

# 7장 정규화(Normalization)

데이터베이스응용: 503.825(02)

안문석

## 목차

01 이상연상

02 함수 종속성

03 정규화

04 정규화 연습(부동산 데이터베이스)

## 학습목표

- 데이터베이스 설계 과정에서 발생할 수 있는 이상현상의 종류와 원인을 알아본다.
- 함수 종속성의 개념을 이해하고 관련 규칙을 알아본다.
- 함수 종속성을 이용한 정규화 과정을 알아본다.
- 데이터베이스 사례를 통한 정규화 과정을 연습한다.

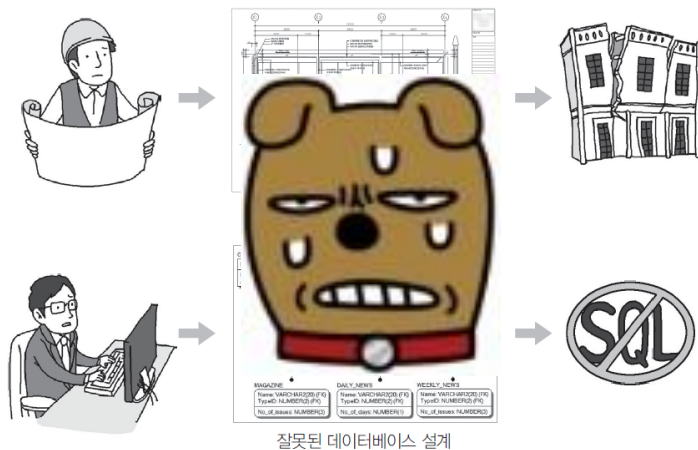
## Normalization and Anomalies

# 01. 이상현상(Anomaly)

- 이상현상의 개념
- 이상현상의 예



## 이상 현상의 개념



# 이상 현상의 개념

- **중복 데이터를 갖는 테이블**
  - 이상현상(Anomaly)을 야기
- **중복(Redundancy)**
  - The duplication of the data

# 이상현상의 개념

- **잘못 설계된 데이터베이스의 이상현상**
  - 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, 일관성 없음) 문제 발생

# 이상 연상의 예: 데이터 조작과 이상연상

## 학생수강 테이블

학생번호	학생이름	학과	주소	강좌이름	강의실
501	박지성	컴퓨터과	영국 맨체스터	데이터베이스	공학관 110
401	김연아	체육학과	대한민국 서울	데이터베이스	공학관 110
402	장미란	체육학과	대한민국 강원도	스포츠경영학	체육관 103
502	주신수	컴퓨터과	미국 클리블랜드	자료구조	공학관 111
403	박세리	체육학과	대한민국 대전	NULL	NULL

### DELETE

- 스포츠경영학 수강 취소 시
- 장미란 학생 삭제-수강정보 이외의 다른 정보인 학생정보도 같이 삭제됨
- 연쇄삭제 문제

### UPDATE

- 박지성 학생 주소 변경
- 불일치 문제

### INSERT

- 박세리 학생 삽입
- NULL 값 문제

## deletion anomaly

### Faculty and Their Courses

Faculty ID	Faculty Name	Faculty Hire Date	Course Code
389	Dr. Giddens	10-Feb-1985	ENG-206
407	Dr. Saperstein	19-Apr-1999	CMP-101
407	Dr. Saperstein	19-Apr-1999	CMP-201

DELETE

All information about **Dr. Giddens** is lost when he temporarily ceases to be assigned to any courses.

# insertion anomaly

## Faculty and Their Courses

Faculty ID	Faculty Name	Faculty Hire Date	Course Code
389	Dr. Giddens	10-Feb-1985	ENG-206
407	Dr. Saperstein	19-Apr-1999	CMP-101
407	Dr. Saperstein	19-Apr-1999	CMP-201

424	Dr. Newsome	29-Mar-2007
-----	-------------	-------------

?

Until the new faculty member, **Dr. Newsome**, is assigned to teach at least one course, his details cannot be recorded.

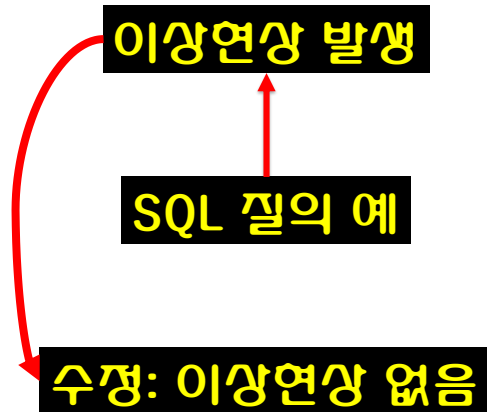
# update anomaly

## Employees' Skills

Employee ID	Employee Address	Skill
426	87 Sycamore Grove	Typing
426	87 Sycamore Grove	Shorthand
519	94 Chestnut Street	Public Speaking
519	96 Walnut Avenue	Carpentry

**Employee 519** is shown as having different addresses on different records.

## 이상연상의 예



## 잘못 설계된 계절학기 수강 테이블

- Summer 테이블 생성: 계절학기 수강정보 저장
- 제약조건: 계절학기는 안 학생이 안 과목만 신청

```

DROP TABLE IF EXISTS Summer;
/* 기존 테이블이 있으면 삭제 */
CREATE TABLE Summer(
    sid INTEGER,
    class VARCHAR(20),
    price INTEGER
);
  
```

sid	class	price

## 잘못 설계된 계절학기 수강 테이블

- Summer 테이블에 데이터 삽입: SQL 문

```
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;
```

Summer

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

## 잘못 설계된 계절학기 수강 테이블

- SQL 조회(retrieval) 질의문 실습
- 검색문은 이상연상 없음. 조작(I-U-D문)의 경우 발생

질의	SQL 문
계절학기를 듣는 학생의 학번과 수강하는 과목은?	SELECT sid, class FROM Summer;

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



## 잘못 설계된 계절학기 수강 테이블

질의	SQL 문
C 강좌의 수강료는?	<pre>SELECT price FROM Summer WHERE class='C' ;</pre>

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

2023-10-18

컴퓨터공학과

17

## 잘못 설계된 계절학기 수강 테이블

질의	SQL 문
수강료가 가장 비싼 과목은?	<pre>SELECT DISTINCT class FROM Summer WHERE price = (SELECT max(price) FROM Summer);</pre>

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

2023-10-18

컴퓨터공학과

18

## 잘못 설계된 계절학기 수강 테이블

질의	SQL 문
계절학기를 듣는 학생 수와 수강료 총액은?	SELECT COUNT(*), SUM(price) FROM Summer;

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

COUNT(*)	SUM(price)
4	65000

2023-10-18

컴퓨터공학과

19

## 잘못 설계된 계절학기 수강 테이블

### • 삭제이상

질의 7-1 200번 학생의 계절학기 수강신청을 취소하십시오.

```
/* 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;
```

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

```
/* 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);
```

2023-10-18

컴퓨터공학과

20

# 잘못 설계된 계절학기 수강 테이블

## • 삽입이상

질의 7-2 계절학기에 새로운 짜바 강좌를 개설하시오.

/\* 짜바 강좌 삽입 \*/ => 신청학생 무, NULL 삽입해야 함.  
INSERT INTO Summer VALUES (NULL, 'JAVA', 25000);

/\* Summer 테이블 조회 \*/  
SELECT \*  
FROM Summer;

학생 정보에 학번이 없는 학생은 없으므로 이상

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

SELECT COUNT(\*) "수강인원"  
FROM Summer;

SELECT COUNT(sid) "수강인원"  
FROM Summer;

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

수강인원  
5

수강인원  
4

수강인원  
4

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

## 2.1 잘못 설계된 계절학기 수강 테이블

## • 삽입이상

학생 정보에 학번이 없는 학생은 없으므로 이상

Summer

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

Student

sid	name
100	이상수
150	김민혁
200	박춘자
250	홍다솜

Null

JAVA

25000

# 잘못 설계된 계절학기 수강 테이블

## • 수정이상

질의 7-3 FORTRAN 강좌의 수강료를 20,000원에서 15,000원으로 수정하시오.

```
/* FORTRAN 강좌 수강료 수정 */
```

```
UPDATE Summer
```

```
SET price=15000
```

```
WHERE class='FORTRAN';
```

```
SELECT *
```

```
FROM Summer;
```

```
SELECT DISTINCT price "FORTRAN 수강료"
```

```
FROM Summer
```

```
WHERE class='FORTRAN';
```

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

FORTRAN 수강료
15000

# 잘못 설계된 계절학기 수강 테이블

## • 수정이상

질의 7-3 FORTRAN 강좌의 수강료를 20,000원에서 15,000원으로 수정하시오.

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

```
UPDATE Summer
```

```
SET price=20000
```

```
WHERE class='FORTRAN';
```

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

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

질의 7-3 FORTRAN 강좌의 수강료를 20,000원에서 15,000원으로 수정하시오.

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

```
UPDATE Summer
```

```
SET price=15000
```

```
WHERE class='FORTRAN' AND sid=100;
```

# 잘못 설계된 계절학기 수강 테이블

/\* Summer 테이블을 조회하면 **FORTRAN** 강좌의 수강료가 **안** 건만 수정되었음 \*/

```
SELECT *
FROM Summer;
```

/\* FORTRAN 수강료를 조회하면 **두 건**이 나옴(데이터 불일치 문제 발생) \*/

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

/\* 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;
```

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

FORTRAN 수강료
15000
20000

● Normalization  
 ✓ To help us to remove the Anomaly  
 ✓ To create good database

# 수정된 계절학기 수강 테이블

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(class, price)

class	price
FORTTRAN	20000
PASCAL	15000
C	10000

SummerEnroll(sid, class)

sid	class
100	FORTTRAN
150	PASCAL
200	C
250	FORTTRAN

2023-10-18

컴퓨터공학

27

# 수정된 계절학기 수강 테이블

## • SummerPrice 테이블, SummerEnroll 테이블 생성

```
/* 기존 테이블이 있으면 삭제하고 새로 생성하기 위한 준비 */
DROP TABLE IF EXISTS SummerPrice;
DROP TABLE IF EXISTS SummerEnroll;
```

```
/* SummerPrice 테이블 생성 */
CREATE TABLE SummerPrice
( class VARCHAR(20),
  price INTEGER
);
```

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

```
SELECT * FROM SummerPrice;
```

SummerPrice(class, price)

class	price
FORTTRAN	20000
PASCAL	15000
C	10000

2023-10-18

컴퓨터공학

28

## 수정된 계절학기 수강 테이블

- SummerPrice 테이블, SummerEnroll 테이블 생성

/\* SummerEnroll 테이블 생성 \*/

CREATE TABLE SummerEnroll

( sid INTEGER,  
class VARCHAR(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;

SummerEnroll(sid,class)

sid	class
100	FORTRAN
150	PASCAL
200	C
250	FORTRAN

## 수정된 계절학기 수강 테이블

- 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;

SummerEnroll(sid,class)

sid	class
100	FORTRAN
150	PASCAL
200	C
250	FORTRAN

## 수정된 계절학기 수강 테이블

### • 각 질의에 대한 SQL문 실습하기

SummerPrice 테이블과 SummerEnroll 테이블을 이용하여 처리하는 질의와 SQL 문

질의	SQL 문
C 강좌의 수강료는?	SELECT price FROM SummerPrice WHERE class= 'C' ;

SummerPrice(class, price)

class	price
FORTTRAN	20000
PASCAL	15000
C	10000

price
10000

## 수정된 계절학기 수강 테이블

### • 각 질의에 대한 SQL문 실습하기

SummerPrice 테이블과 SummerEnroll 테이블을 이용하여 처리하는 질의와 SQL 문

질의	SQL 문
수강료가 가장 비싼 과목은?	SELECT DISTINCT class FROM SummerPrice WHERE price = (SELECT max(price) FROM SummerPrice);

SummerPrice(class, price)

class	price
FORTTRAN	20000
PASCAL	15000
C	10000

class
FORTTRAN



## 수정된 계절학기 수강 테이블

### • 각 질의에 대한 SQL문 실습하기

SummerPrice 테이블과 SummerEnroll 테이블을 이용하여 처리하는 질의와 SQL 문

질의	SQL 문
계절학기를 듣는 학생 수와 수강료 총액은?	<pre>SELECT COUNT(*), SUM(price) FROM SummerPrice, SummerEnroll WHERE SummerPrice.class=SummerEnroll.class;</pre>

SummerPrice(class, price)

class	price
FORTTRAN	20000
PASCAL	15000
C	10000

SummerEnroll(sid, class)

sid	class
100	FORTTRAN
150	PASCAL
200	C
250	FORTTRAN

2023-10-18

컴퓨터

33

## 수정된 계절학기 수강 테이블

### • 각 질의에 대한 SQL문 실습하기

SummerPrice 테이블과 SummerEnroll 테이블을 이용하여 처리하는 질의와 SQL 문

질의	SQL 문
계절학기를 듣는 학생 수와 수강료 총액은?	<pre>SELECT COUNT(*), SUM(price) FROM SummerPrice, SummerEnroll WHERE SummerPrice.class=SummerEnroll.class;</pre>

COUNT(*)	SUM(price)
4	65000

2023-10-18

컴퓨터공학과

34

# 수정된 계절학기 수강 테이블

## • 삭제이상 없음

질의 7-4 200번 학생의 계절학기 수강신청을 취소하시오.

C 수강료
10000

SID	CLASS
100	FORTTRAN
150	PASCAL
250	FORTTRAN

C 수강료
10000

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

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

```
SELECT * FROM SummerEnroll;
```

```
/* C 강좌의 수강료가 존재하는지 확인 */ => 삭제이상 없음!!
SELECT price "C 수강료"
FROM SummerPrice
WHERE class='C ';
```

SummerPrice(class, price)

class	price
FORTTRAN	20000
PASCAL	15000
C	10000

SummerEnroll(sid, class)

sid	class
100	FORTTRAN
150	PASCAL
200	G
250	FORTTRAN

# 수정된 계절학기 수강 테이블

## • 삽입이상 없음

질의 7-5 계절학기에 새로운 자바 강의를 개설하시오.

```
/* 자바 강좌 삽입, NULL 값을 입력할 필요 없음 */
INSERT INTO SummerPrice VALUES ('JAVA', 25000);
```

SummerPrice(class, price)

class	price
FORTTRAN	20000
PASCAL	15000
C	10000
JAVA	25000

```
SELECT * FROM SummerPrice;
```

```
/* 수강신청 정보 확인 */
SELECT * FROM SummerEnroll;
```

SummerEnroll(sid, class)

sid	class
100	FORTTRAN
150	PASCAL
250	FORTTRAN

# 수정된 계절학기 수강 테이블

## • 수정이상 없음

질의 7-6 FORTRAN 강좌의 수강료를 20,000원에서 15,000원으로 수정하십시오.

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

SummerPrice(class, price)

class	price
FORTRAN	15000
PASCAL	15000
C	10000

SummerEnroll(sid, class)

sid	class
100	FORTRAN
150	PASCAL
250	FORTRAN

FORTRAN 수강료
15000

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

# 7장 정규화(Normalization)

데이터베이스응용  
(8W-1) 2022-10-21(금)  
한문석

## 02. 함수 종속성

- 함수 종속성의 개념
- 함수 종속성과 기본키
- 함수 종속성 다이어그램
- 이상연상과 결정자
- 함수 종속성 규칙
- 함수 종속성 예제

## Relation

이것이 릴레이션 맞는가?

EmployeeNumber	FirstName	LastName	Department	Email	Phone
100	Jerry	Johnson	Accounting	JJ@somewhere.com	834-1101
200	Mary	Abernathy	Finance	MA@somewhere.com	834-2101
300	Liz	Smathers	Finance	LS@somewhere.com	834-2102
400	Tom	Caruthers	Accounting	TC@somewhere.com	834-1102
500	Tom	Jackson	Production	TJ@somewhere.com	834-4101
600	Eleanore	Caldera	Legal	EC@somewhere.com	834-3101
700	Richard	Bandalone	Legal	RB@somewhere.com	834-3102

# Relation

이것이 릴레이션 맞는가?

EmployeeNumber	FirstName	LastName	Department	Email	Phone
100	Jerry	Johnson	Accounting	JJ@somewhere.com	834-1101
200	Mary	Abernathy	Finance	MA@somewhere.com	834-2101
300	Liz	Smathers	Finance	LS@somewhere.com	834-2102
400	Tom	Caruthers	Accounting	TC@somewhere.com	834-1102
500	Tom	Jackson	Production	TJ@somewhere.com	834-4101
600	Eleanore	Caldera	Legal	EC@somewhere.com	834-3101
700	Richard	Bandalone	Legal	RB@somewhere.com	834-3102

**속성은 단일 값을 가진다**

# Relation

이것이 릴레이션 맞는가?

EmployeeNumber	FirstName	LastName	Department	Email	Phone
100	Jerry	Johnson	Accounting	JJ@somewhere.com	834-1101
200	Mary	Abernathy	Finance	MA@somewhere.com	834-2101
300	Liz	Smathers	Finance	LS@somewhere.com	834-2102
400	Tom	Caruthers	Accounting	TC@somewhere.com	834-1102
500	Tom	Jackson	Production	TJ@somewhere.com	834-4101
600	Eleanore	Caldera	Legal	EC@somewhere.com	834-3101
700	Richard	Bandalone	Legal	RB@somewhere.com	834-3102

**속성은 서로 다른 이름을 가진다**

# Relation

이것이 릴레이션 맞는가?

EmployeeNumber	FirstName	LastName	Department	Email	Phone
100	Jerry	Johnson	Accounting	JJ@somewhere.com	834-1101
200	Mary	Abernathy	Finance	MA@somewhere.com	834-2101
300	Liz	Smathers	Finance	LS@somewhere.com	834-2102
400	Tom	Caruthers	Accounting	TC@somewhere.com	834-1102
500	Tom	Jackson	Production	TJ@somewhere.com	834-4101
600	Eleanore	Caldera	Legal	EC@somewhere.com	834-3101
700	Richard	Bandalone	Legal	RB@somewhere.com	834-3102

**한 속성의 값은 모두 같은 도메인 값을 가진다**

# Relation

이것이 릴레이션 맞는가?

EmployeeNumber	FirstName	LastName	Department	Email	Phone
100	Jerry	Johnson	Accounting	JJ@somewhere.com	834-1101
200	Mary	Abernathy	Finance	MA@somewhere.com	834-2101
300	Liz	Smathers	Finance	LS@somewhere.com	834-2102
400	Tom	Caruthers	Accounting	TC@somewhere.com	834-1102
500	Tom	Jackson	Production	TJ@somewhere.com	834-4101
600	Eleanore	Caldera	Legal	EC@somewhere.com	834-3101
700	Richard	Bandalone	Legal	RB@somewhere.com	834-3102

**속성의 순서는 무순서**

# Relation

이것이 릴레이션 맞는가?

EmployeeNumber	FirstName	LastName	Department	Email	Phone
100	Jerry	Johnson	Accounting	JJ@somewhere.com	834-1101
200	Mary	Abernathy	Finance	MA@somewhere.com	834-2101
300	Liz	Smathers	Finance	LS@somewhere.com	834-2102
400	Tom	Caruthers	Accounting	TC@somewhere.com	834-1102
500	Tom	Jackson	Production	TJ@somewhere.com	834-4101
600	Eleanore	Caldera	Legal	EC@somewhere.com	834-3101
700	Richard	Bandalone	Legal	RB@somewhere.com	834-3102

**투플의 순서는 무순서**

# Relation

이것이 릴레이션 맞는가?

EmployeeNumber	FirstName	LastName	Department	Email	Phone
100	Jerry	Johnson	Accounting	JJ@somewhere.com	834-1101
200	Mary	Abernathy	Finance	MA@somewhere.com	834-2101
300	Liz	Smathers	Finance	LS@somewhere.com	834-2102
400	Tom	Caruthers	Accounting	TC@somewhere.com	834-1102
500	Tom	Jackson	Production	TJ@somewhere.com	834-4101
600	Eleanore	Caldera	Legal	EC@somewhere.com	834-3101
700	Richard	Bandalone	Legal	RB@somewhere.com	834-3102

**릴레이션 내의 중복된 투플은 허용하지 않는다**

# Relation

이것이 릴레이션 맞는가?

EmployeeNumber	FirstName	LastName	Department	Email	Phone
100	Jerry	Johnson	Accounting	JJ@somewhere.com	834-1101
200	Mary	Abernathy	Finance	MA@somewhere.com	834-2101
300	Liz	Smathers	Finance	LS@somewhere.com	834-2102
400	Tom	Caruthers	Accounting	TC@somewhere.com	834-1102
500	Tom	Jackson	Production	TJ@somewhere.com	834-4101
600	Eleanore	Caldera	Legal	EC@somewhere.com	834-3101
700	Richard	Bandalone	Legal	RB@somewhere.com	834-3102

# 릴레이션이 아닌 테이블

왜? **한 셀당 여러 개의 데이터 존재**

EmployeeNumber	FirstName	LastName	Department	Email	Phone
100	Jerry	Johnson	Accounting	JJ@somewhere.com	834-1101
200	Mary	Abernathy	Finance	MA@somewhere.com	834-2101
300	Liz	Smathers	Finance	LS@somewhere.com	834-2102
400	Tom	Caruthers	Accounting	TC@somewhere.com	834-1102, 834-1191, 834-1192
500	Tom	Jackson	Production	TJ@somewhere.com	834-4101
600	Eleanore	Caldera	Legal	EC@somewhere.com	834-3101
700	Richard	Bandalone	Legal	RB@somewhere.com	834-3102, 834-3191



## 함수 종속성의 개념

- 쿠키상자를 사려고 한다.
- 한 쿠키상자의 값은 5,000원이다.
- 쿠키상자를 여러 개 구입했을 때, 가격은?  
– 쿠키 가격 = 쿠키상자 수  $\times$  5,000



2023-10-18

컴퓨터공학과

49

## 함수 종속성의 개념

- 쿠키 값과 쿠키상자의 수의 관계는?  
– 쿠키 값은 쿠키상자의 수에 종속된다.  
– **함수적으로 종속(functionally dependent)**된다.
- **쿠키상자의 수  $\rightarrow$  쿠키 값**  
– “쿠키상자의 수가 쿠키 값을 결정한다.” 라고 읽음  
– 이때 쿠키상자의 수를 **결정자(determinant)**라고 함.

2023-10-18

컴퓨터공학과

50

## For example

- 자동차와 그 차의 배기량을 추적하는 시스템을 설계한다고 가정하자!
- 각각의 자동차는 유일한 자동차 식별번호를 갖는다(차번호).
- **차번호 → 배기량**
  - 적절
  - 자동차 엔진이 하나 이상의 배기량을 갖는 것은 부적절(이 경우, 자동차는 유일하게 하나의 엔진만 갖는다고 가정했을 때)
- **배기량 → 차번호**
  - 이것은 부적절
  - 같은 엔진 배기량을 갖는 자동차가 많음

## 함수 종속성의 개념

### 함수 종속성

FD





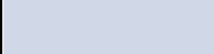
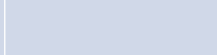
Functional Dependency

## 함수 종속성의 개념

- 릴레이션 R과 R에 속하는 속성의 집합 X, Y가 있을 때,
  - 릴레이션 R(X, Y)
  - X 각각의 값이 Y의 값 한 개와 대응이 될 때,
  - 'X는 Y를 함수적으로 결정한다' 라고 함

## 함수 종속성의 개념

$R(X, Y) \xrightarrow{\text{Instance}} r(X, Y)$

	X	Y
t1		
t2		
		

$t1.X = t2.X$ 이면  $t1.Y = t2.Y$ 라는 조건 만족 시  
r은 FD  $X \rightarrow Y$ 를 만족한다.

## 함수 종속성의 개념

- 릴레이션 R과 R에 속하는 속성의 집합 X, Y가 있을 때,
  - $X \rightarrow Y$ 로 표기
  - X를 결정자(determinant)
  - Y를 종속 속성(dependent attribute)

## 함수 종속성의 개념

함수 종속성은  
보통 릴레이션 설계 때  
속성의 의미로부터 정해짐

## 함수 종속성의 개념

- 어떤 속성 A의 값을 알면 다른 속성 B의 값이 유일하게 정해지는 의존 관계
- ‘속성 B는 속성 A에 종속한다(dependent)’
- ‘속성 A는 속성 B를 결정한다(determine)’
- ‘**A** → **B**’ 로 표기
- A를 B의 결정자라고 함

2023-10-18

컴퓨터공학과

57

## 함수 종속성의 개념

- 학생수강성적 릴레이션
  - 마당대학의 학생정보와 수강정보 저장
- 각 속성 사이에는 의존성 존재

### 학생수강성적

학생번호	학생이름	주소	학과	학과사무실	강좌이름	강의실	성적
501	박지성	영국 맨체스터	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	주신수	미국 클리블랜드	컴퓨터과	공학관101	자료구조	공학관 111	4.0
501	박지성	영국 맨체스터	컴퓨터과	공학관101	자료구조	공학관 111	3.5

2023-10-18

컴퓨터공학과

58

## 2. 함수 종속성의 개념

### ■ 학생수강성적 릴레이션에서 종속관계에 있는 예

- 학생번호 → 학생이름
- 학생번호 → 주소
- 강좌이름 → 강의실
- 학과 → 학과사무실

학생번호	학생이름	주소	학과	학과사무실	강좌이름	강의실	성적
501	박지성	영국 맨체스터	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	주인수	미국 클리블랜드	컴퓨터과	공학관101	자료구조	공학관 111	4.0
501	박지성	영국 맨체스터	컴퓨터과	공학관101	자료구조	공학관 111	3.5

## 2. 함수 종속성의 개념

### ■ 종속하지 않는 예

- 학생이름 → 강좌이름
- 학과 → 학생번호

학생번호	학생이름	주소	학과	학과사무실	강좌이름	강의실	성적
501	박지성	영국 맨체스터	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	주인수	미국 클리블랜드	컴퓨터과	공학관101	자료구조	공학관 111	4.0
501	박지성	영국 맨체스터	컴퓨터과	공학관101	자료구조	공학관 111	3.5

## 2. 함수 종속성의 개념

### ■ 종속아는 것처럼 보이지만 그렇지 않은 예

#### ■ 학생이름 → 학과

- 동명이인이 존재했을 때

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

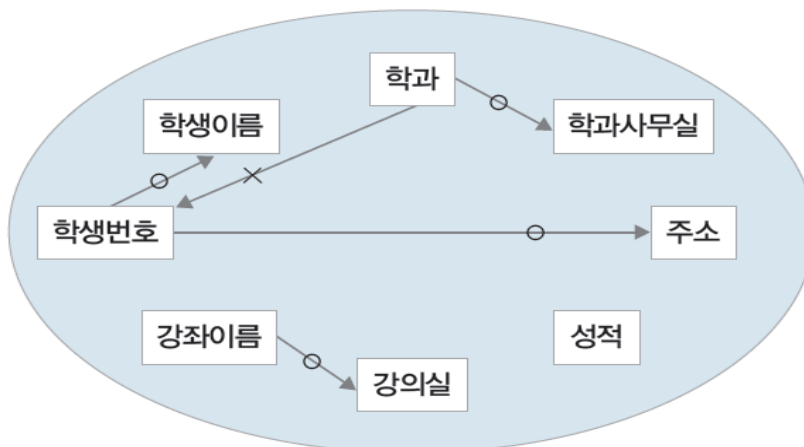
2023-10-18

컴퓨터공학과

61

## 2. 함수 종속성의 개념

### 학생수강신청 릴레이전의 종속관계



2023-10-18

컴퓨터공학과

62

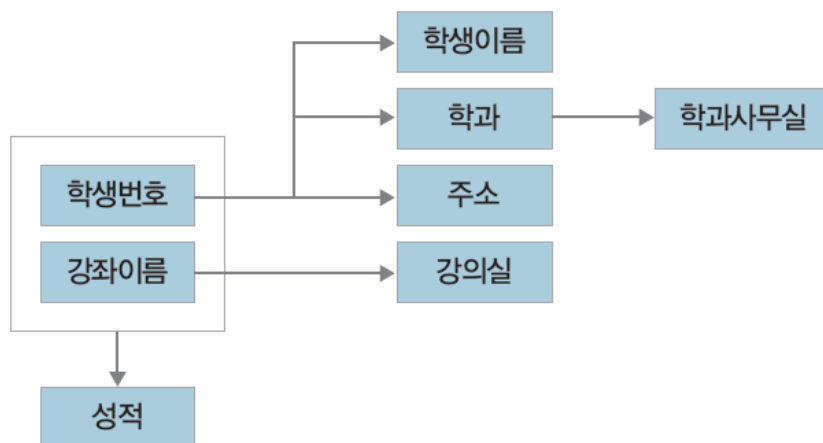
## 2. 함수 종속성 다이어그램

### ■ Functional Dependency Diagram::FDD

#### ■ 함수 종속성을 나타내는 표기법

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

## 2. 함수 종속성 다이어그램





### 3. 함수 종속성 규칙

#### 함수 종속성 규칙

(functional dependency rule)

$R(X, Y, Z)$

$X, Y, Z :: R$ 에 포함된 속성의 집합

함수 종속성에 관한 다음과 같은 규칙  
이 성립

2023-10-18

컴퓨터공학과

65

### 3. 함수 종속성 규칙

#### Trivial FD(당연 함수종속)

종속자가 결정자의 속성집합에 포함하면 당연 함수종속이라 함

Symbolically:  $A \twoheadrightarrow B$  is **trivial functional dependency** if  $B$  is a subset of  $A$ .

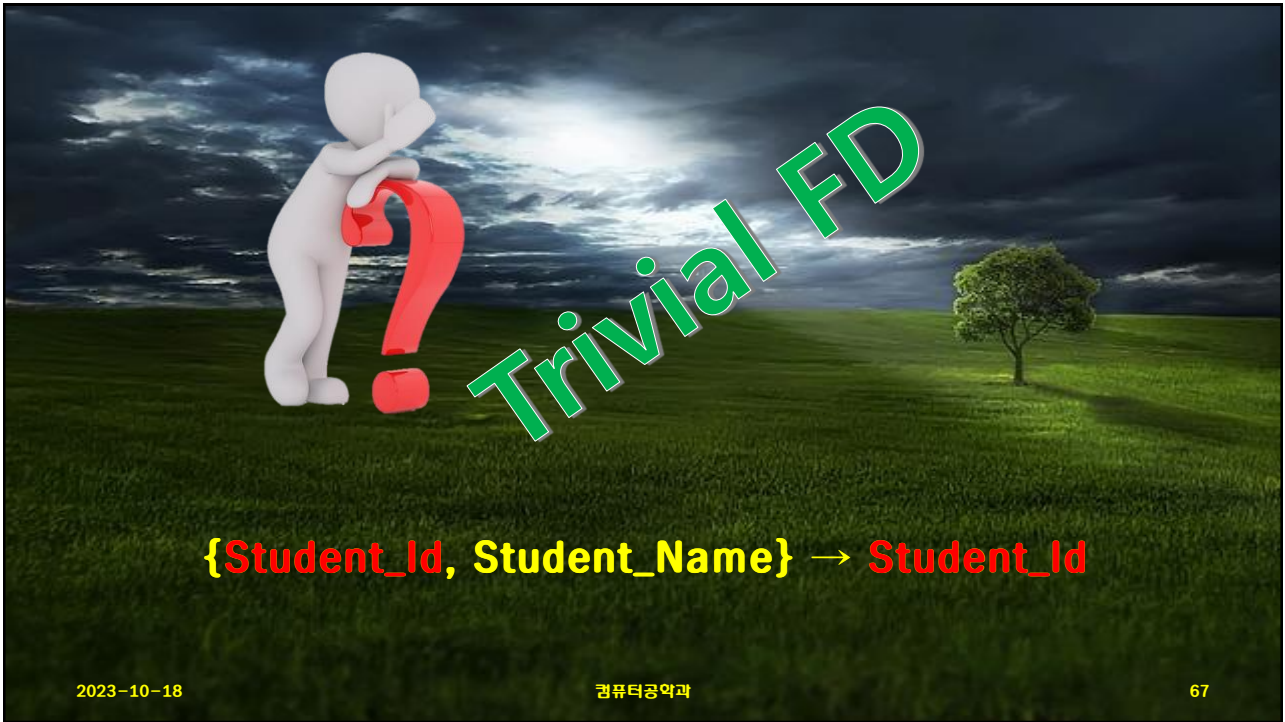
$X \rightarrow X :: \text{학생이름} \rightarrow \text{학생이름}$

$(X, Y) \rightarrow Y :: (\text{학번}, \text{학생이름}) \rightarrow \text{학번}$

2023-10-18

컴퓨터공학과

66



### 3. 함수 종속성 규칙(Armstrong's Axioms)

**부분집합(Subset::Reflexive rule) 규칙:**

if  $Y \subseteq X$ , then  $X \rightarrow Y$ : **Trivial FD**

**증가(Augmentation) 규칙:**

If  $X \rightarrow Y$ , then  $XZ \rightarrow YZ$

**이행(Transitivity) 규칙:**

If  $X \rightarrow Y$  and  $Y \rightarrow Z$ , then  $X \rightarrow Z$

### 3. 함수 종속성 규칙

위 세 가지 규칙으로부터 추가적 규칙

**결합(Union) 규칙:**

If  $X \rightarrow Y$  and  $X \rightarrow Z$ , then  $X \rightarrow YZ$

**합성(Composition) 규칙:**

If  $X \rightarrow Y$  and  $A \rightarrow B$ , then  $XA \rightarrow YB$

**분해(Decomposition) 규칙 :**

If  $X \rightarrow YZ$ , then  $X \rightarrow Y$  and  $X \rightarrow Z$

**유사이행(Pseudo transitivity) 규칙:**

If  $X \rightarrow Y$  and  $WY \rightarrow Z$ , then  $WX \rightarrow Z$

### 합집합(Union: Additive) 규칙 증명

■  $X \rightarrow Y$  and  $X \rightarrow Z$ , then  $X \rightarrow YZ$

1.  $X \rightarrow Y$ (주어짐)

2.  $X \rightarrow Z$ (주어짐)

3.  $X \rightarrow XY$ (1에 대해 증가규칙 적용,  $X$ 를 양쪽 편에 부가,  $XX = X$ )

4.  $XY \rightarrow YZ$ (2에 대해 증가규칙 적용,  $Y$ 를 양쪽에 증가)

5.  $X \rightarrow YZ$ (3, 4에 대해 이행규칙 적용)

## 분해(Decomposition) 규칙 증명

■ If  $X \rightarrow YZ$ , then  $X \rightarrow Y$  and  $X \rightarrow Z$

1.  $X \rightarrow YZ$  (주어짐)
2.  $YZ \rightarrow Z$  ( $Y \subseteq YZ$ , 부분집합 규칙)
3.  $X \rightarrow Z$  (1, 2에 대해 이행규칙 적용)

## 합성(Composition) 규칙 증명

■  $X \rightarrow Y$  and  $A \rightarrow B$ , then  $XA \rightarrow YB$

1.  $X \rightarrow Y$  (Given)
2.  $A \rightarrow B$  (Given)
3.  $XA \rightarrow YA$  (Augmentation of 1 and A)
4.  $XA \rightarrow Y$  (Decomposition of 3)
5.  $XA \rightarrow XB$  (Augmentation of 2 and X)
6.  $XA \rightarrow B$  (Decomposition of 5)
7.  $XA \rightarrow YB$  (Union 4 and 6)

## 유사이행(Pseudotransitivity) 규칙 증명

- If  $X \rightarrow Y$  and  $WY \rightarrow Z$ , then  $WX \rightarrow Z$
1.  $X \rightarrow Y$ (주어짐)
  2.  $WY \rightarrow Z$ (주어짐)
  3.  $WX \rightarrow WY$ (1에 대해 증가규칙 적용, W를 양쪽 편에 증가)
  4.  $WX \rightarrow Z$ (3, 2에 대해 이행규칙 적용)

2023-10-18

컴퓨터공학과

73

## 3. 함수 종속성 규칙

학생수강성적 릴레이션에 함수 종속성 규칙을 적용한 예

적용 규칙	사례	설명
<b>부분집합 규칙</b> if $Y \subseteq X$ , then $X \rightarrow Y$	$(\text{학과}, \text{주소}) \rightarrow \text{학과}$	학과는 (학과, 주소)의 부분집합 속성이므로, '(학과, 주소) $\rightarrow$ 학과' 성립
<b>증가 규칙</b> If $X \rightarrow Y$ , then $XZ \rightarrow YZ$	$(\text{학생번호}, \text{강좌이름}) \rightarrow (\text{학생이름}, \text{강좌이름})$	'학생번호 $\rightarrow$ 학생이름' 이므로 강좌 이름을 추가하여, '(학생번호, 강좌이름) $\rightarrow$ (학생이름, 강좌이름)' 성립
<b>이행 규칙</b> If $X \rightarrow Y$ and $Y \rightarrow Z$ , then $X \rightarrow Z$	$\text{학생번호} \rightarrow \text{학과사무실}$	'학생번호 $\rightarrow$ 학과', '학과 $\rightarrow$ 학과사무실' 이므로 이행 규칙을 적용하여, '학생번호 $\rightarrow$ 학과사무실' 성립

2023-10-18

컴퓨터공학과

74

### 3. 함수 종속성 규칙

#### 학생수강생적 릴레이션에 함수 종속성 규칙을 적용한 예

적용 규칙	사례	설명
<b>결합 규칙</b> If $X \rightarrow Y$ and $X \rightarrow Z$ , then $X \rightarrow YZ$	학생번호 $\rightarrow$ (학생이름, 주소)	'학생번호 $\rightarrow$ 학생이름', '학생번호 $\rightarrow$ 주소' 이므로 결합 규칙을 적용하여, '학생번호 $\rightarrow$ (학생이름, 주소)' 성립
<b>분해 규칙</b> If $X \rightarrow YZ$ , then $X \rightarrow Y$ and $X \rightarrow Z$	학생번호 $\rightarrow$ 학생이름, 학생번호 $\rightarrow$ 주소	'학생번호 $\rightarrow$ (학생이름, 주소)' 이므로 분해하여, '학생번호 $\rightarrow$ 학생이름', '학생번호 $\rightarrow$ 주소' 성립
<b>유사이행 규칙</b> If $X \rightarrow Y$ and $WY \rightarrow Z$ , then $WX \rightarrow Z$	(강좌이름, 학생이름) $\rightarrow$ 성적	'학생이름 $\rightarrow$ 학생번호' (학생이름이 같은 경우가 없다고 가정한다), '(강좌이름, 학생번호) $\rightarrow$ 성적' 이므로 유사이행 규칙을 적용하여, '(강좌이름, 학생이름) $\rightarrow$ 성적' 성립

2023-10-18

컴퓨터공학과

75

### 4. 함수 종속성과 기본키

- 릴레이션의 함수 종속성 파악 수단
  - 우선 기본키를 찾아야 함
- 함수 종속성에서 기본키의 역할
  - 알면 정규화 과정을 쉽게 이해

#### 함수 종속성과 기본키

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

즉 기본키는 릴레이션의 모든 속성에 대해

**결정자(determinant)**임.

2023-10-18

컴퓨터공학과

76

## 4. 함수 종속성과 기본키

학생 릴레이션

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

- 예) 이름이 같은 학생이 없다고 가정하면,
- ‘이름 → 학과, 이름 → 주소, 이름 → 취득학점’ 이므로 ‘이름 → 이름, 학과, 주소, 취득학점’ 이 성립
- 즉 이름 속성이 학생 릴레이션의 전체를 결정함

2023-10-18

컴퓨터공학과

77

## 5. 이상연상과 결장자

### ■ 이상연상

- **한 개의 릴레이션에 두 개 이상의 정보가 포함되어 있을 때**

### ■ 학생수강성적 릴레이션의 경우

- **학생 정보(학생번호, 학생이름, 주소, 학과)**
- **강좌 정보(강좌이름, 강의실)**

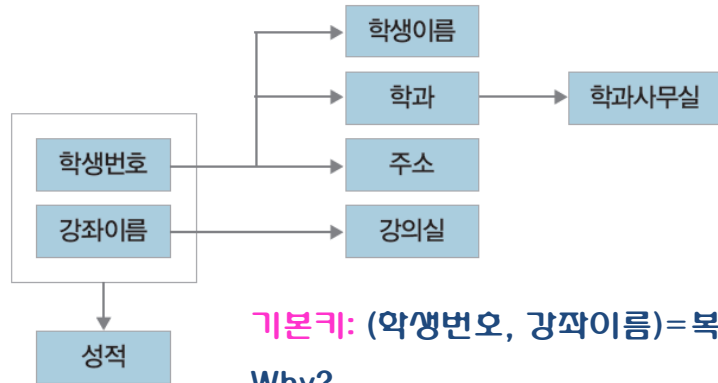
2023-10-18

컴퓨터공학과

78

## 5. 이상연상과 결정자

### 학생수강성적 릴레이전의 함수 종속성 다이어그램



## 5. 이상연상과 결정자

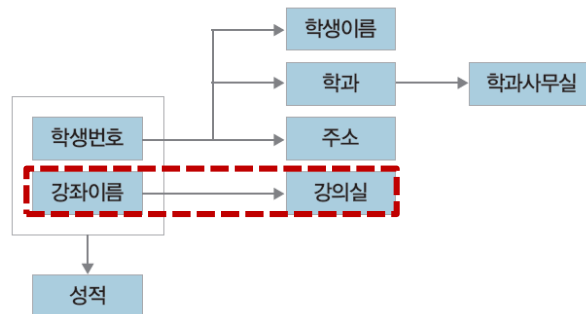
- 학생수강성적 릴레이전의 **부분 릴레이전 분해**
- 분해할 때
  - 부분 릴레이전의 결정자는 원래 릴레이전에 남김
  - 분해된 부분 릴레이전이 원래 릴레이전과 관계를 형성할 수 있음



## 5. 이상연상과 결장자

### ■ 1단계 : 학생수강성적 릴레이연에서 (강좌이름, 강의실)을 분리

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



2023-10-18

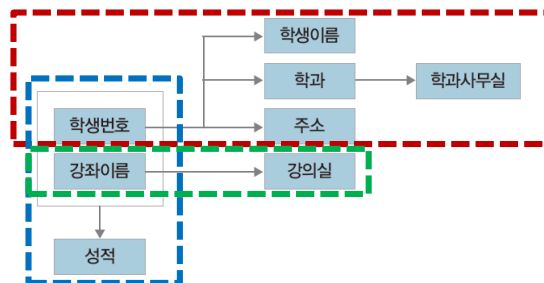
컴퓨터공학과

81

## 5. 이상연상과 결장자

### ■ 2단계 : 학생수강성적1 릴레이연에서 (학생번호, 강좌이름, 성적)을 분리

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



2023-10-18

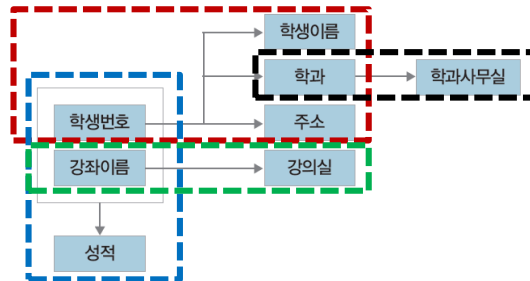
컴퓨터공학과

82

## 5. 이상연상과 결정자

### ■ 3단계 : 학생학과 릴레이션에서 (학과, 학과사무실) 분리

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



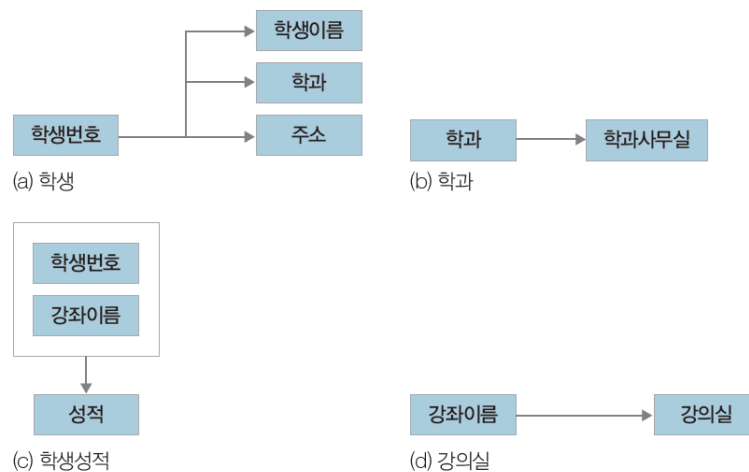
2023-10-18

컴퓨터공학과

83

## 5. 이상연상과 결정자

### 학생수강성적 릴레이션을 분해한 결과



2023-10-18

컴퓨터공학과

84

## 6. 함수 종속성 예제

- 함수 종속성은 보통 릴레이션을 설계할 때 **속성의 의미**로부터 **쟁에짐**
- 역으로 릴레이션에 저장된 **속성 값**으로부터 추정할 수 있음

예제 7-1 다음 릴레이션 R에서 아래 함수 종속성이 성립하는지 살펴보세요.

R

A	B	C
2	3	8
5	9	6
7	9	6
5	2	2

[함수 종속성]

❌ ①  $A \rightarrow B$

②  $B \rightarrow C$

❌ ③  $(B, C) \rightarrow A$

④  $(A, B) \rightarrow C$

2023-10-18

컴퓨터공학과

85

## 6. 함수 종속성 예제

예제 7-2 다음 릴레이션 R에서 성립하는 함수 종속성을 모두 찾아보세요.

- 결정자가 한 개인 경우 :

$B \rightarrow C, C \rightarrow B, D \rightarrow A, D \rightarrow B, D \rightarrow C$

- 결정자가 두 개인 경우 :

$AB \rightarrow C$  ( $B \rightarrow C$  이므로  $AB \rightarrow C$  는 함수종속성 규칙에서 당연히 성립)

$AB \rightarrow D$

$AC \rightarrow B$  (함수종속성 규칙에서 당연히 성립)

$AC \rightarrow D$

$AD \rightarrow B$  (함수종속성 규칙에서 당연히 성립)

- 결정자가 세 개인 경우 :

$ABC \rightarrow D$  (함수종속성 규칙에서 당연히 성립) ... 등

정답은 **당연히 성립하는 것들을 제외한** 다음 규칙만 적어주면 됨

$B \rightarrow C, C \rightarrow B, D \rightarrow A, D \rightarrow B, D \rightarrow C, AB \rightarrow D, AC \rightarrow D$

R

A	B	C	D
a1	b4	c1	d6
a1	b2	c4	d5
a2	b4	c1	d4
a2	b2	c4	d3
a2	b3	c2	d2

2023-10-18

컴퓨터공학과

86

## 03. 정규화

- 정규화 과정
- 무손실 분해
- 정규화 정리

## 03 정규화(Normalization)

- 데이터를 구조화하는 프로세스: **RDBMS 설계에서 중복을 최소화**
- 이상현상이 발생하는 릴레이션을 분해
- 이상현상을 없애는 과정
- 이상현상이 있는 릴레이션
  - 이상현상을 일으키는 함수 종속성의 유형에 따라 등급을 구분 가능
- 릴레이션은 정규형 개념으로 구분
  - 정규형이 높을수록 이상현상은 줄어듦

## 03 정규화

### 이동수단과 릴레이션의 등급 구분



오토바이  
1등급



자동차  
2등급



기차  
3등급



비행기  
4등급

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

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

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

## 1.1 제 1정규형

- 릴레이션 R의 모든 속성 값이 **원자값**을 가짐
- 원자값::Atomic Value
- 제 1정규형으로 변환
  - 고객취미들(이름, 취미들) 릴레이션
  - ⇒ 고객취미(이름, 취미) 릴레이션으로 바꾸어 저장
  - 제 1정규형을 만족

## 1.1 제 1정규형

- A relation in which the intersection of each row and column contains **one and only one value**

고객취미들(이름, 취미들)

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



고객취미(이름, 취미)

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

## 1.2 제 2정규형

- 릴레이션의 기본키가 복합키일 때
  - 복합키의 일부분이 다른 속성의 결정자인지 여부 판단
- 릴레이션 R이 **제 1정규형**이고 기본키가 아닌 속성이 기본키에 **완전 함수 종속**인 것
- A relation is in second normal form **if it is in 1NF and every non-primary key attribute is fully functionally dependent on the primary key.**

## 1.2 제 2정규형

### • 완전 함수 종속

– Full Functional Dependency

– FFD 조건

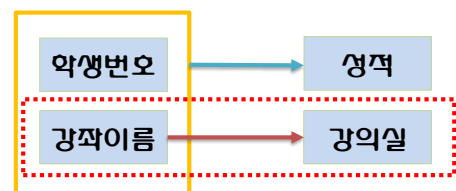
- A와 B가 릴레이션 R의 속성이고
- $A \rightarrow B$  종속성이 성립할 때,
- B가 A의 속성 전체에 함수 종속하고
- 부분 집합 속성에 함수 종속하지 않음

## 1.2 제 2정규형

### 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



부분함수종속

Partially Functional Dependency

## 1.2 제 2정규형

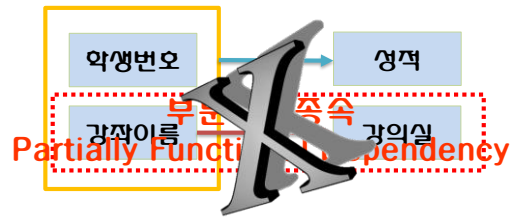
### • 제 2정규형으로 변환

– 수강강좌 릴레이션에서 이상연상 발생

• (강좌이름, 강의실)을 분해

– 강좌이름 → 강의실: PFD

– 즉, **부분함수종속성 제거**



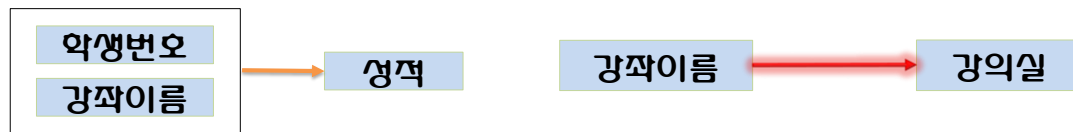
## 1.2 제 2정규형

### 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



## 1.3 제 3정규형

- 이행적 종속 여부 판단
- 3 정규형 조건
  - 릴레이션 R이 제 2정규형
  - 직접 종속:
    - 기본키가 아닌 속성이 기본키에 비이행적(non-transitive)으로 종속
- A relation that is in first and second normal form and in which no non-primary key is transitively dependent on the primary key.

## 1.3 제 3정규형

### ■ 이행적 종속이란?

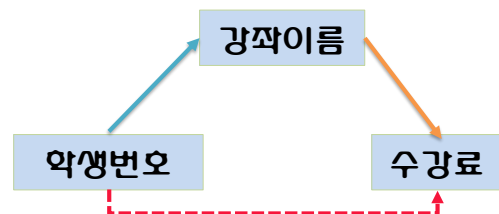
- $A \rightarrow B, B \rightarrow C$ 가 성립할 때,
- $A \rightarrow C$ 가 성립되는 함수 종속성

#### 계절학기

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

#### 계절학기 릴레이션

\* 학생은 한 강좌만 신청할 수 있다고 가정



## 1.3 제 3정규형

### • 제 3정규형으로 변환

- 계열학기 릴레이션에서 이상연상을 일으키는 (강좌이름, 수강료)를 분해함

계열수강

학생번호	강좌이름
501	데이터베이스
401	데이터베이스
402	스포츠경영학
502	자료구조

수강료

강좌이름	수강료
데이터베이스	20000
스포츠경영학	15000
자료구조	25000

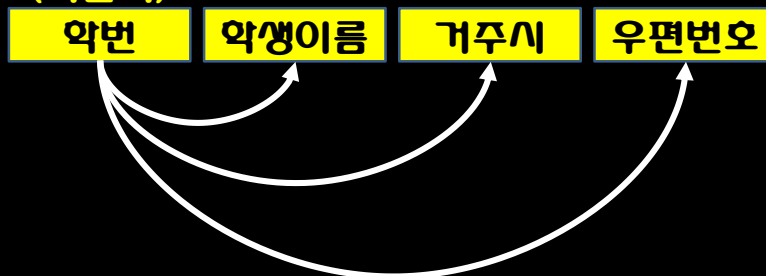
학생번호 → 강좌이름

강좌이름 → 수강료

계열학기 릴레이션을 계열수강, 수강료 릴레이션으로 분해

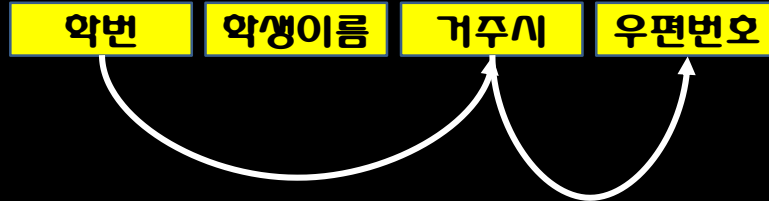
## 1.3 제 3정규형

(기본키)



## 1.3 제 3정규형

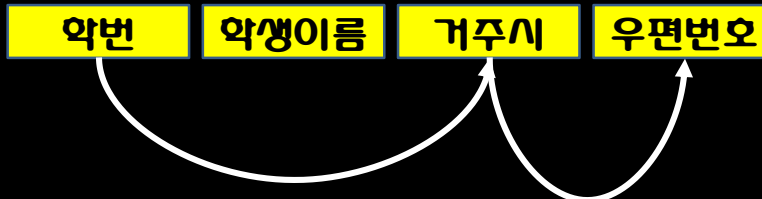
(기본키)



transitive dependency.

## 1.3 제 3정규형

(기본키)



transitive dependency.

학번    학생이름    우편번호

우편번호    거주시

## 1.4 BCNF

- 릴레이션에 존재하는 함수종속성에서
  - 모든 결정자가 후보키인 정규형
  - 식별자 속성이 일반 속성에 종속되지 않아야 함
- Boyce+Codd NF
- An extension of Third Normal Form on strict terms.
- 모든 BCNF 릴레이션은 제3정규형이지만
  - 제 3 정규형이 꼭 BCNF는 아님  $\Rightarrow$  3.5NF

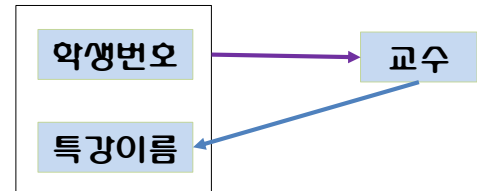
## 1.4 BCNF

- BCNF 정규형 정의
  - 릴레이션 R에서 함수 종속성  $X \rightarrow Y$  일 때
    - 모든 결정자 X가 후보키
      - $X \rightarrow Y$  is a trivial functional dependency ( $Y \subseteq X$ ),
      - X is a superkey for schema R.
- if and only if every determinant is a candidate key.
  - A relation is in BCNF

## 1.4 BCNF

### 특강수강

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



### 특강수강 릴레이션

- \*한 학생은 한 개 이상의 특강을 신청
- \*교수는 1개의 특강만을 담당

## 1.4 BCNF

### • BCNF 정규형으로 변환

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

### 특강신청

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



특강수강 릴레이션을 특강신청,  
특강교수 릴레이션으로 분해

# 1.4 BCNF

## • BCNF 정규형으로 변환

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

특강교수

특강이름	교수
소셜네트워크	김교수
인간과 동물	송교수
창업전략	박교수
창업전략	홍교수



특강수강 릴레이션을 특강신청, 특강교수 릴레이션으로 분해

## 2. 무손실 분해

- 릴레이션 R을 릴레이션 R1과 R2로 분해할 때,

- 무손실(lossless-join) 분해

- $R1 \bowtie R2 = R$

- 무손실 분해 조건

- 공통된 속성이 R1이나 R2의 키이어야 함

- $R1 \cap R2 \rightarrow R1$

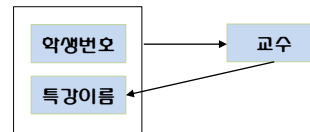
- $R1 \cap R2 \rightarrow R2$

- 위 조건 중 하나를 만족해야 함

## 2. 무손실 분해

특강수강: BCNF로 분해

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



## 2. 무손실 분해

특강수강 릴레이선의 분해

•  $R1 \cap R2 \rightarrow R2$

구분	릴레이션 분해	무손실 분해 여부
[분해1]	특강수강(학생번호, 특강이름, 교수) $\rightarrow$ R1(학생번호, 교수), R2(교수, 특강이름)	R1과 R2의 공통 속성은 교수이며, 교수는 R2의 키 $\rightarrow$ 무손실 분해 규칙을 만족

R1::특강신청

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

R2::특강교수

특강이름	교수
소셜네트워크	김교수
인간과 동물	송교수
창업전략	박교수
창업전략	홍교수

## 2. 무손실 분해

### 특강수강 릴레이션의 분해

구분	릴레이션 분해	무손실 분해 여부
[분해2]	특강수강(학생번호, 특강이름, 교수) → R3(학생번호, 특강이름), R4(교수, 특강이름)	- R3와 R4의 공통 속성은 특강이름 - 특강이름은 R3나 R4의 키가 아님 → 무손실 분해 규칙 불만족

R3

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

R4

특강이름	교수
소셜네트워크	김교수
인간과 동물	송교수
창업전략	박교수
창업전략	윤교수

2023-10-18

컴퓨터공학과

111

## 2. 손실 분해

- R3, R4 릴레이션을 다시 조인하면 의미없는 튜플이 생김
  - 가짜튜플(spurious tuple)
- 무손실 분해 조건을 만족하지 못하고 손실(loss) 분해됨

특강수강

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

R3 ⋈ R4

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

≠

### 특강수강 릴레이션과 R3 ⋈ R4 릴레이션의 비교

2023-10-18

컴퓨터공학과

112



## 4 정규형

- 다치종속성을 갖는 릴레이션
  - 릴레이션이 **BCNF**이고 **MVD**가 없는 경우
- Multi-valued(One-to-Many) Dependency
  - 다치 종속성
  - 조건
    - X의 각 값에 대하여, 대응하는 Y값이 여러 개 일 때
      - $X \twoheadrightarrow Y$ : X가 Y를 다중값으로 결정한다.
    - 릴레이션은 **적어도 3개 속성**은 가져야 함
    - 릴레이션 R(A,B,C)에 대해,
      - A와 B사이에 MVD가 존재하면 B와 C는 서로 독립적이어야 함

2023-10-18

컴퓨터공학과

113

## 4 정규형

- 4정규형 규칙
  - It should be in the **Boyce-Codd Normal Form**.
  - And, the table should **not have any Multi-valued Dependency**.
- 예: 학생등록 테이블

s_id	course	hobby
1	Science	Cricket
1	Maths	Hockey
2	C#	Cricket
2	Php	Hockey

Right?

Boyce-Codd Normal Form

s_id	course	hobby
1	Science	Cricket
1	Maths	Hockey
1	Science	Hockey
1	Maths	Cricket

2023-10-18

컴퓨터공학과

114

## 4 정규형

MVD준제

연관성 없음

s_id	course	hobby
1	Science	Cricket
1	Maths	Hockey
1	Science	Hockey
1	Maths	Cricket

학생 등록 테이블 분해  
▽  
과목테이블 + 취미테이블

s_id	course
1	Science
1	Maths
2	C#
2	Php

s_id	hobby
1	Cricket
1	Hockey
2	Cricket
2	Hockey

Bad Design

## 4 정규형

다중값 종속성(MVD): 1 정규형에서 비롯됨

고객	유대폰	선호음식
웅길동	1122/8989	버거/피자
이태원	5252	피자

고객	유대폰	선호음식
웅길동	1122	버거
웅길동	8989	버거
웅길동	1122	피자
웅길동	8989	피자
이태원	5252	피자

고객 → 유대폰,

고객 → 선호음식

고객은 유대폰과 선호음식 속성을 다중 결정

함수종속성  $X \rightarrow Y$ 에서 모든  $x$ 는 정확히 하나의 값을 갖는  $y$ 를 결정

## 4 정규영 예

영화

영화명	촬영장소	장르
MV1	한국	코미디
MV1	한국	스릴러
MV2	호주	액션
MV2	호주	범죄
MV3	인도	드라마

4정규영이 아님. 왜?

- 하나 이상의 영화에 동일한 장르 가능
- 촬영 장소에 동일한 영화 가능

## 4 정규영으로 변환 예

영화\_촬영

영화명	촬영장소
MV1	한국
MV2	호주
MV3	인도

영화\_장르

영화명	장르
MV1	코미디
MV1	스릴러
MV2	액션
MV2	범죄
MV3	드라마

## 5 정규형

- **조인 종속성(join dependency)**
  - 하나의 릴레이션을 무손실, 비부가적 분해
  - 다시 조인 시 원래의 릴레이션으로 복원 가능한 경우
- **PJ정규형(PJ/NF)**
  - 릴레이션을 분할하고(Project) 합치는(Join) 개념
  - 릴레이션 R에 속하는 모든 조인종속이 R의 후보키를 통해서만 만족해야 함
  - if it is **in 4NF** and **not contains any join dependency** and joining should **be lossless**.

## 5정규형

- **무손실 조인(Lossless Join)**

■ Decomposition of  $R = (A, B, C)$   
 $R_1 = (A, B)$      $R_2 = (B, C)$

A	B	C
$\alpha$	1	A
$\beta$	2	B

$r$

A	B
$\alpha$	1
$\beta$	2

$\Pi_{A,B}(r)$

B	C
1	A
2	B

$\Pi_{B,C}(r)$

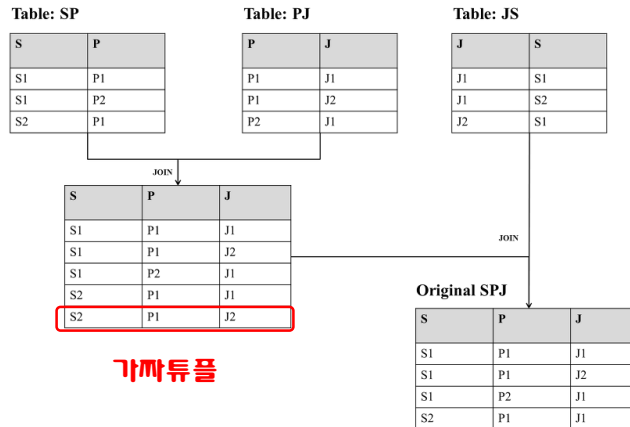
$\Pi_A(r) \bowtie \Pi_B(r)$

A	B	C
$\alpha$	1	A
$\beta$	2	B

## 5 정규형

### • 비부가적 조인(Nonadditive Join)

– 조인 결과에 데이터(가짜 튜플)가 존재하지 않는 조인.



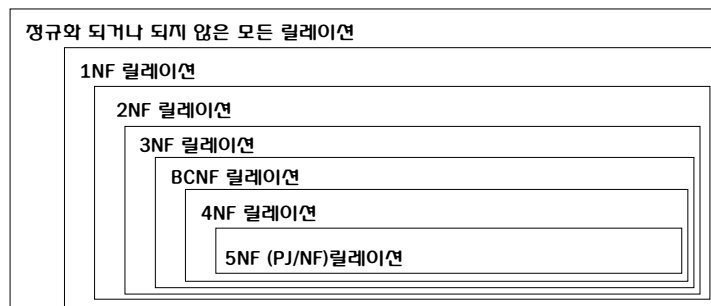
2023-10-18

컴퓨터공학과

121

## 3. 정규화 정리

- 대부분의 릴레이션은 BCNF까지 정규화
- 실제적인 이상현상이 없어짐
- 보통 BCNF까지 정규화를 진행



정규화의 포함 관계

2023-10-18

컴퓨터공학과

122

## 요약

1. 이상현상

2. 압수 증속성

3. 정규화

4. 증속성

5. 무손실 분해