**Chapter 2: Relational Model**

**- 데이터 모델: 데이터와 그 관계를 표현하는 프레임워크/formalism**

1. 가장 간단한, 레코드를 Linked List 형태로 표현.

2. Instructor, Student가 Advise라는 관계로 이어져 있음. E-R모델.

3. Key-Value 모델.

4. 엑셀 파일. Row와 Column이 있음.

데이터 모델 – Relational: 엑셀 파일과 비슷한 구조. 데이터베이스로 구현.

Row <-> tuple/record, column <-> attribute/field

**- Relation (in Set Theory)**

relation인 이유? set이기 때문.

집합 A에서 Binary Relation R은 A의 원소들의 순서 쌍의 집합.

Ex) A = {1, 2, 3}이면 < = {<1,2>, <1,3>, <2,3>}, <=이면 <에다가 <1,1>, <2,2>, <3,3> 추가.

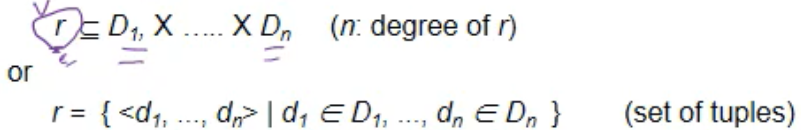
R 은 A X A의 Subset이다.

Relation R2 between elements of set B and set C

Ex) B = {미국, 일본, 중국, 한국}, C = {베이징, 서울, 워싱턴, 도쿄} 이면 수도 = {<미국, 워싱턴>, <일본, 도쿄>, <중국, 베이징>, <한국, 서울>}.

**- Basic Structure of a Relation**

Formally, 주어진 집합들 D1, D2, …, Dn에서 relation r은 D1 X D2 X … X Dn의 subset임.



하단 예시는 pdf 참조.

**- Attribute Types**

Relation의 각각의 attribute는 이름을 가지고 있음.

각각 attribute가 취할 수 있는 값의 집합을 **domain**이라고 함.

**Attribute value들은 (일반적으로) atomic 해야 함.**

- 즉, indivisible 함. **뒤에 나오는 1NF를 지키면 atomic.**

Multivalued attribute, composite attribute는 non-atomic.

Atomic은 각 데이터베이스에서 가장 작은 단위가 된다는 것을 의미함. 그래서 모든 DB에서 절대적인 건 아님.

Null 값은 많은 연산의 정의에 복잡함을 유발함.

- 메인 프레젠테이션에선 null 값의 효과를 무시할 것이고 나중에 고려하자.

**- Relation Schema**

Relation이 어떤 attribute로 구성되는지 나타낸 것.

A1, A2, …, An은 attributes.

R = (A1, A2, …, An) 은 Relation Schema.

**예를 들어, Customer-schema = (customer-name, customer-street, customer-city)**

r(R)은 relation schema R의 relation (variable)임.

Ex) i: int라고 하면, I = 0, I = i+1이라고 하면 instance는 계속 변하지만 이 instance를 담는 그릇을 relation이라고 함.

**Ex) customer(Customer-schema) 로 나타낼 수 있음.**

**- Relation Instance**

Relation의 현재 값들(relation instance)들은 테이블에 의해 특정됨.

**R의 Element t는 tuple**인데, 테이블에서 row로 표현됨.

**- Relations are Unordered**

tuple들의 순서는 서로 관계 없음. **tuple들은 무작위한 순서로 저장될 수 있음.**

또한 중복도 의미가 없음. 왜냐? Set 이기 때문에 기본적으로 중복이 없음.

실제 운용에서는 order를 써야하는 상황이 있을 수도 있음.

**- Relational Database**

위에서 설명한 것은 각각 relation들의 schema를 얘기한 것.

**Relational database는 많은 relation로 이루어진 데이터베이스.**

모든 정보를 하나의 relation에 저장하는 것은 좋은 방식이 아님.

**Database Design (Chapter 7 & 8)**에서 relational schema를 어떻게 정할지를 다룬다.

**- A relational database for a university**

Pdf참조.

**- Keys**

K가 R의 subset이라고 하자. **만약 K의 값들이 각각의 possible relation인 r(R)의 유일한 tuple을 식별할 수 있다면 K는 superkey이다.** 만약 **K가 minimal하면** **K는 candidate key**이다. 이 때 possible r이라는 것은 우리가 모델링하고 있는 enterprise의 relation r이라고 함. 예시는 pdf 참조.

보통 key라고 하면 candidate key를 의미함.

**- Keys**

Primary key: **candidate key 중의 하나**가 primary key로 선택된다.

Foreign key: 한 relation의 값 중 다른 것에 무조건 나타나야 하는 것.

- Referencing relation, Referenced relation이 있음.

**- Schema Diagram for a Banking Enterprise / The University**

Pdf 참조. 화살표들은foreign key 참조를 나타냄. 화살표가 가리키는 곳은 Primary key, 시작하는 곳은 그 키를 참조하는 foreign key.

University에서는 밑줄 친 애들이 primary key고 banking은 구역이 나눠져 있는 곳이 primary key.

**- Query Languages**

데이터베이스의 정보를 retrieve 하는 언어. (DML의 일부, retrieval을 query라고 함. 하지만 DML+DDL 전체를 query라고 하기도 함.)

**Procedural – Specify what data are needed and how to get those data.**

절차적 언어. 데이터를 얻는 순서 같은 것도 지정해줌. Ex) Relational Algebra

**Nonprocedural – Declarative. Specify only what data are needed**

Ex) SQL

Pure Languages: Relational Algebra, Tuple Relational Calculus, Domain Relational Calculus.

얘네는 사용되고 있는 쿼리 언어들의 underlying basis를 형성함.

**- Relational Algebra**

Algebra == 대수. Calculus == 해석학.

이건 순서가 있음. 순서 중요!

**Algebra는 Operator와 Operand에 대한 학문.** Relational Algebra에서 Operand는 Relation이고 Operator는 basic operator, relation들을 받아 새로운 결과를 내놓음.

Relational Algebra는 **Procedural language임.**

자주 쓰이는 operator는 selection, projection, natural join, cartesian product, union 등이 있음.

그 뒤 operation들은 pdf 참고.

Selection은 record 필터, project는 column 필터. Union은 합집합 (relation이 set이기 때문에 자연스러움). Set difference는 차집합.

Cartesian-Product는 set A와 B의 모든 원소의 조합을 뽑는 것. 여기선 Relation r, s의 모든 조합.

Natural-Join은 relation r, s에서 공유하고 있는 attribute가 있으면 그 attribute들의 값이 일치하는 값들의 조합만 남김. Cartesian Product는 공유하고 있는 attribute가 있으면 중복된 값이 나오기 때문에 이렇게 됨.

연산자들을 합칠 수 있음. (Composition). 한 연산의 결과를 다른 연산에다가 쓸 수 있다는 의미.

 <- Natural Join도 이런 식으로 나타낼 수 있음. Cartesian Product 후 공유 attribute가 같은 값을 selection 한 뒤에 중복 값은 project.