# 맹준영_3, 4장

## 목표

- 왜 정규화가 필요한지?
- 정규화는 어떤 과정인지?

## ▼ 3.1 왜 DB 설계가 중요한가?

- 테이블이 적절하게 설계되지 않으면 쿼리를 통한 데이터 조작이 어렵다.
- 정규화를 통해 테이블의 깔끔한 설계를 가능하게 한다.

## ▼ 3.2 정규화

### 관계형 모델을 보완하는 이론

> 🔥 “정규화”
>
> - RDB를 잘 다루는 데 필요한 기술이며 관계형 모델을 전제로 구축된 DB 설계 이론
> - 관계형 모델을 보완하는 이론

- 정규화 ≠ 관계형 모델의 일부

### 변칙 (Anomalies)

- 정규화의 이점 = 모순 방지
- 변칙 (Anomalies) : 릴레이션을 잘못 설계해 모순이 발생하는 상태
  - SQL의 이상현상 (삽입, 갱신, 삭제 이상)

> 변칙의 예
>
| 이름 | 과목 | 학년 |
| --- | --- | --- |
| 맹준영 | DB | 2 |
| 김민섭 | DB | 2 |
| 맹준영 | Spring Boot | 1 |
| 김민섭 | Spring Boot | 2 |

> 릴레이션은 참의 집합 ⇒ 릴레이션 내 튜플 = 모두 참
>
> ”맹준영”은 2학년일 수도 1학년일 수도 있다. 릴레이션의 값이 명확하게 구분되지 않는다.
> ”모순이 발생한 상태, 즉 변칙이 발생했다.”

- 릴레이션 내에서 변칙이 일어난 이유는 DB 설계가 잘못되었기 때문
- DB의 규모가 크고 복잡하거나 요구사항이 추가되어 응용프로그램의 변경이 일어날 수록 DB의 설계 변경이 일어나며 변칙이 발생할 가능성이 존재

> “DB의 최초 설계도 중요하지만 실무에서 요구사항 변경으로 인한
> 컬럼 추가, 제약 조건 변경에 대해서도 변칙이 일어나지 않게 집중적인 고려 필요”

### 변칙이 발생하는 이유

- 릴레이션에 같은 데이터가 중복되는 경우 변칙이 발생할 가능성이 있다.
- 위의 예에서 2번째 튜플 (”김민섭, DB, 2”)를 1학년으로 변경하고자 한다면 “김민섭”이 있는 모든 행을 변경 필요
- 결국 모순의 원인은 “중복”

> 🔥 **“모순이 생기지 않게 중복을 제거하는 작업 → 정규화를 진행하는 가장 큰 목적”**

## ▼ 3.3 정규형

- 정규화의 각 단계를 “정규형 (Normal Form, NF)”이라고 한다.
- 제 1정규형 ~ 제 6 정규형까지 존재하며, 제 3정규형과 제 4정규형 사이에 보이스코드 정규형(BCNF)이 존재
- 각 정규형은 이전 단계까지의 정규형의 조건을 만족한다.
- 일반적으로 DB 설계는 BCNF나 5NF를 목표로 한다.

### 🔥 제 1 정규형 (1NF)

- 테이블이 릴레이션이어야 한다.
- 1NF의 조건
  - 행이 위에서 아래로 정렬돼 있지 않다.
  - 열이 왼쪽에서 오른쪽으로 정렬돼 있지 않다.
    - SQL은 컬럼과 행에 순서가 존재
    - 순서 의존 쿼리를 사용하지 않는다.
  - 중복하는 행이 존재하지 않는다.
    - 기본키, 유니크 제약 조건으로 해결 가능
    - 실제 저장 값이 중복되지 않도록 하는 것
    - 행 전체는 중복이 없더라도 테이블에 중복이 발생할 수 있다.
  - 열의 값은 도메인에 속하는 요소의 값을 딱 한개만 가진다.
    - 의미가 있는 한 묶음의 데이터를 한 단위로 취급한다. = 도메인의 요소
    - 집합의 요소는 더이상 분해 불가능한 원자값 = 도메인
    - 열의 값은 도메인 중에서 골라낸 요소의 하나
  - 모든 열의 값은 정의된 것이어야 하고 각 행은 항상 존재한다.
    - NULL이 포함되지 않도록 설계 고려

> **열의 값은 반드시 스칼라여야 하는가?**
>
> 아니다. 도메인에 포함된 데이터의 종류에는 특별한 제한이 없다.
> 스칼라인 것 ≠ 원자인 것

- 1NF의 조건을 만족하지 않는 테이블의 예 → 반복그룹
- 하나의 컬럼에 여러 개의 값을 할당하는 것
- ex) 종목에 “농구”, “배드민턴”으로 두 개의 값이 할당 되었다.
  - Key : 이름

| 이름 | 종목 |
| --- | --- |
| 손흥민 | 축구 |
| 맹준영 | 농구, 배드민턴 |

- 반복그룹 대신 여러 개의 행으로 데이터를 저장 → 데이터 중복 제거, NULL 배제
  - Key : 이름, 종목

| 이름 | 종목 |
| --- | --- |
| 손흥민 | 축구 |
| 맹준영 | 농구 |
| 맹준영 | 배드민턴 |

- 반복그룹을 여러 개의 행으로 데이터를 저장함으로써 키가 변경된다.

## 후보키와 슈퍼키

☁️ **후보키**
→ 릴레이션에 포함된 튜플의 값을 고유하게 하는 속성의 집합 (여러 개 존재 가능)
→ 후보키에 포함되지 않는 속성 = 키가 아닌 속성

**슈퍼키**
→ 후보키의 슈퍼셋, 후보키에서 추가 속성을 가지는 키
→ 후보키는 슈퍼키의 일종, 릴레이션의 제목 (모든 속성)은 슈퍼키이다.

## 🔥 함수 종속성 (Functional Dependancy, FD)

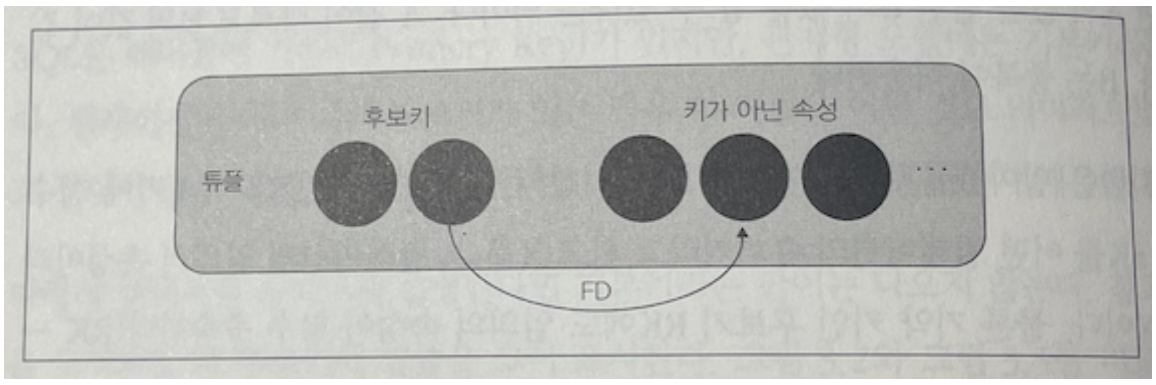
☁️ **함수 종속성**
→ 릴레이션의 속성의 부분집합 = A, B
→ 릴레이션 내 모든 튜플, A의 값이 같다면 B의 값도 같은 경우

**⇒ B는 A에 함수 종속이다. = A의 값을 알면 B의 값을 알 수 있다.**

[표현식]
{A} → {B}

- A를 통해 B를 알 수 있으므로 A를 통해 B를 식별 가능 = A는 키의 성질을 갖는다.
- 2NF ~ BCNF는 이러한 A는 키의 성질을 구분하는 것

## 🔥 제 2 정규형 (2NF)




- 후보키의 진부분집합에서 키가 아닌 속성에 함수 종속성을 제거하는 작업
  - 후보키의 임의의 속성이 후보키의 속성 이외의 속성에 함수 종속인 경우 = 부분 함수 종속성
  - 부분 함수 종속성을 제거하는 작업

> 진부분집합 : 부분집합 중에 원래 자신의 집합을 제외한 것

| 이름 | 학년 | 종목 |
|---|---|---|
| 맹준영 | 2 | 배드민턴 |
| 맹준영 | 2 | 축구 |
| 김민섭 | 4 | 헬스 |
| 김민섭 | 4 | 테니스 |
| 손흥민 | 3 | 축구 |

- 모든 튜플은 중복이 아니다.  (1NF 만족)
- 2NF를 만족하지 않는다.
  - (이름, 종목)이 후보키이다.
  - 후보키의 진부분집합 : 이름 → 키가 아닌 속성 : 학년 ⇒ 부분 함수 종속
- (이름, 학년) 튜플만 놓고 보면 중복이 존재한다. → 자명하지 않은 함수 종속성, 키의 성질을 갖는다.
- 한 개의 릴레이션에 대해 프로젝션을 통해 두 개의 릴레이션을 생성한다. = 무손실 분해

> 무손실 분해 : 분해한 다음에 릴레이션에 포함된 정보를 사용해 원래의 릴레이션을 재구축이 가능한 것

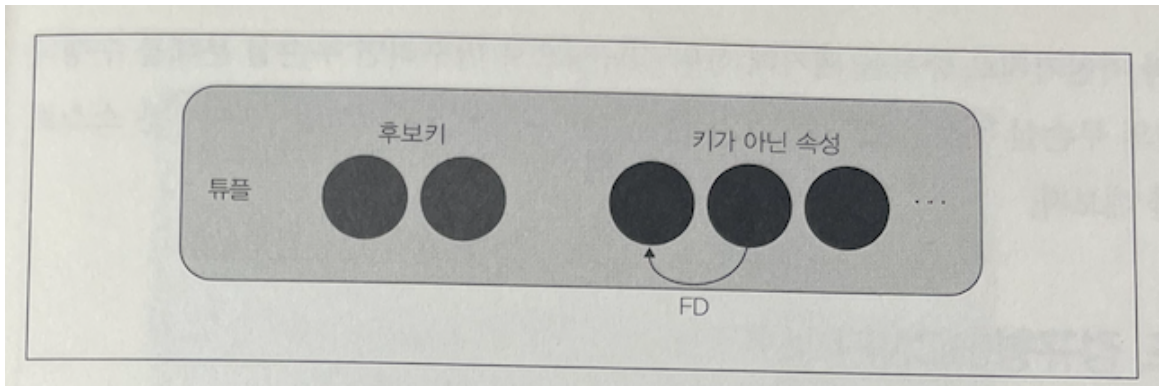
| 이름 | 종목 |
|---|---|
| 맹준영 | 배드민턴 |
| 맹준영 | 축구 |
| 김민섭 | 헬스 |
| 김민섭 | 테니스 |
| 손흥민 | 축구 |

| 이름 | 학년 |
|---|---|
| 맹준영 | 2 |
| 김민섭 | 4 |
| 손흥민 | 3 |

- 프로젝션을 통한 무손실 분해를 진행
- JOIN을 통해 원래의 릴레이션으로 재구축이 가능하다.

## 🔥 제 3 정규형 (3NF)

- 추이 함수 종속성을 제거하는 작업

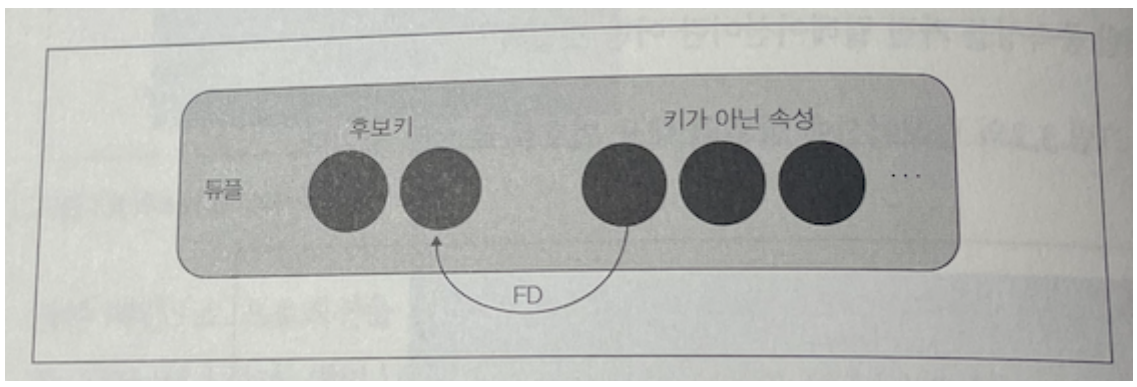



## 추이 함수 종속성 : 키가 아닌 속성끼리의 함수 종속성

| 이름 | 학과 | 대표번호 |
|---|---|---|
| 맹준영 | DB | 123 |
| 김민섭 | DB | 123 |
| 백예린 | JAVA | 999 |

- 키가 아닌 속성인 (학과)가 키가 아닌 속성 (대표번호)에 함수 종속성이 존재 = 학과 → 대표번호
- 학과와 대표번호를 무손실 분해를 통해 릴레이션으로 만드는 것이 가능

### 🔥 보이스 코드 정규형 (BCNF)




- 자명하지 않은 함수 종속성 (튜플 전체는 중복 X, 일부 튜플 중복 존재) 제거하는 작업
- BCNF 이상의 정규형에는 무손실 분해가 불가능하다.
- 키가 아닌 속성에서 후보키의 진부분집합에 대한 함수종속성을 제거하는 것

> **후보키의 진부분집합(A) → 후보키의 진부분집합(B)을 만족하는 함수종속성은 존재하는가?**
>
> 존재하지 않는다.
> A를 통해 B를 결정할 수 있다면 애초에 A가 B없이도 키의 역할을 하므로 후보키라는 가정에 어긋난다.

## ▼ 3.4 요약

- BCNF를 진행하면 자동적으로 5NF 조건을 만족하는 경우가 많다.
- 이상적인 정규형을 만족하기 위해선 함수종속성으로는 한계가 존재
- 결합 종속성의 개념을 통해 4NF ~ 6NF를 만든다.

## ▼ 4.1 결합 종속성 (JD)

- BCNF를 통해 후보키, 키가 아닌 속성의 함수종속성을 제거했지만, 중복을 모두 제거한 것은 아니다.

☁️ 결합 종속성 (Join Dependency, JD)
→ 키 자체에 다중성이 포함됐을 때 나타나는 중복

[정의]
A, B, … C를 릴레이션 R의 제목의 부분집합이라고 할 때
A,B,C의 프로젝션에 대응하는 릴레이션을 결합한 결과 = 릴레이션 R 이면 “결합 종속성”이다.

[표현식]
☆{A, B, …, C}

- A, B, C의 제목 중 하나가 R의 제목 전체와 같은 경우 “결합 종속성이 명백하다.”
- 릴레이션 내에 명백하지 않은 결합 종속성이 존재하면 4NF ~ 6NF의 대상이다.
- 결합 종속성은 프로젝션으로 나눈 릴레이션을 다시 결합하면 원래의 릴레이션으로 돌아가는 성질 → 무손실 분해 가능
- 함수 종속성은 결합 종속성의 일종이다.

## 암시적인 결합 종속성

- 암시적인 결합 종속성 : 슈퍼키로 분해할 수 있는 결합 종속성, 즉 공통의 후보키를 갖는 결합 종속성
- 아래의 예는 {이름}을 후보키로 갖는 두 개의 릴레이션으로 나누어 튜플의 수가 같고 키가 아닌 속성만 다른 릴레이션을 두개 생성한 것
  → 결합하면 원래의 릴레이션과 동일

| 이름 | 나이 | 학년 |
|---|---|---|
| 맹준영 | 28 | 2 |
| 김민섭 | 28 | 2 |
| 백예린 | 26 | 1 |
| 손흥민 | 31 | 4 |

| 이름 | 나이 |
|---|---|
| 맹준영 | 28 |
| 김민섭 | 28 |
| 백예린 | 26 |
| 손흥민 | 31 |

| 이름 | 학년 |
|---|---|
| 맹준영 | 2 |
| 김민섭 | 2 |
| 백예린 | 1 |
| 손흥민 | 4 |

## 키가 아닌 속성과 결합 종속성

- 키가 아닌 속성이 존재하는 경우 후보키는 키가 아닌 속성을 결정할 수 있는 함수 종속성이 존재한다.
  - 프로젝션으로 후보키의 속성을 나누면 함수종속성이 사라지므로 무손실 분해가 불가능하다.
- 4NF, 5NF 작업은 키가 아닌 속성이 존재하지 않는 릴레이션만 대상
- 4NF, 5NF는 후보키에 여러 속성이 포함되는 복합키인 경우만 필요한 작업

☁️ BCNF가 바로 5NF까지 만족하는 경우

**1. 키가 아닌 속성이 존재할 때**
**2. 키에 포함되는 속성이 1개뿐인 단일키일 때**

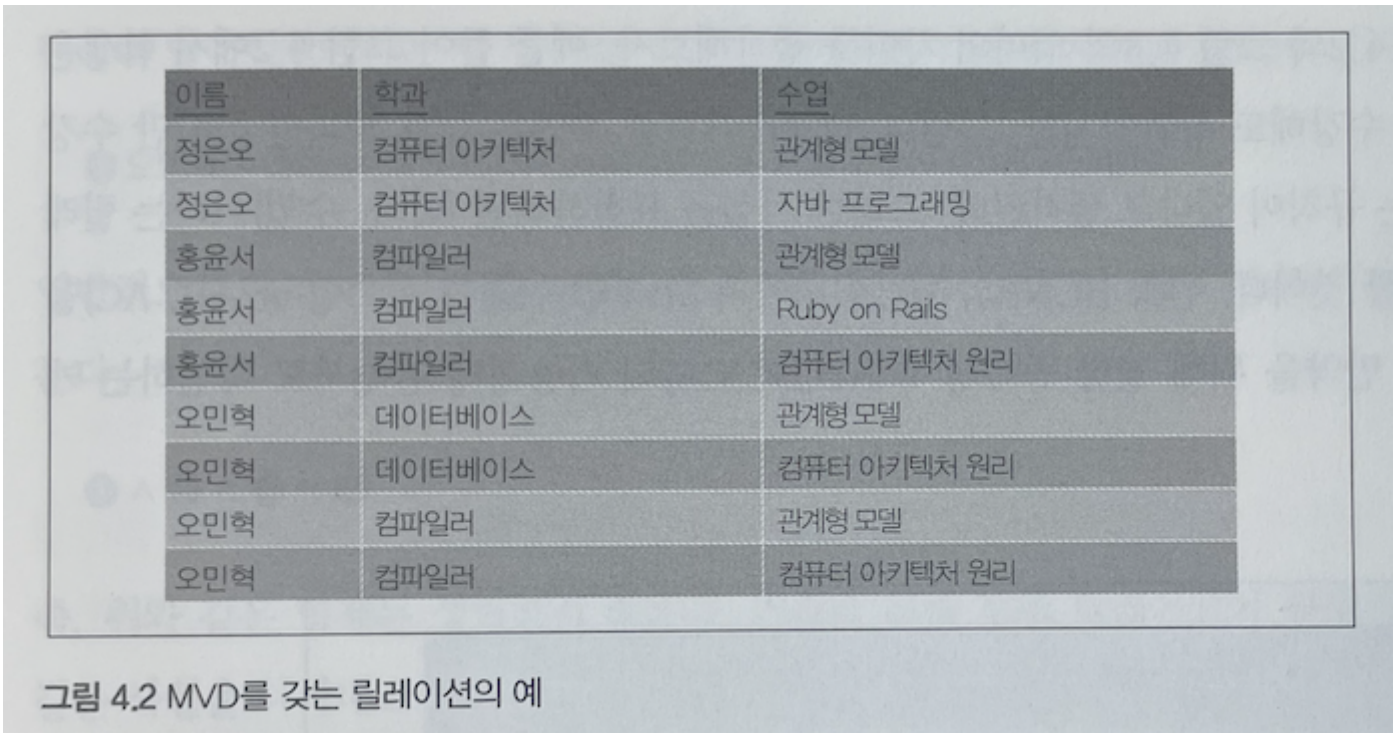
> 키가 아닌 속성이 존재하는 경우 후보키를 무손실 분해할 수 있는 결합 종속성이 존재하지 않는가 ?
> ex) R = {a,b,c,x}
>
> if. 후보키 {a,b,c} → 키가 아닌 속성 {x} : 후보키를 통해 x를 결정
> - {a,b,c}를 분해하는 경우 x를 결정 못해 함수종속성이 사라짐 → 무손실 분해 불가능

> if. {a,b,c} → {x}에서 {a,b} → {x}도 가능하다면 후보키의 진부분집합 {a,b}가 {x}를 결정하는 부분 함수종속이 존재하므로 2NF 조건에 위배
>
> if. {a, b} → {x} 인 경우 함수 종속성을 제거하기 위해 {a,b,c}, {a,b,x}의 두 릴레이션으로 무손실 분해 가능
> → {a,b,c} = 후보키이면서 키가 아닌 속성을 갖지 않으므로 정규화 대상 검토 필요

## ▼ 4.2 결합 종속성에 의한 정규화 (4NF ~ 6NF)

### 제 4 정규형 (4NF)

- 다치 종속성 (MultiValued Dependency, MVD)에 의한 정규화
  - 표현식 : A→→B, A→→C
  - ☆{AB, A
- 다치 종속성 : 키가 아닌 속성을 갖지 않는 릴레이션을 결합 종속성에 의해 공통의 속성을 포함한 두 개의 릴레이션으로 무손실 분해할 수 있는 것



| 이름 | 학과 | 수업 |
|---|---|---|
| 정은오 | 컴퓨터 아키텍처 | 관계형 모델 |
| 정은오 | 컴퓨터 아키텍처 | 자바 프로그래밍 |
| 홍윤서 | 컴파일러 | 관계형 모델 |
| 홍윤서 | 컴파일러 | Ruby on Rails |
| 홍윤서 | 컴파일러 | 컴퓨터 아키텍처 원리 |
| 오민혁 | 데이터베이스 | 관계형 모델 |
| 오민혁 | 데이터베이스 | 컴퓨터 아키텍처 원리 |
| 오민혁 | 컴파일러 | 관계형 모델 |
| 오민혁 | 컴파일러 | 컴퓨터 아키텍처 원리 |

그림 4.2 MVD를 갖는 릴레이션의 예

- {이름, 학과}, {이름, 수업}으로 릴레이션을 무손실 분해 가능

### 제 5 정규형 (5NF)

- 명백하지 않거나 암시적이지 않은 결합 종속성을 제거하는 것
  - 표현식 : A→→B, B→→C, C→→A
  - ☆{AB, BC, CA}
- BCNF를 한 릴레이션에서 ☆{AB,AC}을 발견한 경우 ☆{AB, BC, CA}의 가능성도 확인하는 작업 필요

### 제 6 정규형 (6NF)

- 릴레이션 자신을 포함한 결합 종속성만 남을때까지 모든 결합 종속성을 제외하는 상태
- 키가 아닌 속성의 개수가 0개 또는 1개가 될때까지 무손실 분해한 상태
- 불필요한 결합이 많아지며 실용적이지 않다.

## ▼ 4.3 요약

- 함수 종속성은 결합 종속성의 일종
- BCNF까지 도달한 릴레이션 = 후보키가 단일 속성으로 구성되어 있거나 키가 아닌 속성이 존재 하지 않는 경우 5NF를 만족
- 5NF에 도달한 릴레이션 = 중복이 없다고 할 수 있다.