

데이터베이스 및 설계

Chap 4. 관계 데이터베이스 (#1/2)



2012.03.26.

오 병 우

컴퓨터공학과

관계 데이터 모델

● 관계 데이터 모델(relational data model)의 탄생

◆ 1970년대 IBM의 E. F. Codd에 의해 제안

- Data structure
- Data manipulation (operation)
- Data integrity (constrains)

● 관계 데이터 모델의 특성

◆ 릴레이션(relation)과 수학적 이론에 기초

◆ 일반 사용자는 테이블 형태로 생각

- 통상적인 테이블의 개념과는 다름
 - 관계 데이터 모델의 직관적인 이해에는 도움이 됨
- 테이블의 열(column) = 필드(field) 혹은 아이템(item)
 ≍ 관계 데이터 모델의 애트리뷰트(attribute)
- 테이블의 행(row) = 레코드(record)
 ≍ 관계 데이터 모델의 투플(tuple)

Relations

Table

- ◆ A concrete representation of an abstract object (i.e, relation)
- ◆ Easy to use and easy to understand

구체적인

Distinct attribute name

- ◆ Several attributes from the same domain
- ◆ Distinct role name common domain name
 - Attribute 이름은 중복되면 안됨

별개의

관계 데이터 모델 예제

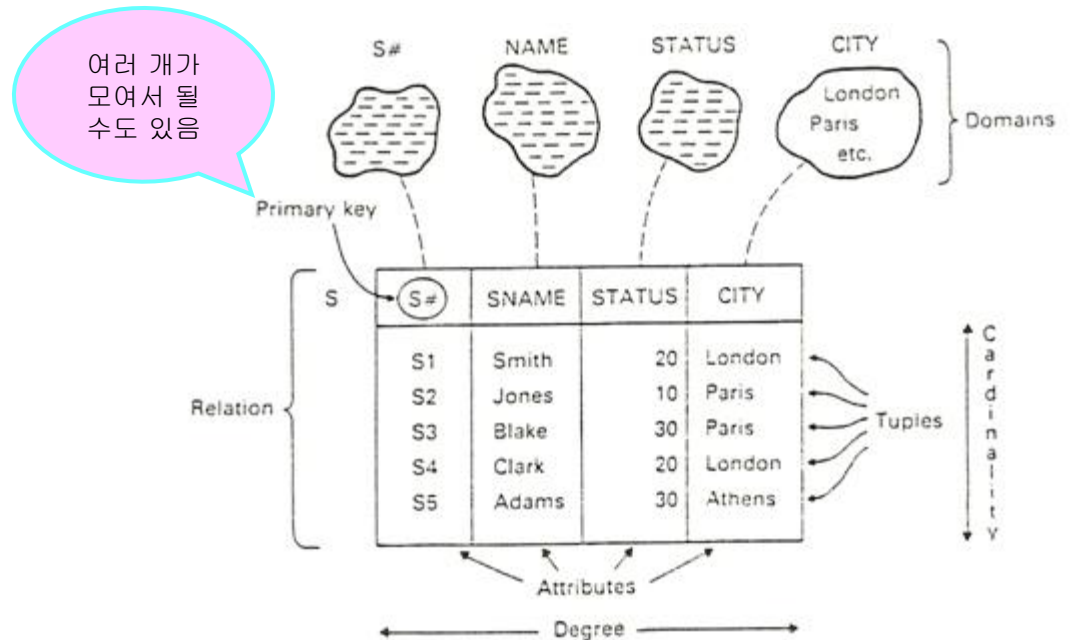
학생(STUDENT) 테이블 : 릴레이션

학생
(STUDENT)

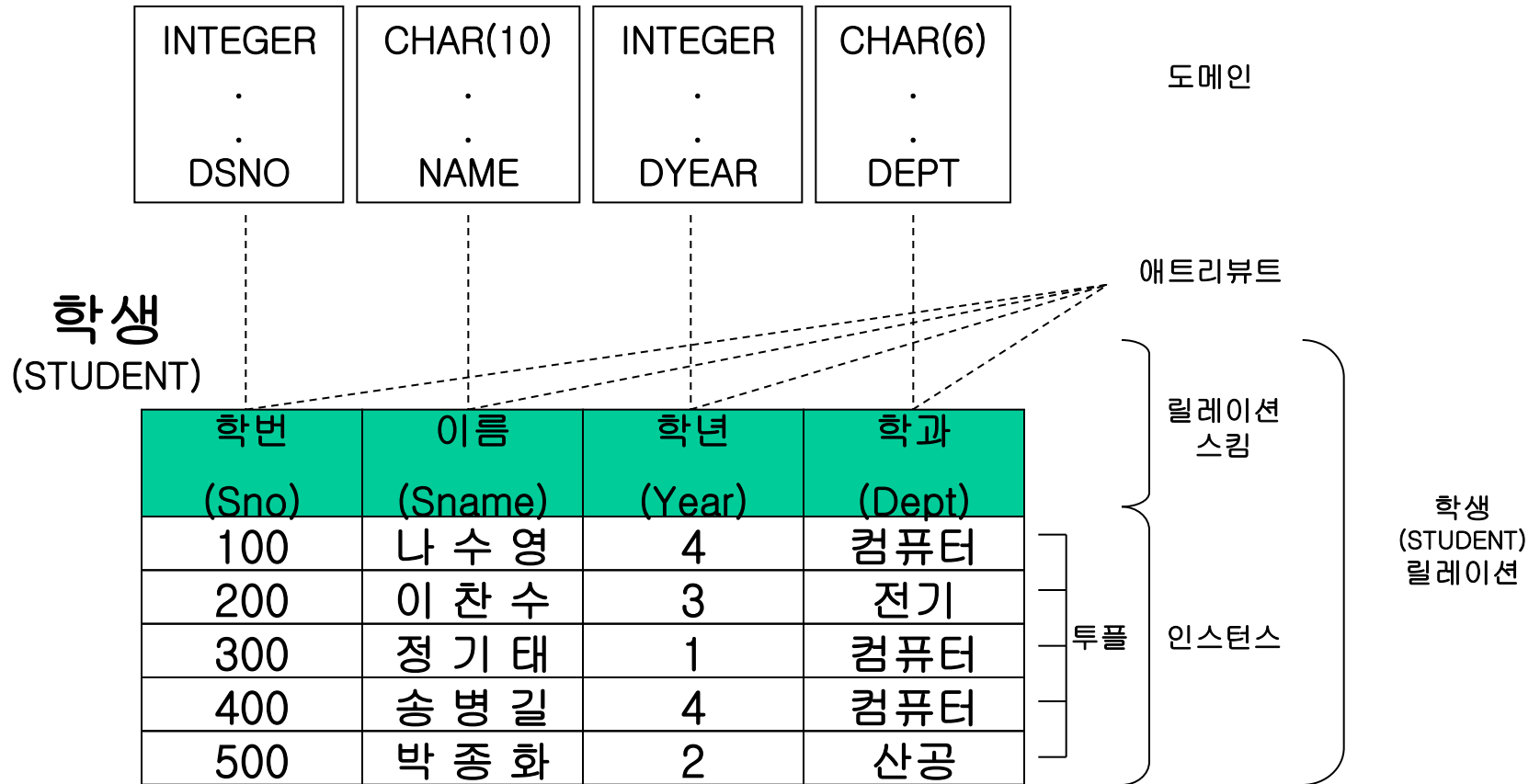
학번 (Sno)	이름 (Sname)	학년 (Year)	학과 (Dept)
100	나수영	4	컴퓨터
200	이찬수	3	전기
300	정기태	1	컴퓨터
400	송병길	4	컴퓨터
500	박종화	2	산공

Formal Relational Terms

- Relation
 - ◆ table
- Tuple
 - ◆ row of a table
- Attribute
 - ◆ column of a table
- Cardinality
 - ◆ number of tuples
- Degree
 - ◆ number of attributes
- Primary key
 - ◆ unique identifier for the table
- Domain
 - ◆ a pool of values for an attribute



릴레이션의 예제



Domains

Atomic (가장 작은 단위)

- ◆ The smallest semantic unit of data (\equiv scalar value)
- ◆ Non decomposable (\rightarrow lose meaning)

애트리뷰트 이름과 도메인 이름은 같을 수도 있음

의미의
분해할 수 있는

Domain (\supseteq attribute)

- ◆ A named set of scalar values, all of the same type
- ◆ 애트리뷰트가 취할 수 있는 값(value)들의 집합
- ◆ A pool of values (the actual values appearing in attributes)
- ◆ 1. simple domains \Rightarrow simple attributes

– Domains of scalar values

– \rightarrow 단순 애트리뷰트 : 원자값

- ◆ 2. composite domains \Rightarrow composite attributes

– A combination of simple domains

\rightarrow 복합 애트리뷰트 : 복합값

연, 월, 일 \Rightarrow 날짜: <연,월,일>

– Attribute 값을 의미상 하나의 단위로 취급하면 문제 없음

값의 집합이 되면
Unnormalized relation
(비정규화 릴레이션)

값의 집합은 허용하지 않음

Domains

Significance of domains

◆ Domains contain comparisons

◆ (ex)

```
SELECT P.*, SP.*
FROM P, SP
WHERE P.P# = SP.P# ;
```

defined on the same domain
make sense

```
SELECT P.*, SP.*
FROM P, SP
WHERE P.WEIGHT = SP.QTY
```

defined on different domains
do not make sense

◆ Prevent users from making silly mistakes → inform the user

Attribute

Attribute name (vs. domain name)

- ◆ 해당 도메인의 이름을 그대로 사용할 수도 있고, 별도의 이름을 사용할 수도 있음
- ◆ 임의의 도메인이 릴레이션에서 담당하고 있는 역할의 이름으로 지정 (SNAME)

```
CREATE DOMAIN NAME CHAR(20);
CREATE TABLE S
    (S#          CHAR(5)          NOT NULL,
     SNAME      DOMAIN (NAME)    NOT NULL,
     CITY       CHAR(15)         NOT NULL,
     PRIMARY KEY (S#) );
```

Composite domain

- ◆ A combination of simple domains

```
CREATE DOMAIN MONTH CHAR(2);
CREATE DOMAIN DAY CHAR(2);
CREATE DOMAIN YEAR CHAR(2);
CREATE DOMAIN DATE (MONTH DOMAIN (MONTH),
                   DAY DOMAIN (DAY),
                   YEAR DOMAIN (YEAR));
```

도메인 예제

● 릴레이션 학생(STUDENT)의 정의

```

DCL DOMAIN DSNO INTEGER;
DCL DOMAIN NAME CHAR(10);
DCL DOMAIN DYEAR INTEGER;
DCL DOMAIN DEPT CHAR(6);
DCL RELATION STUDENT
(Sno      DOMAIN DSNO,
 Sname    DOMAIN NAME,
 Year      DOMAIN DYEAR,
 Dept     DOMAIN DEPT);

```

● 도메인 명세가 생략된 릴레이션 STUDENT의 정의

```

DCL RELATION STUDENT
(Sno      INTEGER,
 Sname    CHAR(10),
 Year      INTEGER,
 Dept     CHAR(6));

```

Relations

- A relation (R) on domains D_1, D_2, \dots, D_n ($n = \text{degree}$)
 - ◆ Heading : (relation) schema, scheme, intension
 - A fixed set of attribute – domain pairs

$$\{(A_1:D_1), (A_2:D_2), \dots, (A_n : D_n) \}$$
 where A_j 's must all be distinct
 - ◆ Body : (relation) instance, state, extension
 - A time-varying set of tuples, where
 - each tuple : a set of attribute–value pairs

$$\{(A_1:V_{i1}), (A_2:V_{i2}), \dots, (A_n:V_{in}) \} \ i = 1, \dots, m \ (m = \text{cardinality})$$

릴레이션 스키 (Relation Scheme)

- 릴레이션 내포 (relation intension) 또는 릴레이션 스키마라고도 함

릴레이션 이름 + 애트리뷰트 이름

$$R(A_1, A_2, \dots, A_n), \quad A_i \Leftrightarrow D_i$$

$$R(\{A_1, A_2, \dots, A_n\})$$

- 정적 성질

- ◆ 시간에 무관

- ◆ 릴레이션 타입

릴레이션 인스턴스 (Relation Instance)

- 릴레이션 외연 (relation extension)이라고도 함
- 릴레이션 R 의 릴레이션 인스턴스
 - ◆ 어느 한 시점에 릴레이션 R 이 포함하고 있는 튜플들의 집합

$$\{ \langle V_1, V_2, \dots, V_n \rangle \mid V_i \in D_i \}$$
 - ◆ 릴레이션의 내용, 상태
- $\{ (\text{attr}_1=V_1, \text{attr}_2=V_2, \dots, \text{attr}_n=V_n) \}$
- 동적 성질
 - ◆ 삽입, 삭제, 갱신으로 시간에 따라 변함
 - ◆ 릴레이션 값(보통 릴레이션)

릴레이션 (Relation) R

i. 수학적 정의

릴레이션 R : 카티션 프로덕트(Cartesian product)의 부분집합

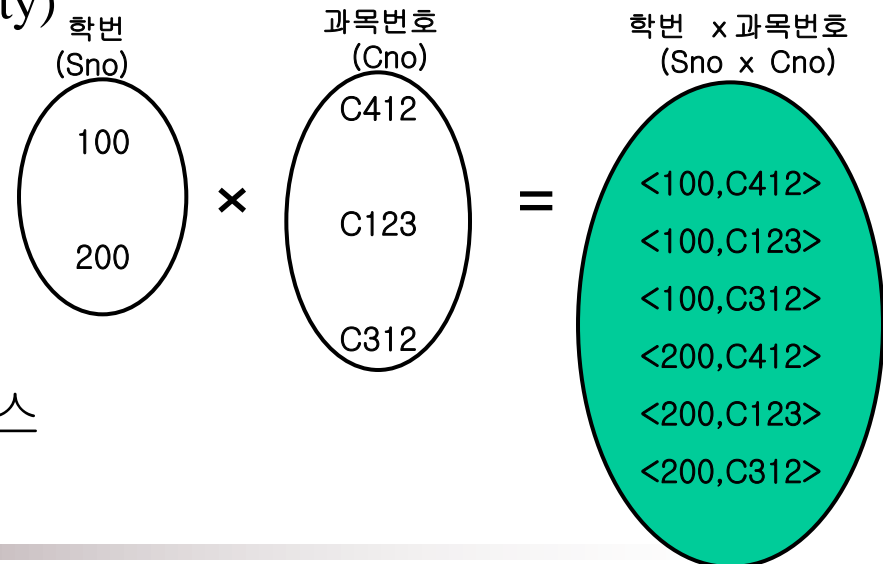
$$R \subseteq D_1 \times D_2 \times \dots \times D_n$$

즉 n-튜플 $\langle d_1, d_2, \dots, d_n \rangle$ 의 집합

단 D_i : i번째 도메인 $d_i \in D_i, i = 1, 2, \dots, n$

n : R의 차수(degree : 일차, 이차, 삼차, ..., n차)

튜플의 수 : 카디널리티(cardinality)



ii. 개념적 정의

릴레이션 스키 + 릴레이션 인스턴스

Relations

중요

● Properties of relations

- ◆ There are no duplicate tuples
 - Relation body is a mathematical set
 - There is always a primary key
- ◆ Tuples are unordered (top to bottom)
 - Relation body is a mathematical set
- ◆ Attributes are unordered (left to right)
 - Relation heading is defined as a set
 - Attributes : always referenced by name, never by position
- ◆ All attribute values are atomic
 - All underlying domains are simple in turn
 - Relations do not contain repeating groups (normalized relation)

릴레이션의 특성 (1)

i. 튜플의 유일성

릴레이션 = 서로 다른 튜플들의 "집합"

ii. 튜플의 무순서성

릴레이션 : 추상적 개념 튜플들의 집합

테이블 : 구체적 개념

iii. 애트리뷰트의 무순서성

릴레이션 스키마 → 애트리뷰트들의 "집합"

튜플 : <attr:value>쌍의 집합

릴레이션의 특성 (2)

iv. 애트리뷰트의 원자성

- ◆ 애트리뷰트 값 = 원자 값(atomic value)
 - 논리적으로 분해 불가능
- ◆ 정규화 릴레이션 (normalized relation)
 - 비정규화 릴레이션은 분해로 정규화
 - 동등한 의미 유지
- ◆ 널 값(null value) = 원자 값
 - unknown, inapplicable
- ◆ 도메인
 - 단순 도메인
 - 복합 도메인 : 값을 하나의 단위로 취급

릴레이션의 Normalization

등록1
(ENROL1)

Degree
: 2

학번 (Sno)	과목성적 (Cgrade)	
	과목번호 (Cno)	성적 (Grade)
100	C413	A
	E412	A
200	C123	B
300	C312	A
	C324	C
	C413	A
400	C312	A
	C324	A
	C413	B
	E412	C
500	C312	B

(a) 비정규 릴레이션

등록
(ENROL)

Degree
: 3

학번 (Sno)	과목번호 (Cno)	성적 (Grade)
100	C413	A
100	E412	A
200	C123	B
300	C312	A
300	C324	C
300	C413	A
400	C312	A
400	C324	A
400	C413	B
400	E412	C
500	C312	B

(b) 정규 릴레이션

Relations

● Kinds of relations

- ◆ 1. base relations (real relations)
 - A named, autonomous relation (direct part of the database)
- ◆ 2. view (virtual relations)
 - A named, derived relation (purely by its definition)
- ◆ 3. snapshots
 - A named, derived relation
 - Real, not virtual

```
CREATE SNAPSHOT SC
      AS SELECT S#, CITY
          FROM S
      REFRESH EVERY DAY;
```

Relations (cont'd)

● Kinds of relations (cont'd)

◆ 4. query results

- The final output relation resulting from a query
- No persistent existence within the database

◆ 5. intermediate results

- Results from some relational expression that is nested within some larger
- No persistent existence within the database

```
SELECT DISTINCT S.CITY
FROM S
WHERE S.S# IN (SELECT SP.S#
               FROM SP
               WHERE SP.P# = 'P2');
```

◆ 6. temporary relations

- A named relation that is automatically destroyed at some appropriate time

Relational Databases

Relational database

~으로 여기다

- ◆ A database that is perceived by the user as a collection of time-varying, normalized relations of assorted degrees
- ◆ Correspondences to file system
 - Relation : file
 - Tuple : record (occurrence)
 - Attribute : field (type)
- ◆ Major features of relational files (vs. traditional files)
 - 1. each “file” contains only one record type
 - 2. the fields have no particular order, left to right
 - 3. the records have no particular order, top to bottom
 - 4. every field is single-valued
 - 5. the records have a unique identifier field or field combination called the primary key

여러가지

관계 데이터베이스

- 관계 데이터베이스

- ◆ 데이터베이스를 시간에 따라 그 내용(상태)이 변할 수 있는 테이블 형태로 표현

- 관계 데이터베이스 스키마

- = {릴레이션 스키마} + 무결성 제약조건

- 관계 데이터 모델 \Leftrightarrow 프로그래밍 시스템

- 릴레이션 \Leftrightarrow 파일

- 튜플 \Leftrightarrow 레코드 (레코드 어커런스)

- 애트리뷰트 \Leftrightarrow 필드(필드 타입)

- ☞ Notes

- ◆ 관계 데이터베이스는 데이터가 꼭 물리적 테이블 형태로 저장되는 것을 의미하지는 않음

Example

● 대학(University) 관계 데이터베이스

학생 (STUDENT)	학번 (Sno)	이름 (Sname)	학년 (Year)	학과 (Dept)
	100	나 수 영	4	컴퓨터
200	이 찬 수	3	전기	
300	정 기 태	1	컴퓨터	
400	송 병 길	4	컴퓨터	
500	박 종 화	2	산공	

과목 (COURSE)	과목번호 (Cno)	과목이름 (Cname)	학점 (Credit)	학과 (Dept)	담당교수 (PRname)
	C123	프로그래밍	3	컴퓨터	김성국
C312	자료 구조	3	컴퓨터	황수관	
C324	화일 구조	3	컴퓨터	이규찬	
C413	데이터베이스	3	컴퓨터	이일로	
E412	반 도 체	3	전자	홍봉진	

Example (cont'd)

- 대학(University) 관계 데이터베이스(cont'd)

등록
(ENROL)

학번 (Sno)	과목번호 (Cno)	성적 (Grade)	중간성적 (Midterm)	기말성적 (Final)
100	C413	A	90	95
100	E412	A	95	95
200	C123	B	85	80
300	C312	A	90	95
300	C324	C	75	75
300	C413	A	95	90
400	C312	A	90	95
400	C324	A	95	90
400	C413	B	80	85
400	E412	C	65	75
500	C312	B	85	80