Normalization & View

2025.11.06

SSAFY 서울 7반 송현우

INDEX

Normalization

Objective

1. 고려되지 않은 삽입, 갱신, 삭제 의존에서부터 관계의 집합을 배제한다.

2. 새로운 자료형이 나타날 때, 관계들의 집합의 재구성의 필요성을 낮추고, 그로 인하여 응용 프로그램의 생명주기를 연장한다.

3. 사용자에게 관계 모델을 더욱 의미있게 한다.

4. 관계들의 집합을 질의의 통계로부터 중립적이게 한다. 질의들은 시간이 지남에 따라 변경되기 때문.

—E.F. Codd, "Further Normalisation of the Data Base Relational Model"

What is Normalization?

➔ 인터뷰어 : ‘정규화’는 어디에서 온 것입니까?

➔ 코드 박사 : 데이터베이스 디자인에 어떤 원리가 도입되는 것이 내게 필수적인 것 같았습니다. 당시 닉슨 대통령이 중국과의 관계 **정상화(normalization)**에 대해 많은 이야기를 하고 있었기 때문에 나는 그것을 정규화(normalization)라 불렀습니다.

What is Relation?

- ❏ Relation = 테이블
- ❏ Tuple = 행
- ❏ Attribute = 열

??

What is Relation?

➢ Relation(relation variable) = **술어**(Predicate)

공급자 S가 부품 P를 N개 수량으로 공급한다

➢ Tuple = **명제**(proposition)

공급자 S1이 부품 P1을 300개 수량으로 공급한다

=> 시점 T에서 관계 변수 R은 참인 명제를 나타내는 튜플을 갖는다

\+ Closed world assumption(CWA)을 전제함

Practice

1. 관계는 튜플에 대해 순서가 없다
2. 관계는 속성에 대해 순서가 없다
3. 관계는 이름 없는 속성을 갖지 않는다
4. 관계는 같은 이름을 가진 속성을 갖지 않는다
5. 관계는 중복된 튜플을 포함하지 않는다
6. 관계는 NULL을 포함하지 않는다
7. **관계는 1차 정규형(1NF)이다**

주의 : 관계형 이론과 SQL은 구분되어야 한다 ! !

1NF(의 느슨한 정의)

1. 1NF는 모든 속성이 원자값(atomic value)이어야 한다.

   => 원자값 = 한 개의 값 ?

2. 1NF는 테이블에 필드의 repeating group이 없어야 함을 의미한다.

   => repeating group이 뭔데

3. 테이블이 1NF이면 테이블은 ‘관계’이며, 중복되는 항목이 없어야 한다.

   => 중복이 뭔데..

Atomicity

에드거 F. 커드 박사의 재정의

1970년 : 원자성 = 비분해성(non-decomposable) !!

1. 정수 : (다소 억지스럽지만) 소인수로 분해될 수 있다.
2. 전화 번호 : 010, 1234, 5678로 분해될 수 있다.
3. 이름 : Last name, First name
4. …

Atomicity

에드거 F. 커드 박사의 재재정의

1990년 : 비분해성이란 DBMS에서의 비분해성을 의미한다.

C. J. Date 박사 : 그러면, 다음과 같은 경우는?

1. 문자열 : 문자열은 문자로 분해할 수 있다. (SUBSTRING, LIKE 연산자)
2. 날짜 및 시간 : 연/월/일/시간/분/초 성분으로 분해할 수 있다.
3. 실수 : 정수부와 소수부로 분해할 수 있다.
4. 관계 : View로 분해할 수 있다.
5. …

1NF(C. J. Date ver.)

"어떤 관계와 동일 구조"이면 1NF이다 !!

1. 행에는 위아래의 순서가 없다.
2. 열에는 좌우의 순서가 없다.
3. 중복되는 행이 없다.
4. 모든 열과 행의 교차점에는 적용 가능한 타입의 값이 한 개만 존재한다. (Atomicity의 대안)
    a. 해당 타입은 무엇이든 될 수 있다.(심지어는 관계 타입도)
    b. repeating group에 대한 논의를 철회한다
5. 모든 열은 정규열이다.
    a. 각 열은 적절한 이름이 존재하며 중복되지 않는다.
    b. 숨겨진 요소가 존재하지 않는다.

-- Chris Date, "What First Normal Form Really Means"

위키백과에 행, 열이 바뀌어 표기되어 있음.. 주의

1NF(C. J. Date ver.)

4-a. 해당 타입은 무엇이든 될 수 있다.(심지어는 관계 타입도)

=> RVA(relation valued attribute)

G

X	Y
86	92
93	23
23	30

G-named

Name	X	Y
Graph A	86	92
Graph A	93	23
Graph A	23	30
Graph B	97	33
Graph B	6	36
Graph B	12	17
Graph C	11	39
Graph C	20	90
Graph C	2	4

OR

G-nested

G

X	Y
86	92
93	23
23	30

X	Y
97	33
6	36
12	17

X	Y
11	39
20	90
2	4

이미지 출처 : http://www.dpxo.net/articles/rvanormalform.html

1NF(C. J. Date ver.)

Relation

=

1NF

=

Normalization

모든 정규화된 관계가 1NF 이내에 있다.

정규화는 본래 1NF만을 의미했음

Practice format

S

SNO	SNAME	STATUS	CITY
S1	Smith	20	London
S2	Jones	30	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

P

PNO	PNAME	COLOR	WEIGHT	CITY
P1	Nut	Red	12.0	London
P2	Bolt	Green	17.0	Paris
P3	Screw	Blue	17.0	Paris
P4	Screw	Red	14.0	London
P5	Cam	Blue	12.0	Paris
P6	Cog	Red	19.0	London

SP

SNO	PNO	QTY
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300
S4	P5	400

Decomposition

S

SNO	SNAME	STATUS	CITY
S1	Smith	20	London
S2	Jones	30	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

SNC

SNO	SNAME	CITY
S1	Smith	London
S2	Jones	Paris
S3	Blake	Paris
S4	Clark	London
S5	Adams	Athens

CT

CITY	STATUS
Athens	30
London	20
Paris	30

1. 중복성 제거
2. 무손실 (관계값)
3. 정보 동치 (관계 변수)

Decomposition

➢ 과연 정보 동치인가?

- 술어를 살펴보자!
  - **기존 술어 S**
    - 공급자 SNO는 SNAME으로 명명됐으며, STATUS라는 상태를 갖는 도시 CITY에 위치해 있다
  - **분해 후 술어 SNC, CT**
    - 공급자 SNO는 SNAME으로 명명됐으며, 도시 CITY에 위치해 있다
    - 도시 CITY는 STATUS라는 상태를 갖는다.

몇 가지 가정..

1. 도시에 공급자가 없더라도 상태를 가질 수 있다고 가정

   => 관계 변수 S는 현실 세계의 문제 상태를 반영하지 못하는 '**논리적으로 잘못된 관계**'
2. 도시에 상태가 없더라도 공급자가 도시에 위치할 수 있다고 가정하자

   => 위와 동일
3. 그렇다면, 관계 변수 S가 현실 세계의 상황을 충실히 반영하려면 어떤 가정이 필요할까? (정보 동치)

Decomposition

정보 동치를 만족하기 위해서는 무결성 제약조건이 추가되어야 한다!

CONSTRAINT … SNC {CITY} = CT {CITY} ;

➢ 즉, 분해는 두 가지 다른 문제를 해결한다.

1. 논리적으로 잘못된 디자인을 고칠 수 있다.
2. 중복성을 해결할 수 있다.

➢ 또한 세가지 제약 가능성을 고려해야 한다.

1. 동등 종속성 EQD (= 정보 동치, 엄밀한 분해)
2. 삽입 종속성 IND (≒ 외래 키)
3. 제약 없음

결국, 분해는 제약 조건**(trade-off)**을 수반할 수 있다.

E/R Model

술어와 제약

술어와 제약 조건은 디자인 작업과 매우 관련이 있다.

1. **가능한 신중하게 관계 변수 술어를 정한다.**
2. **그 술어와 규칙들을 관계 변수와 제약 조건으로 매핑한다.**

위 내용은 E/R Model와 같은 방법론과 맞지 않다.
E/R Model은 quantifier(정량자)에 대한 지원이 충분하지 않고, 결과적으로 ERD는 소수의 제한적인 제약 조건을 제외한 모든 제약 조건을 나타낼 수 없다.
ERD가 높은 추상화 수준으로 해석할 수는 있지만 전체 디자인이 될 수는 없다.

Update Anomaly

삽입 이상(insertion anomaly)

- 로마에 공급자가 없으면 로마의 상태가 10이라는 사실을 삽입할 수 없다

삭제 이상(deletion anomaly)

- 아테네의 공급자를 삭제하면 아테네의 상태가 30이라는 사실을 잃게 된다

수정 이상(modification anomaly)

- 특정 도시의 상태를 변경하려면 모든 도시를 동일하게 수정해야한다.

Normal Forms
(FD)

Normalization layer

1NF

2NF

3NF

BCNF (FD)

4NF

5NF (JD)

FD

S

SNO	SNAME	STATUS	CITY
S1	Smith	20	London
S2	Jones	30	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

$x \rightarrow y$

x : 결정 요인(determinant)

y : 종속 요인 (dependent)

- {CITY} → {STATUS}
- {SNO} → {SNAME, STATUS}
- {SNO, CITY} → {STATUS}

결정 요인에 속성을 추가하거나,

종속 요인으로부터 속성을 제거해도

함수 종속 성립

Key

Candidate Key가 될 수 있는 조건

1. 관계 변수 R의 속성의 부분집합
2. 유일성 : K에 대해 동일한 값을 갖는 서로 다른 튜플을 갖지 않는다.
3. 축소 불가능(irreducible) : K의 어떠한 부분집합도 유일성을 갖고 있지 않는다.

단, 관계 변수가 단 하나의 후보키만을 가진다는 조건은 없다 !!

Key

키 속성 : 관계 변수 R의 속성 A가 적어도 하나의 후보키에 속하면 키 속성이다.

비키 속성 : 관계 변수 R의 속성 A는 R의 어떠한 후보키에도 속하지 않으면 비키 속성이다.

예를 들어, 관계변수 SP에서 SNO와 PNO는 키 속성이고 QTY는 비키 속성이다.

상위 키(super key) : SK에 대해 동일한 값을 가진 두 개의 구별되는 튜플을 포함하지 않을 때, SK는 상위 키이다.

즉, 유일성만 만족하면 상위 키이다.

ex) {SNO, PNO}, {SNO, PNO, QTY}

하위 키(sub key) : SK는 R의 하나 이상의 키의 부분집합일 경우 하위 키이다.

ex) {SNO, PNO}, {SNO}, {PNO}, {}

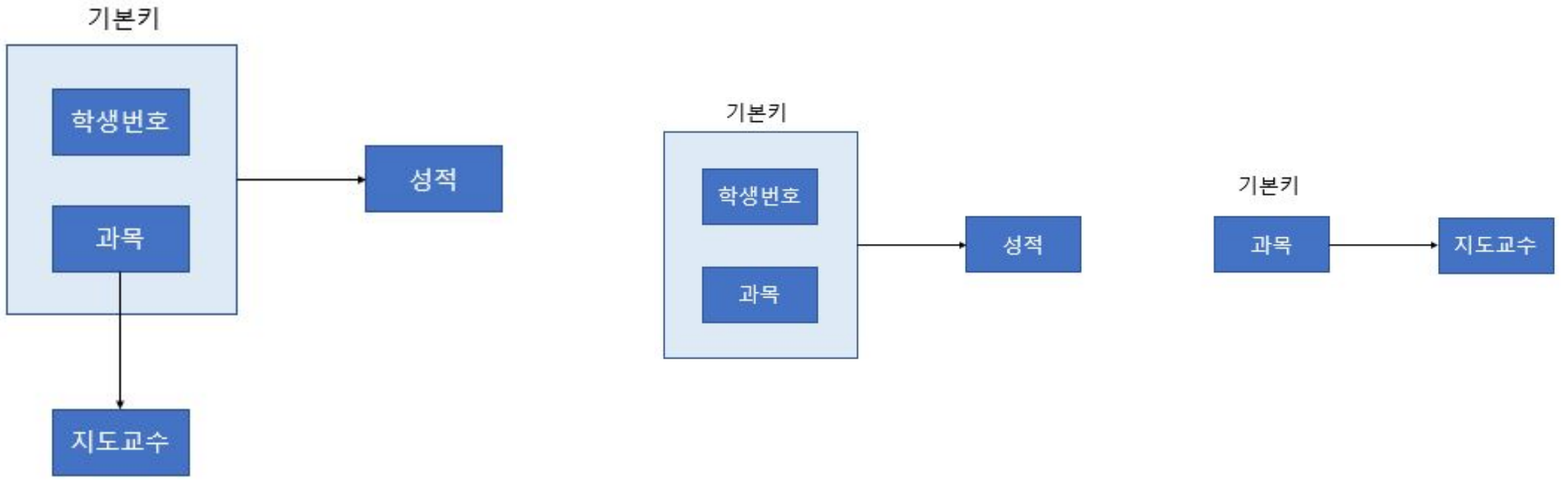
SP

SNO	PNO	QTY
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300
S4	P5	400

2NF

익숙한 정의

1NF에서 Partial Dependency(부분 함수종속)을 제거하면 2NF이다.



이미지 출처 : https://code-lab1.tistory.com/48

C. J. Date : 1NF를 전제로 2NF의 정의를 내리는 형식은 1NF를 이해하지 못한 것이다.

2NF

엄밀한 정의

R의 모든 키 K와 R의 모든 비 키 A에 대해 FD [K → {A}] 가 축소 불가능한 경우에 R은 2NF이다.
(FD의 축소 불가능성은 결정 요인의 축소 불가능성을 뜻함)

= 모든 비 키가 키 전체에 함수 종속이다

유용한 정의

R에서 성립하는 모든 non-trivial한 FD [X → Y]에 대해 다음 중 하나 이상이 참이면 2NF이다.

1. X는 상위 키이다.
2. Y는 하위 키이다.
3. X는 하위 키가 아니다.

위 정의는 논리적 동치이다. 추후 증명

non-trivial만 보는 이유는 SP만 해도 FD가 31개라서!

2NF Practice

ID	학생명	과목명	수업시간
101	김동민	운영체제	90분
101	김동민	컴퓨터구조	180분
102	만쥬	운영체제	90분
103	춘장	컴퓨터구조	180분
104	건덕이	모바일프로그래밍	120분
105	건구스	운영체제	90분

이미지 출처 : https://minboykim.github.io/database/DB_Normalization/

S

SNO	SNAME	STATUS	CITY
S1	Smith	20	London
S2	Jones	30	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

3NF

익숙한 정의

2NF에서 Transitive Dependency(이행 종속)을 제거하면 3NF이다.

이미지 출처 : https://code-lab1.tistory.com/48

C. J. Date : 2NF를 전제로 3NF의 정의를 내리는 형식은 별로다.

3NF

정의

R에서 성립하는 모든 non-trivial한 FD [X → Y]에 대해 다음 중 하나 이상이 참이면 3NF이다.

1. X는 상위 키이다.
2. Y는 하위 키이다.

직접 해본 느슨한 증명

1. 비 키는 상위 키가 아니고, 하위 키가 아니다.
2. 이행 종속은 X, Y가 비 키일 때 나타난다.
3. 따라서 X는 하위키가 아니므로(3번 조건) 2NF를 만족하지만, 3NF를 만족하지 않는다.

사실 2NF, 3NF는 BCNF로 가기 위한 디딤돌일 뿐이다.

BCNF

익숙한 정의

3NF에서 모든 X가 후보 키 집합에 속하면 BCNF이다.

이미지 출처 : https://code-lab1.tistory.com/48

BCNF

정의

R에서 성립하는 모든 non-trivial한 FD [X → Y]에 대해 다음이 참이면 BCNF이다.

1. X는 상위 키이다.

“모든 사실은 단지 키(super key)에만 존재한다”

BCNF인 이유 ?

→ 3NF를 정의한 이후 나중에 정의했는데 이미 다른 사람이 4NF를 발표해버려서!!!

BCNF Practice

SNP

SNO	SNAME	PNO	QTY
S1	Smith	P1	300
S1	Smith	P2	200
S1	Smith	P3	400
..	.....	..	...
S2	Jones	P1	300
S2	Jones	P2	400
..	.....	..	...

Key : {SNO, PNO}, {SNAME, PNO}

{SNO} → {SNAME}

{SNAME} → {SNO}

어떤 사실이 슈퍼키가 아닌 X에 의해 결정된다.

3NF이지만, BCNF를 만족하지 못한다.

BCNF Practice

고객아이디	강좌명	강사번호
ID7785	데이터베이스	T001
ID1023	C언어	T002
ID2309	데이터베이스	T001
ID2309	C언어	T004
ID4982	데이터베이스	T003
ID4982	C언어	T004

- 해당 표에서 나타나고 있는 정규형은 무엇인가?

Proof

앞선 엄밀한 정의를 정의α, 유용한 정의를 정의β라 하자.

1. β ⊂ α = β → α = ~α → ~β

R은 α에 의해 2NF가 아니라고 하자.

따라서, 비 키 A에 대해 존재하는 FD[K→ {A}]에서 K는 축소 가능하다.

축소 가능한 FD[K→ {A}]는 적절한 하위 키 X(X⊂K)에 대해 성립한다.

이때, {A}를 Y라 하면 FD[X→Y]는 β에 의해 R은 2NF가 아니다.

2. ~β → ~α

R은 β에 의해 2NF가 아니라고 하자.

FD[X→ Y]는 R에서 X가 상위 키가 아니고, Y는 하위 키가 아니며, X는 하위 키다.

Y가 비키 속성 A를 포함하는 경우, Y는 축소 가능하므로 R은 α에 의해 2NF가 아니다.

Y가 비키 속성 A를 포함하지 않는 경우, Y의 하위 집합은 모두 하위 키이다. 이는 3NF를 만족하므로 모순이다.

Lossless decomposition

충분조건 - 히스의 정리 요약ver.

➢ 관계 R의 투영인 X, Y, Z에 대해 관계 R이 FD[X→Y]을 만족하는 경우, XY와 XZ로의 투영의 결합이 R이다.

이는 곧 FD가 lossless의 조건임을 의미한다.

참고로 정리에 대한 증명은 A4 2장분량이니 궁금하면 책을 참고하자!

히스의 정리는 충분 조건인 것에 유의하자

대부분의 문헌에서 이에 대해 명시되지 않고 있음

필요충분조건은 파긴의 정리

SNT

SNO	SNAME	STATUS
S1	Smith	20
S2	Jones	30
S3	Blake	30
S4	Clark	20
S5	Adams	30

CT

CITY	STATUS
Athens	30
London	20
Paris	30

손실 분해의 예시.

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

A	B	C
α	1	A
β	2	B

r

A	B
α	1
β	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
α	1	A
β	2	B

이미지 출처 : https://swingswing.tistory.com/92

Multi-relvar constraint

SNC

SNO	SNAME	CITY
S1	Smith	London
S2	Jones	Paris
S3	Blake	Paris
S4	Clark	London
S5	Adams	Athens

ST

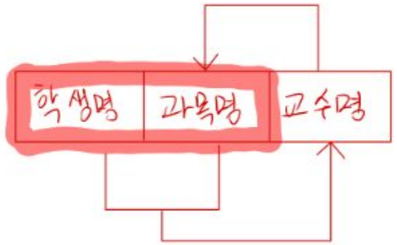
SNO	STATUS
S1	20
S2	30
S3	30
S4	20
S5	30

- 관계 변수 SNC와 ST는 모두 BCNF이다.
- 그러나 FD[{CITY}→ {STATUS}]가 손실되었다.
- 이는 제약 조건이 다중 관계 변수 제약 조건으로 대체됐다는 것을 의미한다.

  (도시의 개수가 도시-상태 쌍 수와 동일해야함)
- S1의 도시의 이름을 바꾸는 경우를 생각해보자.
- 다중 관계 변수 제약 조건을 명시하기는 어렵다.
- 따라서, FD를 보존하는 분해를 선택해야한다.
- FD를 보존하는 분해는 서로 독립적으로 갱신될 수 있다.(독립 투영, 리사넨의 법칙)

FD Retention

학생명	과목명	교수명
김동민	운영체제	진철수
김동민	컴퓨터구조	박철수
건덕이	모바일프로그래밍	지영희
건구스	운영체제	진철수
만쥬	운영체제	김철수



학생명	교수명	:	:	교수명	과목명
김동민	진철수	:	:	진철수	운영체제
김동민	박철수	:	:	박철수	컴퓨터구조
건덕이	지영희	:	:	지영희	모바일프로그래밍
건구스	진철수	:	:	김철수	운영체제
만쥬	김철수	:	:		

이미지 출처 : https://minboykim.github.io/database/DB_Normalization/

- superkey는 {학생명, 과목명}, {학생명, 교수명} 이다.
- FD[{교수명}→{과목명}]를 보면 3NF는 만족하지만 BCNF를 만족하지 못한다.
- 히스의 정리를 통해 분해하면 무손실 분해를 통해 BCNF를 만족하는 것이 가능하다.
- 하지만 FD[{학생명, 과목명}→{교수명}]을 상실한다.
- 이는 다중 관계 변수 제약 조건을 야기한다.
- 즉, BCNF투영과 FD보존을 모두 달성하는 것이 항상 가능하지 않다.
- 물론, 교수가 아직 학생을 가르치지 않더라도 과목을 신설할 수 있으려면 FD보존을 포기하는 편이 바람직하다(IND)
- **때로는 통상적인 정규화 절차가 FD보존을 해칠 수 있다**(예시는 생략)

Normalization procedure

- 정규화 절차는 반드시 1NF, 2NF, 3NF, BCNF 순서대로 작동할 필요가 없다.
- 오히려 FD 보존을 해칠 수 있다.
- BCNF로의 FD 보존 분해 절차가 존재한다.(생략)
- 그럼에도, 피할 수 없는 충돌은 존재하며 이 경우를 특정할 수 있다.(생략)

Denormalization

Denormalization

성능을 위한 탈정규화?

- 이상적인 시스템에서는 논리적인 차원에서 탈정규화할 이유가 없다.
- 즉, 성능 향상을 위한 다른 모든 전략이 요구 사항을 충족하지 못할 경우에만 정규화에서 물러나야한다.
- 논리적으로 탈정규화 같은 문제를 걱정하는 유일한 이유는 현대의 SQL제품 일부의 실패 탓이다.

What is Denormalization?

정의

관계 변수 집합 R1,...,Rn을 탈정규화 한다는 것은 다음과 같은 방법으로 중복성을 증가시키는 것을 의미한다.

1. R1,...,Rn 중 최소 하나보다 R이 낮은 정규화 수준에 있도록 하는 결합 R로 대체한다.
2. R1,...,Rn 각각의 모든 가능한 값 r1,...,rn에서 Ri의 속성에 대한 R의 해당 값 r을 투영한 결과는 ri와 같다.

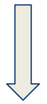
다만, 중복성이 증가한다는 것이 반드시 탈정규화를 의미하지는 않는다.

What is Denormalization?

탈정규화가 아닌것

INO	INAME
1	사과
2	당근
3	노트북

SNO	INO	QTY
gs25	1	5
cu	1	10
cu	2	3
emart24	3	3



INO	INAME	TOTAL_QTY
1	사과	15
2	당근	3
3	노트북	3

➔ 여전히 BCNF를 만족하지만, 중복성이 증가했다

What is Denormalization?

탈정규화는 최후의 수단이어야한다

탈정규화는 때로는 특정 쿼리를 공식화하기 어렵게 만든다.

‘모든 도시의 평균 상태 값’을 가져오는 쿼리를 고려해보자.

SNC

SNO	SNAME	CITY
S1	Smith	London
S2	Jones	Paris
S3	Blake	Paris
S4	Clark	London
S5	Adams	Athens

CT

CITY	STATUS
Athens	30
London	20
Paris	30

S

SNO	SNAME	STATUS	CITY
S1	Smith	20	London
S2	Jones	30	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

```
SELECT AVG ( STATUS ) AS RESULT
FROM CT
```

```
SELECT AVG ( TEMP.STATUS ) AS RESULT
FROM ( SELECT DISTINCT S.CITY , S.STATUS
       FROM S ) AS TEMP
```

➔ DISTINCT에 주목하자.
탈정규화는 중복을 발생시키고 이는 곧 중복제거의 필요를 초래한다.

Surrogate Key

대리 키 vs 튜플 ID

- 우선, 대리 키는 튜플 ID와 같지 않다.
- 대리키는 Entity를 식별하고 튜플 ID는 튜플을 식별한다.
- 튜플 ID는 대개 성능을 함축한다.
- 튜플 ID는 사용자에게 숨겨질 수 있지만, 대리 키는 그럴 수 없다.
- 다시 말하면, 관계 변수 R에 튜플 ID는 포함되지 못한다.
- 즉, 대리 키는 논리적 디자인이고 튜플 ID는 물리적 디자인이다.
- 대리 키가 유용한 지에 대한 논의는 ‘관계형 모델’이 아닌 SQL(디자인 비즈니스) 측면에서 고려할 이야기이다.

Normal Forms
(JD)

4NF

익숙한 정의

BCNF에서 다치 종속(MVD)을 제거하면 4NF이다.

➢ **다치 종속**

같은 테이블 내의 독립적인 두 개 이상의 컬럼이

또 다른 컬럼에 종속되는 것

다치 종속은 정말 특수한 JD이다!

개발자	자격증	언어
홍길동	정보처리기사	C
홍길동	빅데이터 분석기사	C++
장길산	정보보안기사	JAVA

개발자	언어
홍길동	C
홍길동	C++
장길산	JAVA

개발자	자격증
홍길동	정보처리기사
홍길동	빅데이터 분석기사
장길산	정보보안기사

5NF

익숙한 정의

4NF에서 조인 종속을 제거하면 5NF이다.

➢ **조인 종속**

하나의 관계 변수를 여러 개의 관계 변수로 분해하였다가, 다시 조인했을 때 데이터 손실이 없는 것

개발자	자격증	언어
홍길동	정보처리기사	C
홍길동	빅데이터 분석기사	C
홍길동	정보처리기사	C++
홍길동	빅데이터 분석기사	C++
장길산	정보보안기사	JAVA

➔ 앞선 4NF 테이블을 결합한 결과

개발자	자격증
홍길동	정보처리기사
홍길동	빅데이터 분석기사
장길산	정보보안기사

자격증	언어
정보처리기사	C
빅데이터 분석기사	C++
정보보안기사	JAVA

개발자	언어
홍길동	C
홍길동	C++
장길산	JAVA

JD

JD는 FD와 닮아있다

모든 FD는 JD를 의미한다.

그러나 모든 JD가 FD에 의해 내포되는 것은 아니다.

즉, 5NF는 BCNF를 만족하지만 FD에 의해 내포되지 않은 어떠한 JD와 관련이 있다.

기본 아이디어

우리는 지금까지 어떤 관계 변수를 분해할 때, 정확히 두 개의 투영으로 대체해 분해한다고 가정해왔다.

심지어 우리의 목표가 BCNF일 때 그것은 정확히 보증됐다.

코드 박사는 1969년에 이 가능성(세 개 이상의 투영)을 알고 있었고 추후에 발견되어 놀라움을 안겼다!!

JD

정의

X1, …, Xn을 관계 변수 R의 속성 집합이라 하자. 이 때, n진 JD는 R이 X1, …, Xn에 대한 투영으로 무손실 분해될 수 있는 경우에만 R에서 성립한다.

히스의 정리(JD관점으로의 재정의)

- 관계 R의 투영인 X, Y, Z에 대해 관계 R이 FD[X→Y]을 만족하는 경우, JD⋈{XY,YZ}의 제약을 받는다.

JD Practice

S

SNO	SNAME	STATUS	CITY
S1	Smith	20	London
S2	Jones	30	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

SNC

SNO	SNAME	CITY
S1	Smith	London
S2	Jones	Paris
S3	Blake	Paris
S4	Clark	London
S5	Adams	Athens

CT

CITY	STATUS
Athens	30
London	20
Paris	30

SNO	SNAME
S1	Smith
S2	Jones
S3	Blake
S4	Clark
S5	Adams

SNO	CITY
S1	London
S2	Paris
S3	Paris
S4	London
S5	Athens

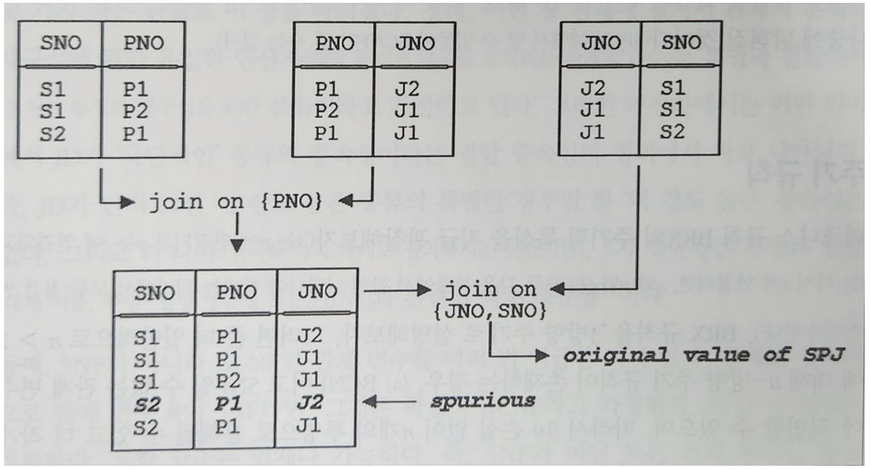
CITY	STATUS
Athens	30
London	20
Paris	30

JD Practice

SNO	PNO	JNO
S1	P1	J2
S1	P2	J1
S2	P1	J1
S1	P1	J1

Key : {SNO, PNO, JNO}

명제 : 공급자 S1이 프로젝트 J1에 부품 P1을 공급한다.



JD⋈{ { SNO, PNO } , { PNO , JNO } , { JNO , SNO } } 가 관계 변수 SPJ에서 분명히 성립한다.

결합 과정

1. 처음 두개의 투영을 결합하면 원래 SPJ 관계 복사본과 하나의 spurious(가짜) 튜플이 생성된다.
2. 다른 투영에 결합되면 spurious 튜플이 사라진다.

5NF

몇 가지 관찰

➢ 5NF는 언제나 가능하다.
즉, 5NF가 아닌 관계 변수는 항상 5NF 투영의 집합으로 분해될 수 있다.

➢ 5NF인 관계 변수 R은 투영으로 제거 가능한 중복성이 없음이 보장된다.
즉, R이 5NF라고 말하는 것은 R의 투영으로 추가적인 분해가 중복성을 제거하지 못할 것이라고 말하는 것이다.

➢ 그러나, R이 5NF라고 하는 것이 중복성에서 자유롭다는 의미는 아니다.
이는 널리 퍼져있는 잘못된 개념이다.
- Q. 관계 변수가 5NF면 중복성이 없는가?
  A. 아니오

5NF

5NF의 중복성

CTXD

CNO	TNO	XNO	DAYS
C1	T1	X1	7
C1	T1	X2	8
C1	T2	X1	9
C1	T2	X2	6

명제 : 교수 TNO가 과정 CNO에서 교과서 XNO를 DAYS동안 가르친다.

키 : {CNO, TNO, XNO}

더 이상의 무손실 분해가 불가능하다!

유용한 정리

R이 BCNF를 만족할 때, R에 복합 키가 존재하지 않는다면 5NF이다

Other Normal Forms

Constraint (informal description in parentheses)	UNF (1970)	1NF (1970)	2NF (1971)	3NF (1971)	EKNF (1982)	BCNF (1974)	4NF (1977)	ETNF (2012)	5NF (1979)	DKNF (1981)	6NF (2003)
Unique rows (no duplicate records)[4]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Scalar columns (columns cannot contain relations or composite values)[5]	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Every non-prime attribute has a full functional dependency on a candidate key (attributes depend on the *complete* primary key)[5]	✗	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓
Every non-trivial functional dependency either begins with a superkey or ends with a prime attribute (attributes depend *only* on the primary key)[5]	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓	✓
Every non-trivial functional dependency either begins with a superkey or ends with an elementary prime attribute (a stricter form of 3NF)	✗	✗	✗	✗	✓	✓	✓	✓	✓	✓	—
Every non-trivial functional dependency begins with a superkey (a stricter form of 3NF)	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓	—
Every non-trivial multivalued dependency begins with a superkey	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓	—
Every join dependency has a superkey component[8]	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓	—
Every join dependency has only superkey components	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	—
Every constraint is a consequence of domain constraints and key constraints	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✗
Every join dependency is trivial	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓

이미지 출처 : https://en.wikipedia.org/wiki/Database_normalization

Other Normal Forms

1NF

2NF

3NF

EKNF, (3,3)NF

BCNF

4NF

ETNF

RFNF, KCNF

SKNF

5NF(PJ/NF)

6NF, **DK/NF**, Overstrong PJ/NF, PJSU/NF

View

What is Redundant?

포괄적인 정의

데이터베이스는 동일한 명제의 두 가지 분명한 표현을 직접적으로 또는 간접적으로 포함하는 경우에만 중복성을 가진다.

즉, 동일한 명제를 나타내는 튜플 혹은 튜플 조합을 갖지 않더라도 중복성을 띌 수 있다.

상세한 정의

D를 데이터베이스 디자인으로 하고, DB를 D를 따르는 데이터베이스 값으로 하며, p를 단순 미양화 명제로 한다. DB가 p의 둘 이상의 구별되는 출현을 포함하는 경우, DB는 중복성을 포함하고 D를 허용한다.

Example

PAYMENTS { CUSTNO , DATE , AMOUNT }

KEY { CUSTNO , DATE }

TATALS { CUSTNO , TOTAL }

KEY { CUSTNO }

CONSTRAINT C12 TOTALS = SUMMARIZE PAYMENTS BY { CUSTNO } :

{ TOTAL := SUM (AMOUNT) } ;

Solution

S1. 순수 디자인(실무)

- 제약 조건은 DBMS에 선언되지 않는다
- 데이터 관리는 사용자의 책임이다.

S2. 제약 조건 선언

- 제약 조건이 DBMS에 선언된다.
- 여전히 사용자의 책임이다

Solution

S3. 뷰 사용

```
VAR TOTALS VIRTUAL
    ( SUMMARIZE PAYMENTS BY { CUSTNO } :
                { TOTAL := SUM ( AMOUNT ) } ) ;
```

이로써 사용자는 더 이상 파생 데이터 관리에 대해 걱정할 필요가 없다.

심지어는 사용자는 TOTALS가 뷰라는 사실을 몰라도 된다!!

단지, 유지 관리 작업이 시스템(뷰)에 의해 실행되며 개입해서는 안된다는 점만 명시하면 된다.

View

정의

a view is the **result set** of a stored query that presents a limited perspective of the database to a user. (...)

as a result set, it is a **virtual table** computed or collated dynamically from data in the database when access to that view is requested.

(출처 : https://en.wikipedia.org/wiki/View_(SQL))

뷰는 파생된 관계 변수이다.

이는, 뷰가 '정규 테이블'임을 뜻한다.

Solution

S4. 스냅샷(m-view) 사용

그러나 뷰의 단점은 뷰를 참조할 때마다 파생 프로세스가 수행된다는 것이다.

```
VAR TOTALS SNAPSHOT …
    ( SUMMARIZE PAYMENTS BY { CUSTNO } :
                    { TOTAL := SUM ( AMOUNT ) } )
    REFRESH ON EVERY UPDATE ;
```

이제 원하는 주기(ON EVERY UPDATE)에 스냅샷이 새로 갱신되어 성능이 개선되었다.

Materialized view

정의

a materialized view is a database **object** that contains the results of a query.

(출처 : https://en.wikipedia.org/wiki/Materialized_view)

m-view 또한 파생된 관계 변수이다.

이는, m-view가 ‘정규 테이블’임을 뜻한다.

Thank you!