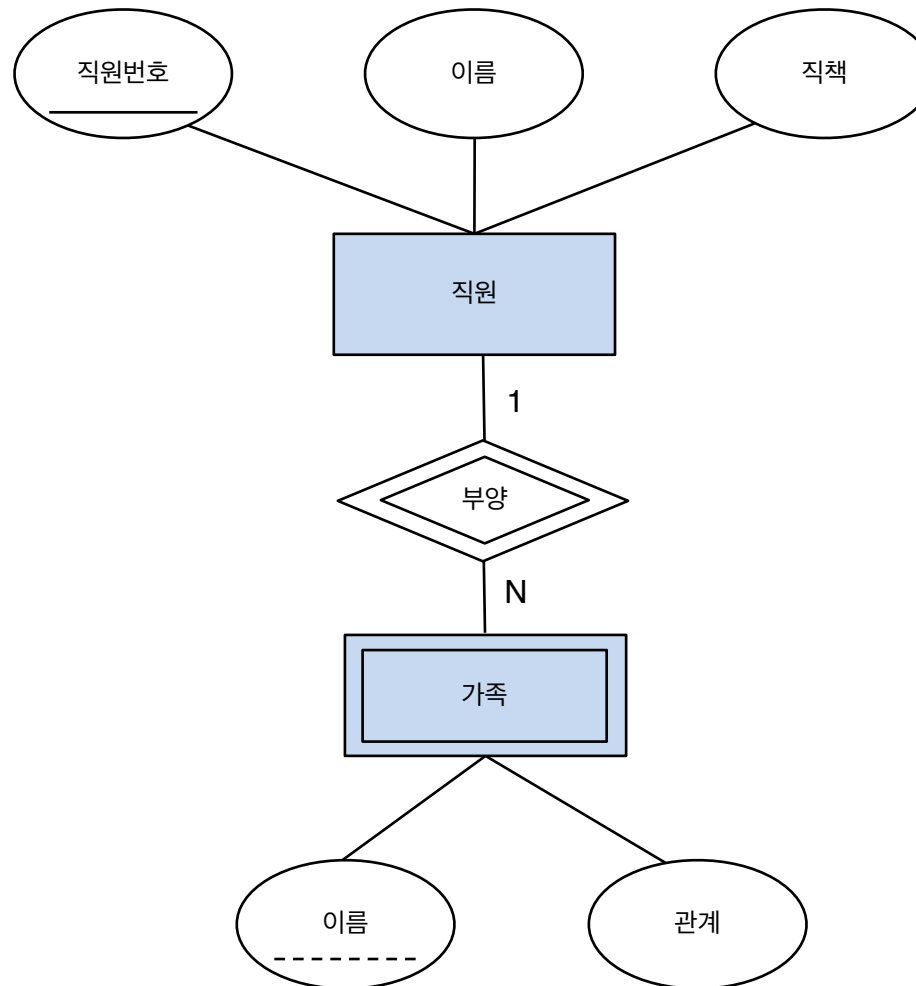
순환적 관계의 예

- 약한 개체(weak entity) 타입 : 상위 개체 타입이 결정되지 않으면 개별 개체를 식별할 수 없는 종속된 개체 타입
- 약한 개체 타입은 독립적인 키로는 존재할 수 없지만 상위 개체 타입의 키와 결합하여 약한 개체 타입의 개별 개체를 고유하게 식별하는 속성을 식별자(discriminator) 혹은 부분키(partial key)라고 함.

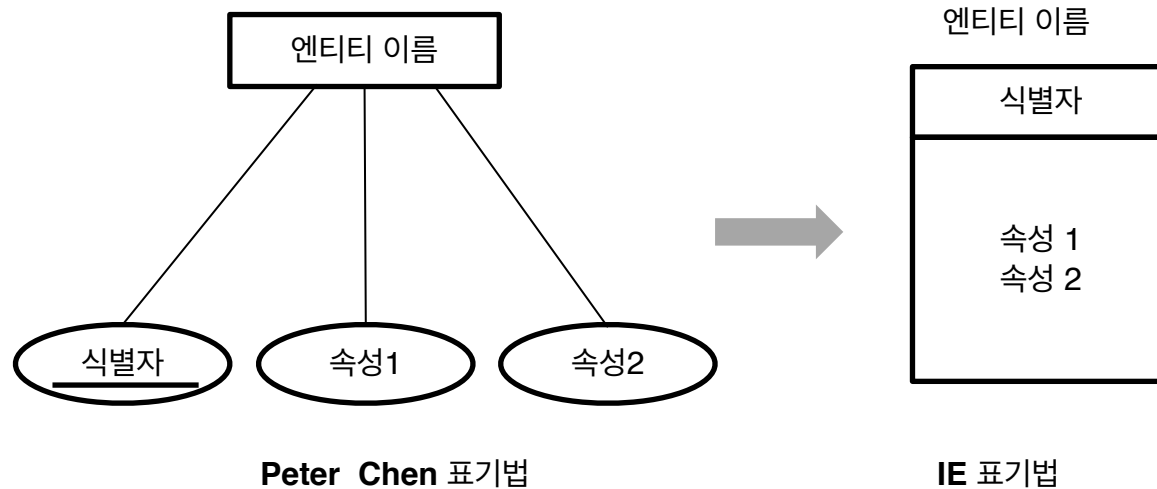
식별자와 약한 개체 타입

기호	의미	설명
	약한 개체 타입	<ul style="list-style-type: none"> • 강한 개체 타입이 있어야 존재할 수 있음 • 이중 직사각형으로 표현
	식별 관계 타입	<ul style="list-style-type: none"> • 강한 개체 타입과 약한 개체 타입의 관계를 나타냄 • 강한 개체 타입의 기본키를 상속받아 사용함 • 이중 마름모꼴로 표현
	키	<ul style="list-style-type: none"> • 강한 개체 타입의 키 속성
	식별자	<ul style="list-style-type: none"> • 약한 개체 타입에서 개별 개체를 구분하는 속성 • 키라고 하지 않고 식별자라고 부름

약한 개체 타입과 식별자의 예



- ER 다이어그램을 더 축약하여 쉽게표현하면 Erwin 등 소프트웨어에서 사용함.
- IE 표기법에서 개체 타입과 속성은 직사각형으로 표현함.



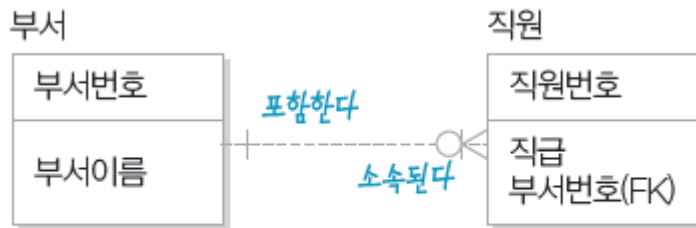
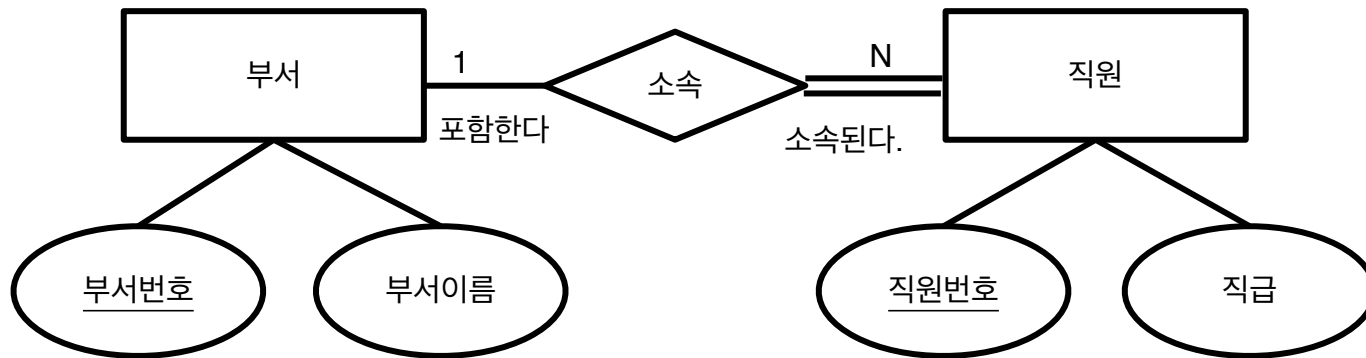
Peter Chen 표기법과 IE 표기법

■ IE 표기법에서 관계는 실선 혹은 점선으로 표기함

IE 표기법 – 관계와 관계 대응수

기호	의미
-----	<ul style="list-style-type: none"> • 비식별자 관계(non-identifying relationship): 강한 개체 타입 • 부모 개체의 키가 일반 속성으로 포함되는 관계
_____	<ul style="list-style-type: none"> • 식별자 관계(identifying relationship): 약한 개체 타입 • 부모 개체의 키가 주식별자로 포함되는 관계
—————<	<ul style="list-style-type: none"> • 일대다(1:N)의 관계 : N 쪽에 새발을 표시
—————○	<ul style="list-style-type: none"> • 0(선택 참여), 최소 참여가 0일 경우
—————+	<ul style="list-style-type: none"> • 1(필수 참여), 최소 참여가 1일 경우

- IE 표기법에서 관계(강한관계, 비식별자 관계)는 점선으로 표기함



(b) IE 표기법으로 작성한 직원-부서 관계

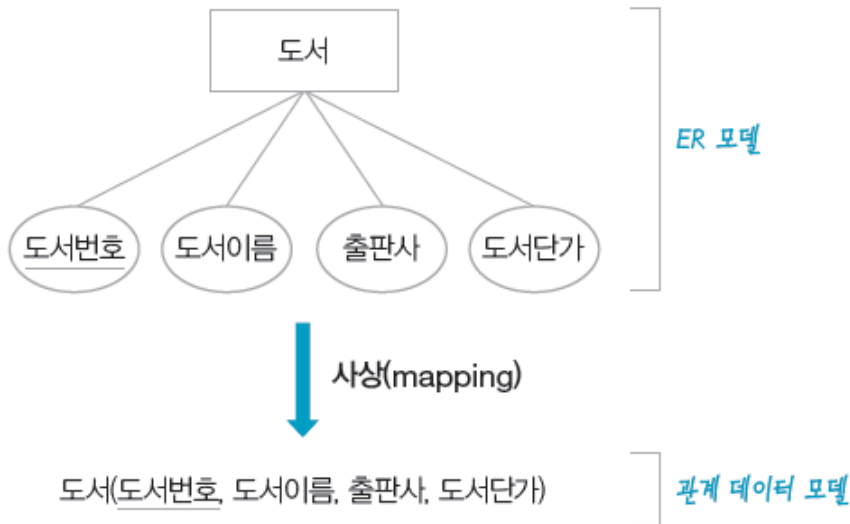
IE 표기법의 예(비식별자 관계)

- IE 표기법에서 관계(약한관계, 식별자 관계)는 실선으로 표기함



IE 표기법의 예(식별자 관계)

- 완성된 ER 모델은 실제 데이터베이스로 구축하기 위해 논리적 모델링 단계를 거치는데, 이 단계에서 사상(mapping)이 이루어짐.

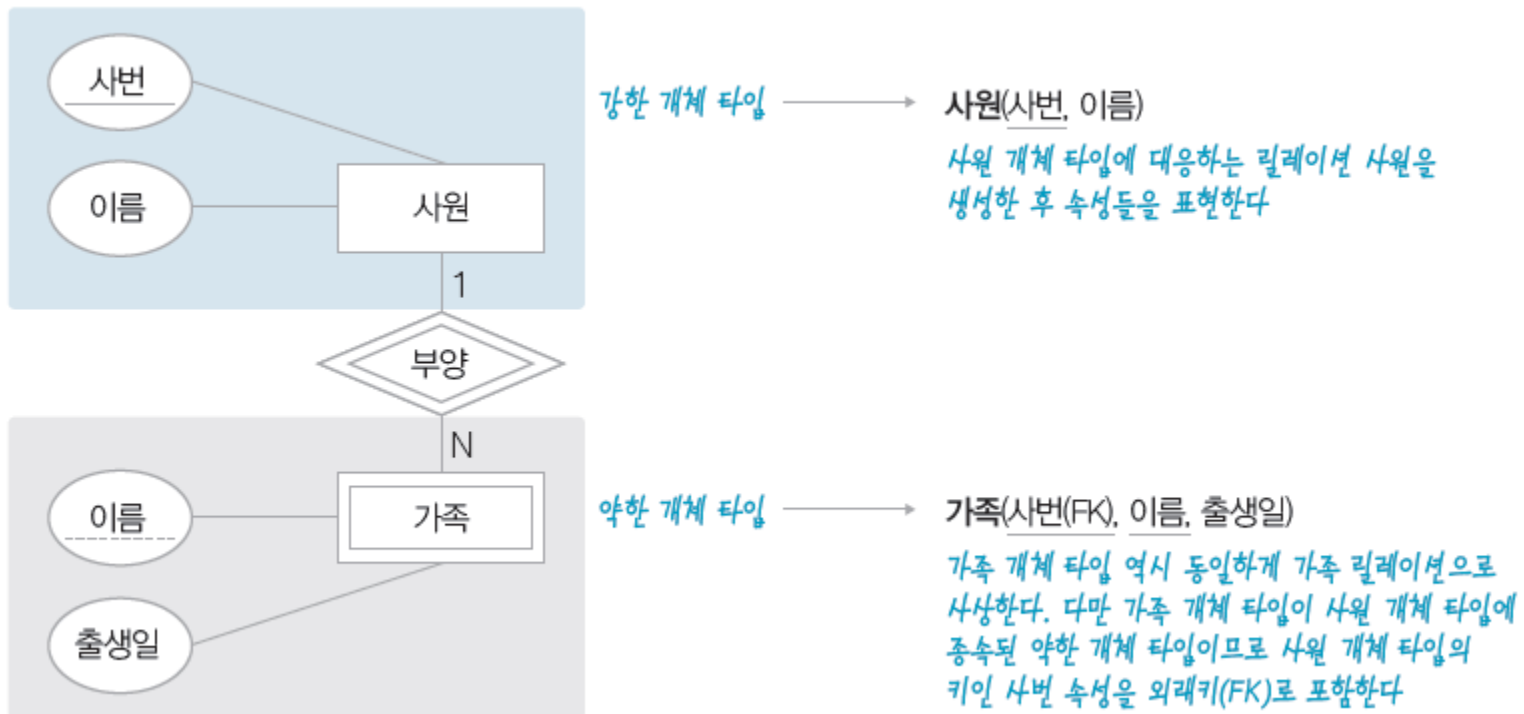


ER 모델을 관계 데이터 모델로 사상

단계	사상할 대상	구분
1단계	개체 타입	강한 개체 타입
2단계		약한 개체 타입
3단계	관계 타입	이진 1:1 관계 타입
4단계		이진 1:N 관계 타입
5단계		이진 N:M 관계 타입
6단계		N진 관계 타입
7단계	속성	다중값 속성

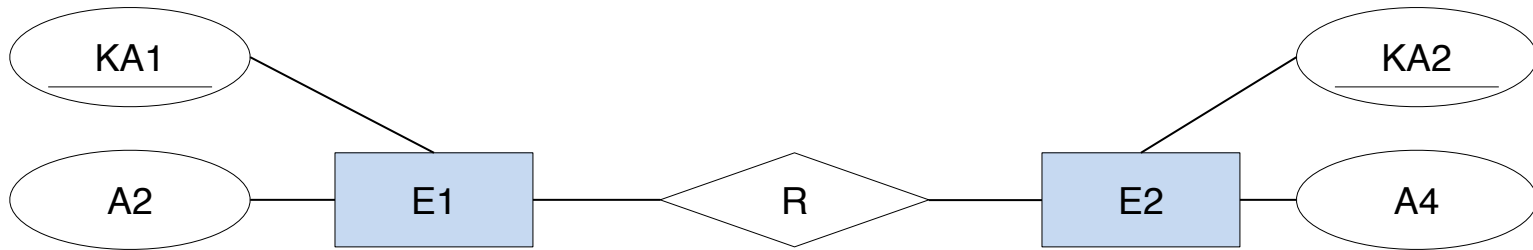
ER 모델과 관계 데이터 모델의 사상 알고리즘

- **[1단계] 강한(정규) 개체 타입** 정규 개체 타입 E의 경우 대응하는 릴레이션 R을 생성함.
- **[2단계] 약한 개체 타입** : 약한 개체 타입에서 생성된 릴레이션은 자신의 키와 함께 강한 개체 타입의 키를 외래키로 사상하여 자신의 기본키를 구성함.



개체 타입의 사상

이진 관계 타입



[방법1] 오른쪽 개체 타입 E2를 기준으로 관계 R을 표현한다.

E1(KA1, A2)

E2(KA2, A4, KA1)

[방법2] 왼쪽 개체 타입 E1을 기준으로 관계 R을 표현한다.

E1(KA1, A2, KA2)

E2(KA2, A4)

[방법3] 단일 릴레이션 ER로 모두 통합하여 관계 R을 표현한다.

ER(KA1, A2, KA2, A4)

[방법4] 개체 타입 E1, E2와 관계 타입 R을 모두 독립된 릴레이션으로 표현한다.

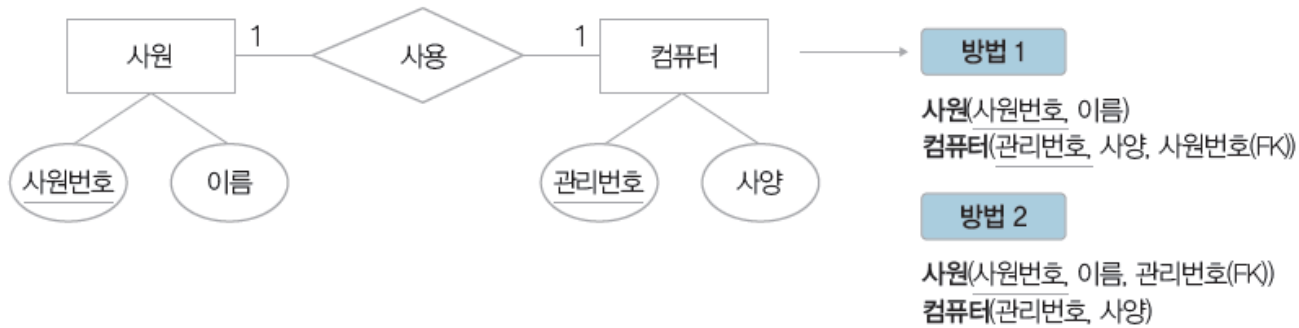
E1(KA1, A2)

R(KA1, KA2)

E2(KA2, A4)

■ [3단계] 이진 1:1 관계 타입

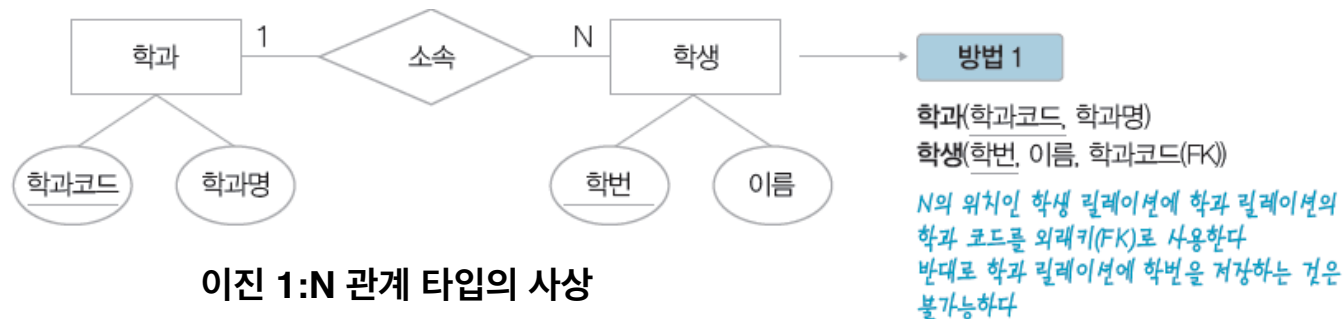
이진 1:1 관계 타입의 경우 [방법1]~[방법4]까지 모든 유형으로 사상 가능. 개체가 가진 정보 유형에 따라 판단.



이진 1:1 관계 타입의 사상

■ [4단계] 이진 1:N 관계 타입

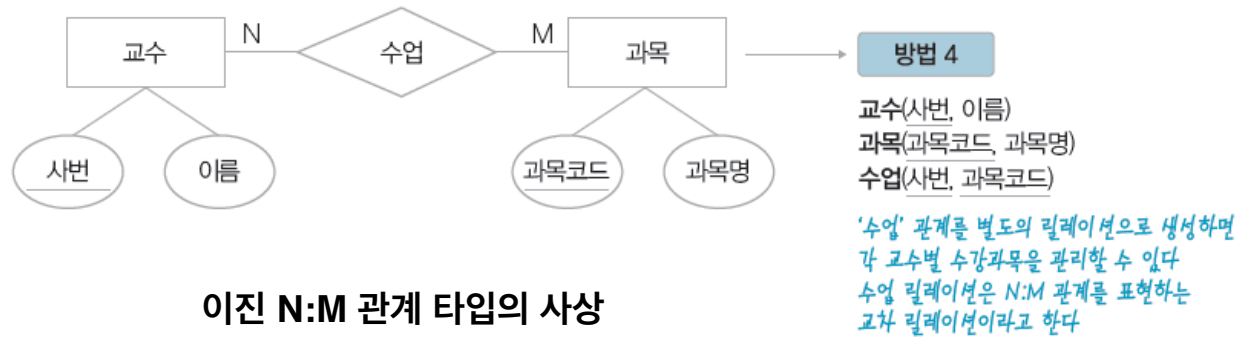
이진 1:N 관계 타입의 경우 N의 위치에 따라 [방법1] 또는 [방법2]의 유형으로 사상됨.



이진 1:N 관계 타입의 사상

■ [5단계] 이진 M:N 관계 타입

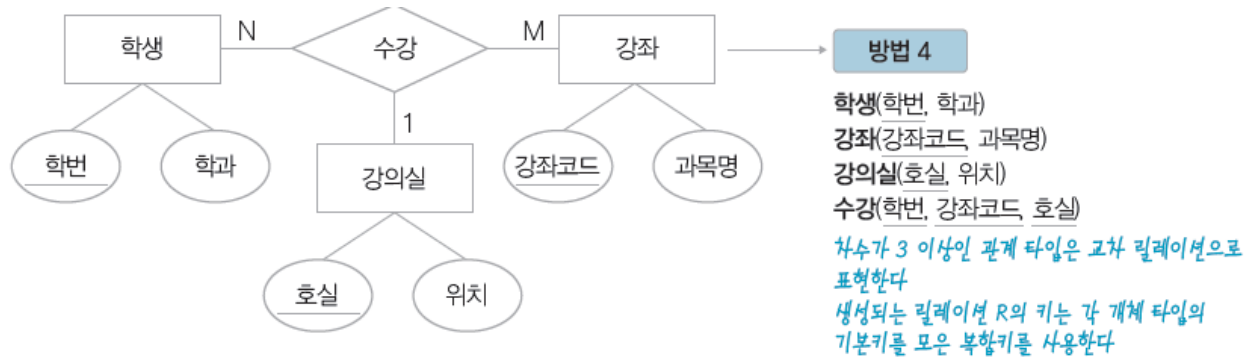
이진 M:N 관계 타입은 [방법4]의 유형으로 사상됨.



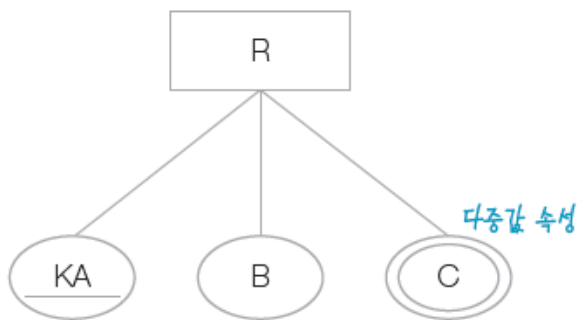
이진 N:M 관계 타입의 사상

■ [6단계] N진 관계 타입

ER 모델의 차수가 3 이상인 다진 관계 타입의 경우 [방법4]의 유형으로 사상된다.



이진 N진 관계 타입의 사상



[방법 1] C 값의 수를 알 수 없는 경우

$R(\underline{KA}, B)$
 $RC(\underline{KA}, \underline{C})$

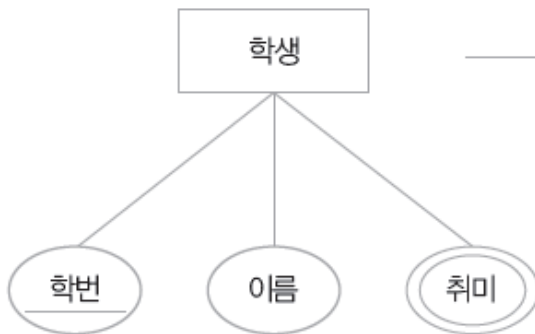
[방법 2] C 값의 수가 세 개 이하일 경우

$R(\underline{KA}, B, C_1, C_2, C_3)$

다중값 속성의 개수에 따른 사상 방법

■ [7단계]

속성의 개수를 알 수 없는 경우 [방법1]을, 속성의 개수가 제한적으로 정해지는 경우 [방법2]를 사용함.



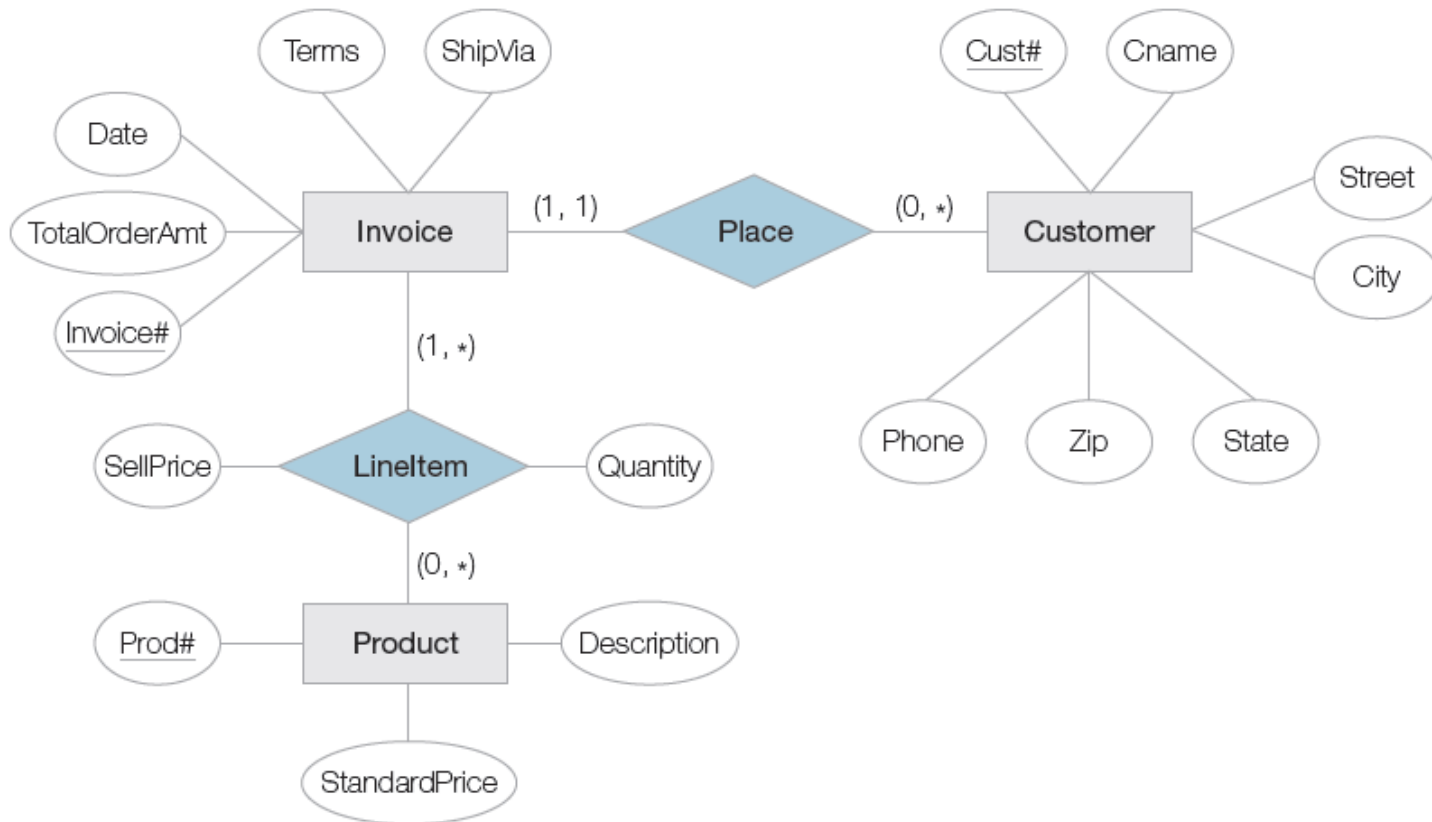
방법 1

학생(학번, 이름)
취미(학번, 취미이름)

한 학생이 게임, 등산, 스키, 노래부르기 등
여러 취미가 있으므로 독립적인 릴레이션을
만들어 저장한다

다중값 속성의 사상

8 다음은 고객과 주문에 관한 ER 다이어그램이다. 개체는 고객(Customer), 제품(Product), 주문(Invoice)으로 구성된다. Place 관계는 '주문한다'를, LineItem은 '주문 항목'을 의미한다. 그림에 해당하는 테이블을 작성하시오(변환된 테이블의 기본키는 밑줄 실선, 외래키는 밑줄 점선으로 표시한다. 기본키인 동시에 외래키일 경우에는 밑줄 실선으로 표시한다. 테이블 변환을 위하여 필요한 사항 중 설명되지 않은 것은 임의로 정하여 설계한다).

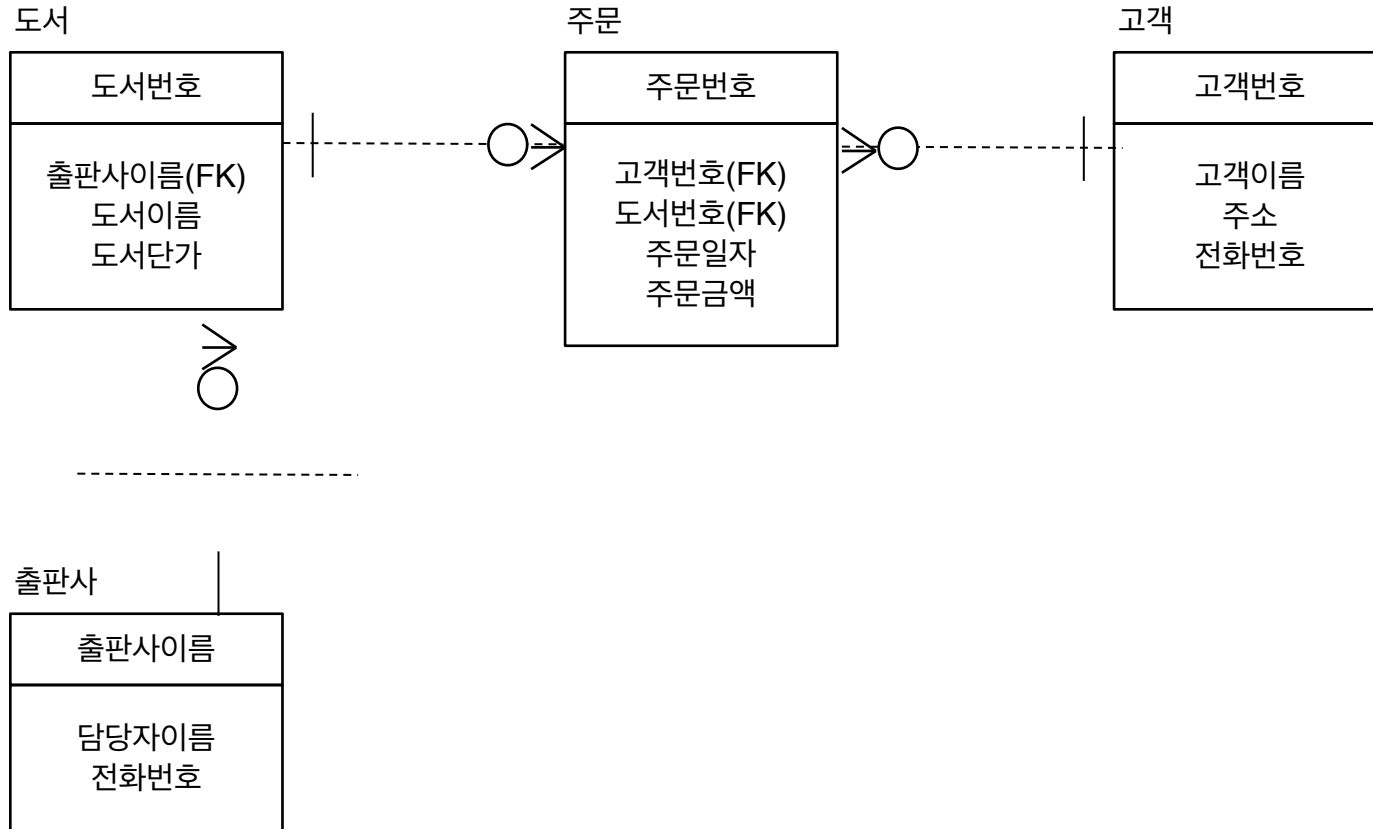


- ERwin 기본 화면 및 툴 둘러보기
- ERwin 실습을 위한 기본 환경 설정하기
- 마당서점 설계 실습
- DBMS에 접속하여 테이블 생성하기

<참고>

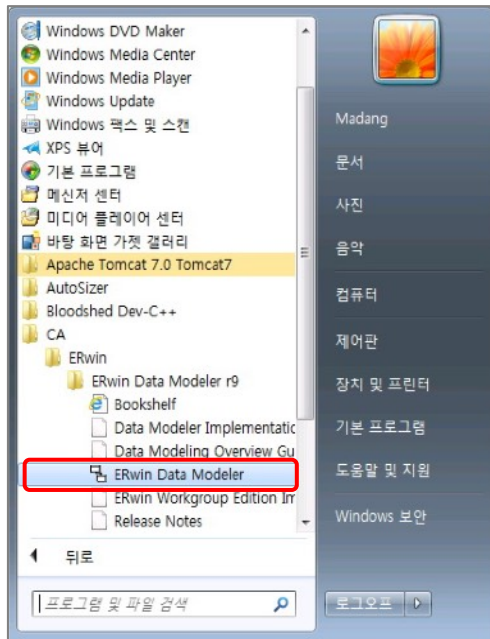
Erwin(<http://erwin.com>) 데이터 모델링 도구는 미국 CA Technologies 회사에서 만든 제품으로 시험용 버전인 Community Edition을 제공한다. 본 교재는 Erwin r9 Community Edition으로 작성되었다. 무료이지만 유효 기간이 있어서 일정 시간이 지나면 다시 내려 받아야 한다.

- ERwin : 데이터 모델링을 하기 위한 프로그램. IE 표기법을 지원함.

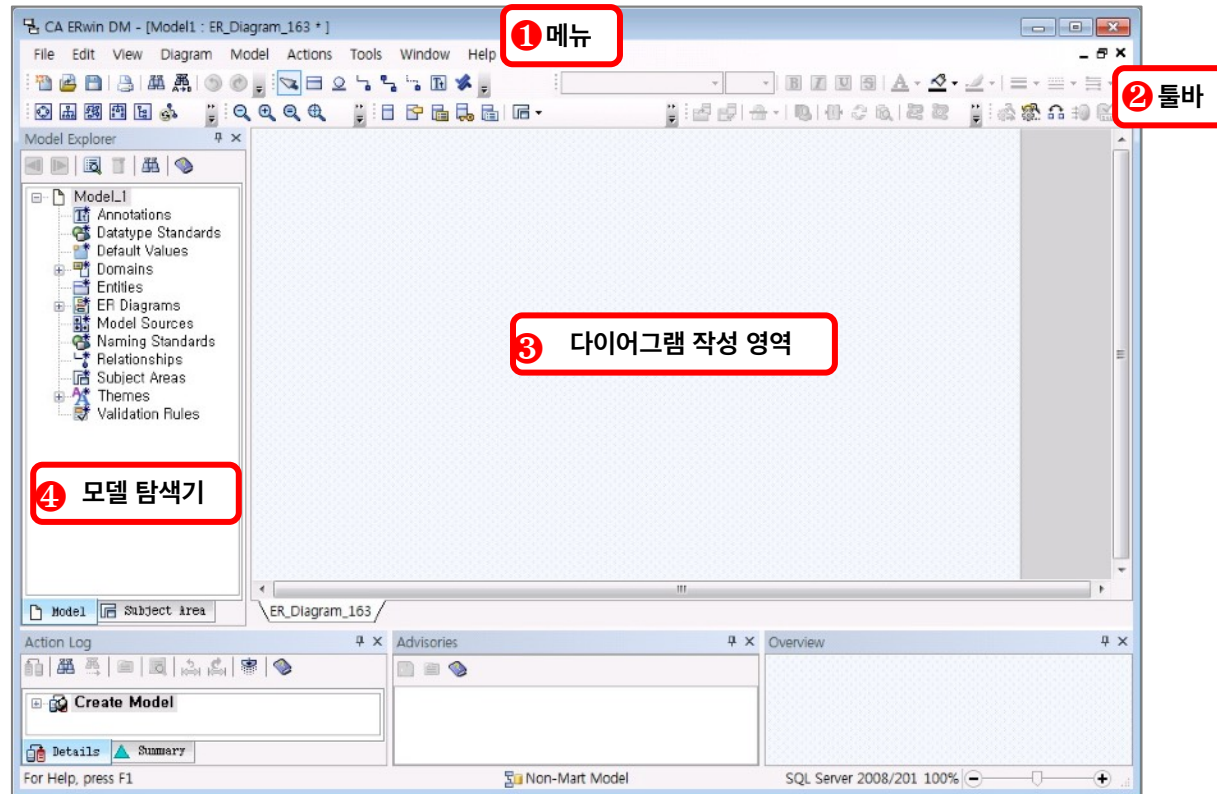


마당서점의 ER 다이어그램

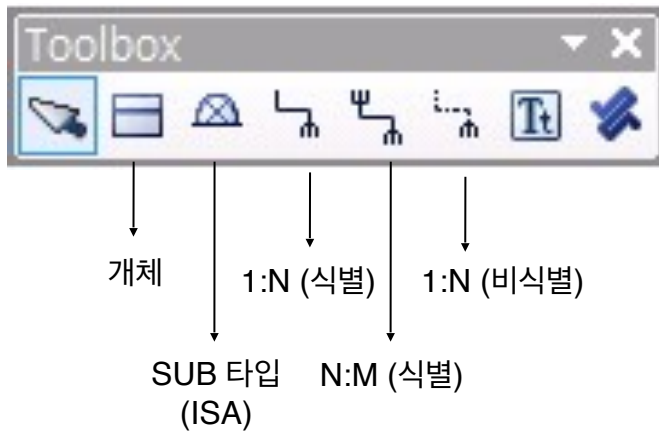
1 ERwin Data Modeler 실행



2 ERwin의 기본 화면



■ 툴바



- ① **개체** : 개체 타입의 이름, 식별자, 속성을 표현함.
- ② **SUB 타입** : ISA 모델의 슈퍼클래스와 서브클래스처럼 부모, 자식 관계에서 자식 개체가 서로 배타적인 관계를 가지는 여러 서브 개체 타입을 표현함.
- ③ **1:N(식별), N:M(식별), 1:N(비식별)** : 1, N, M은 두 개체 간의 관계에서 관계 대응수를 말함. 식별 관계는 두 개체가 부모(1), 자식(N) 관계일 때 부모의 기본키가 자식의 기본키가 되거나 기본키의 구성원으로 사용되는 관계로 실선으로 나타냄. 비식별 관계는 부모의 기본키가 자식의 기본키가 아닌 속성의 일부로 전이되는 관계로 점선으로 나타냄. 관계의 필수(1)와 선택(0)은 관계선의 옵션을 통해 선택할 수 있음.

- 목적과 대상에 맞는 모델, DBMS, 표기법을 선택해야 함.

Erwin 실습을 위한 기본 환경 설정

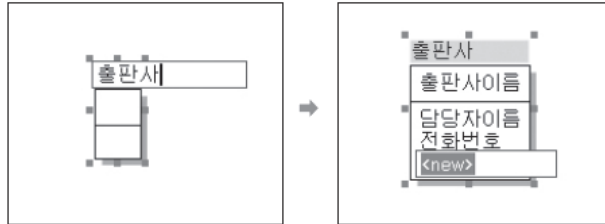
모델 타입	Logical/Physical	(새로만들때) File -> New -> Type
DBMS	Oracle	(새로만들때) File -> New -> Type
표기법	IE 표기법	Model -> Properties -> Notation 에서 변경

- 기본 환경 설정 순서(교재 341쪽~345쪽 참고)

- ① 모델 타입, DBMS 선택하기
- ② IE 표기법으로 변경하기
- ③ 툴바에 메뉴 추가하기(선택사항)

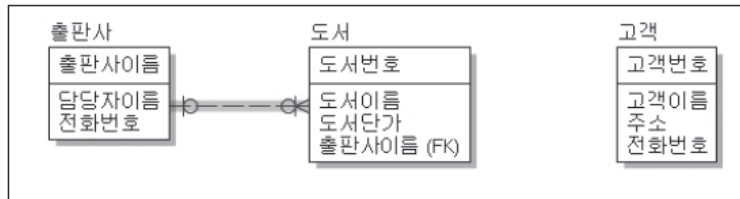
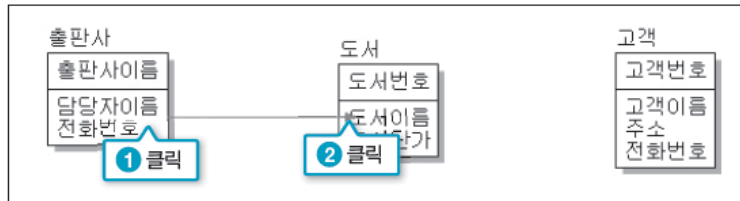
■ 1. 마당서점의 논리적 모델링

① 마당서점의 요구사항 분석 후 개체 만들기

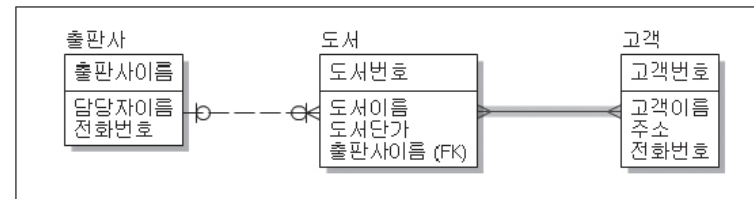
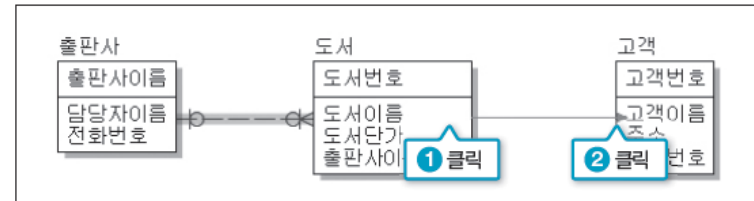


출판사 개체 생성

② 개체 간 관계 표현하기

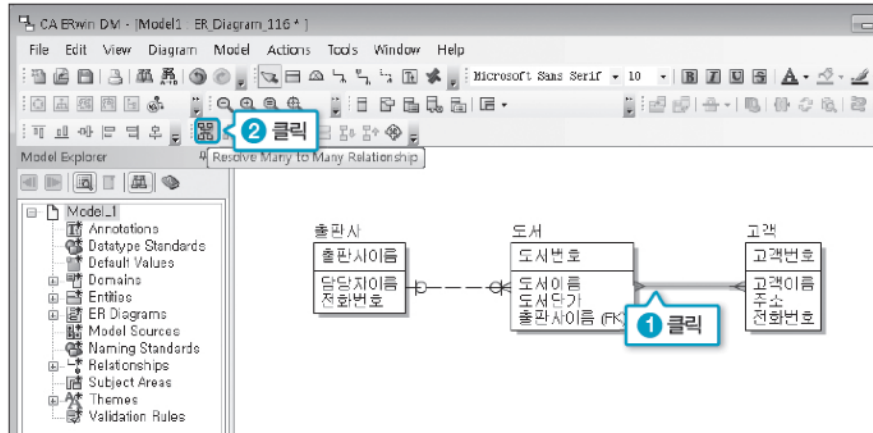


출판사, 도서 개체의 관계 설정(1:N 비식별)

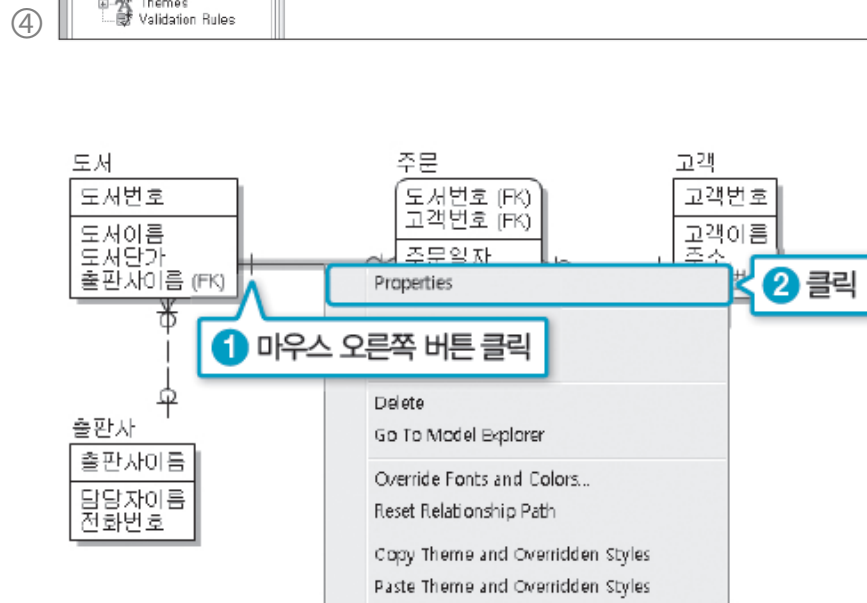


도서, 고객 개체의 관계 설정(N:M 관계)

③ N:M 관계 해소하기



N:M 관계 해소



개체 간 식별 관계 변경

Name	Parent	Child	Logical Only
BookPublisher	출판사	도서	<input type="checkbox"/>
OrdBook	도서	주문	<input type="checkbox"/>
OrdCustomer	고객	주문	<input type="checkbox"/>

General | Definition | Role Name | RI Actions | Style | UDP | Notes

'OrdCustomer' Type Properties

Type: Non-Identifying

Full Option: Nulls Not Allowed

'OrdCustomer' Relationship Properties

Parent-to-Child Phrase:

Child-To-Parent Phrase:

'OrdCustomer' Cardinality Properties

Cardinality: Zero, One or More

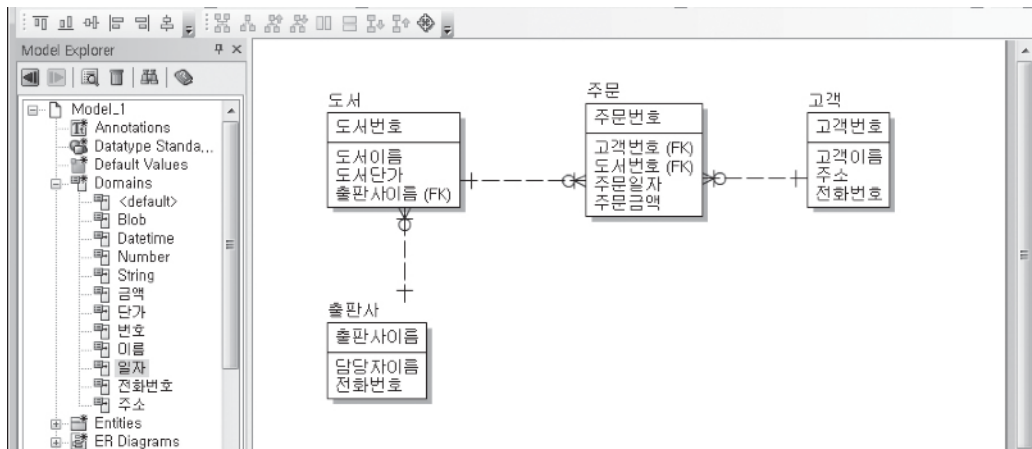
Cardinality Value:

Close Cancel

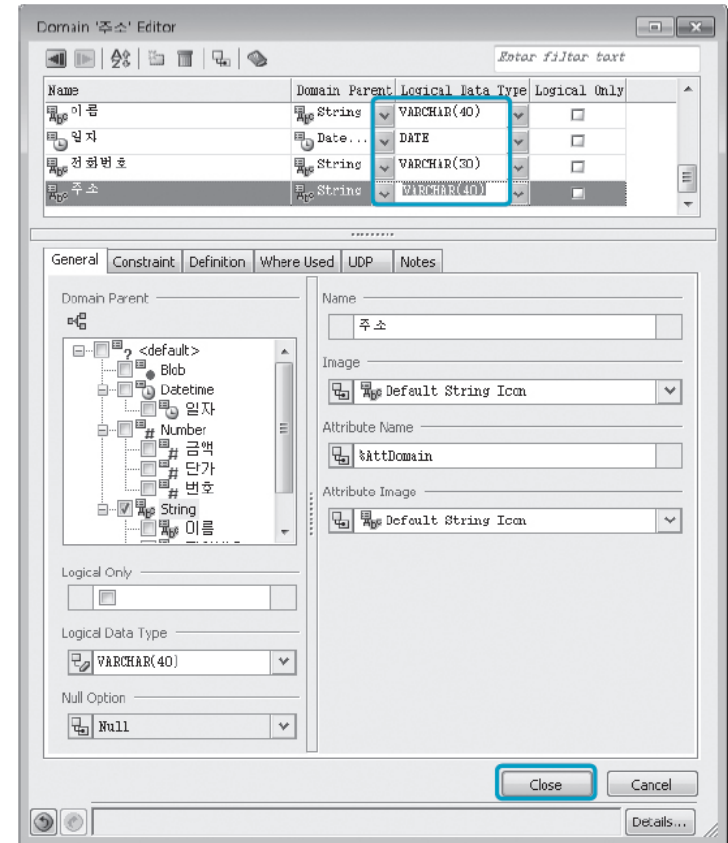
- 도메인이란 속성이 가질 수 있는 값을 정의하는 것. ER 다이어그램이 완성 후 도메인을 정의함.

구분(기본도메인)	도메인	데이터 타입
숫자(Number)	번호	NUMBER
	금액	NUMBER
	단가	NUMBER
문자(String)	이름	VARCHAR(40)
	주소	VARCHAR(40)
	전화번호	VARCHAR(30)
날짜·시간(Datetime)	일자	DATE

마당서점의 도메인별 데이터 타입 정의표

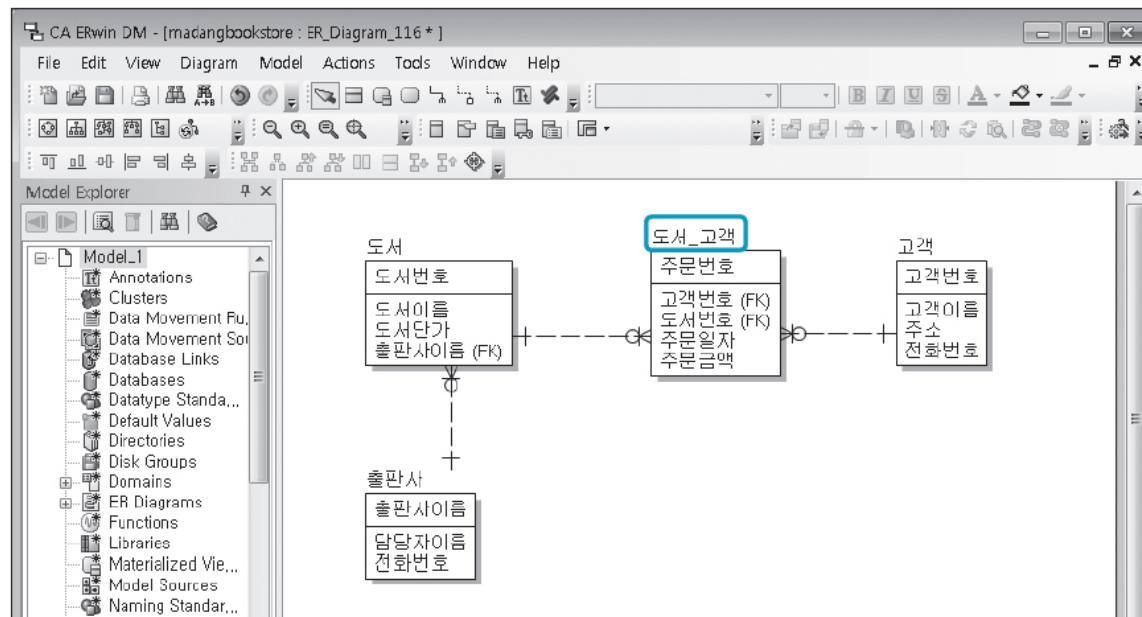


도메인 정의표에 따라 생성한 도메인



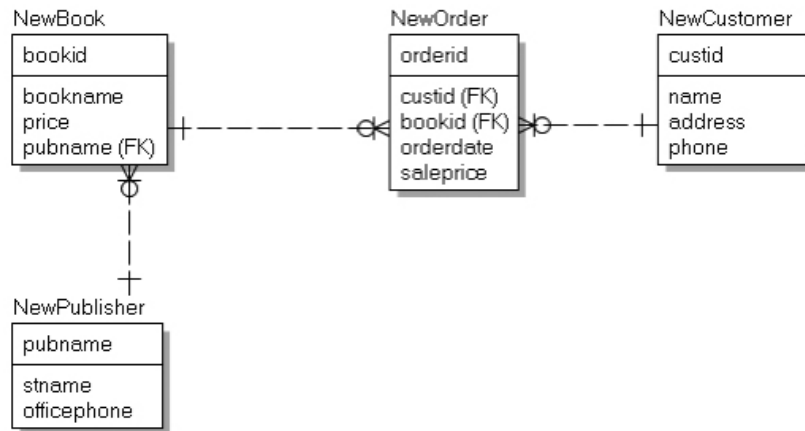
도메인별 데이터 타입 설정

- ① ER 다이어그램 불러오기 : [File] –[Open]
- ② Physical 타입으로 변경하기 : [View] – [Physical Model]

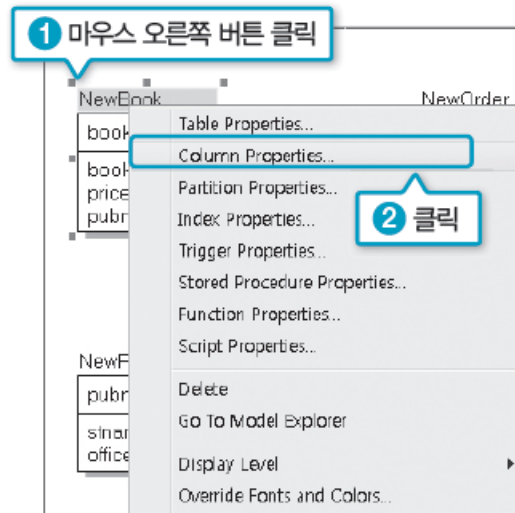


모델 타입의 변경

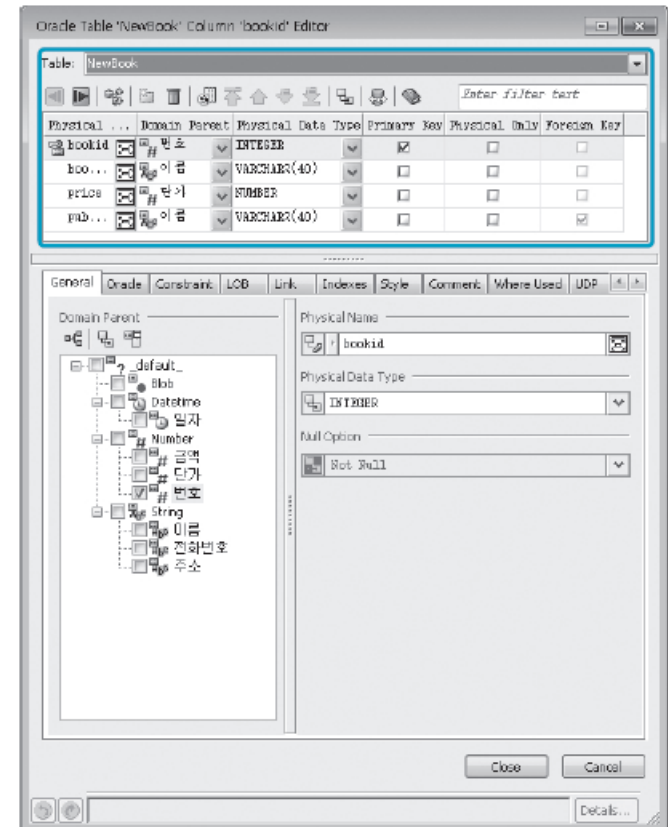
③ 물리적 모델링



④ 컬럼의 속성 확인하기



마당서점의 테이블



컬럼 속성 확인

<여기서 잠깐> Erwin 모델링 팁

(1) 최소 관계를 0->1로 변경하려면

1:N의 관계는 관계를 오른쪽 마우스로 클릭한 후

-> Properties -> Cardinality Properties

-> One or More로 수정(1쪽), Nulls Not Allowed(N쪽)

(2) 관계 이름을 표시하려면

모델 그림의 바탕화면에서 오른쪽 마우스 클릭한 후

-> Properties -> Relationship -> Display Logical Relationship Name(check)

-> Properties -> Relationship -> Physical Logical Relationship Name(check)

(3) 1:1 순환 관계의 표현

자기 자신으로 바탕화면을 한번 클릭해가면서 자기자신으로 클릭

1:N 카디널리티 표현을 위한 외래키는 별도로 생성해야 함

(관계를 오른쪽마우스선택 -> properties -> Role Name -> (이름 입력))

1 DBMS에 접속하기

[Actions]-[Database Connection] 메뉴 선택 후 [Oracle Connection] 창에서 다음과 같이 설정

Database : Oracle 11g
 Authentication : Database Authentication
 User Name : madnag
 Password : madang
 Connection String : orcl
 Connect As SYSDBA : 체크하지 않음

* 오라클 11g Express Edition 사용하는 경우
 Connection String : **xe**

<여기서 잠깐>

Erwin은 32비트 프로그램으로, 64비트에서는 다음과 같은 설정들을 해줘야 한다.

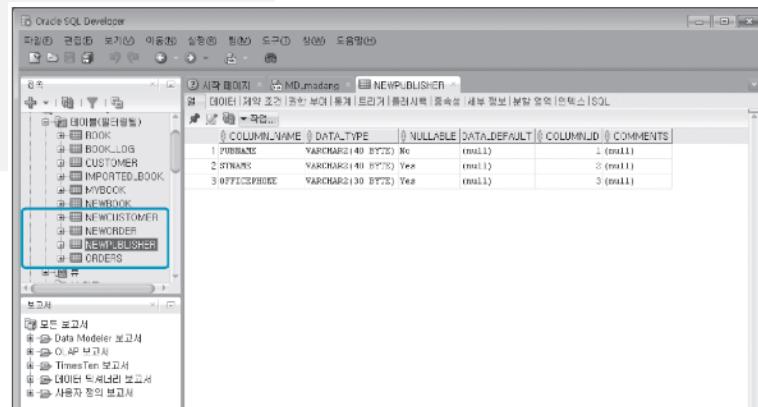
- (1) 오라클 32비트 클라이언트를 설치한다.(www.oracle.com->Downloads->Databa->Instant Client)
- (2) PATH 환경변수에 설치된 클라이언트 루트폴더를 경로 맨 앞에 추가해준다.
- (3) ORACLE_HOME 환경변수를 오라클이 설치된 폴더로 설정해준다.
 (예 : C:\oraclexe\app\oracle\product\11.2.0\server)
- (4) Erwin 다시 시작(혹은 부팅 후 다시 시작)

2 테이블 생성하기

[Actions]-[Forward Engineer]-[Schema] 메뉴 선택->

각 대상별로 오라클에 적용할 내용 설정함->

테이블과 인덱스를 제외한 모든 부분의 체크 해제-



madang 데이터베이스에 추가된 테이블