Chapter 2

Introduction to the Relational Model

Slides by Silbershatz, Modifications by Rogers and Brown

Example of a Relation

Attributes (or columns)

Tuples (or rows)

ID	Name	Dept_Name	Salary
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
14061	Mozart	Music	59000
42305	Buchholz	Computer Science	78000
49675	Hemmingway	English	73000
85720	Crick	Biology	72000

Attribute Types

- The set of allowed values for each attribute is called the **domain** of the attribute
- Attribute values are (normally) required to be **atomic**, that is, indivisible
- The special value, **Null**, is a member of every domain
  - The null value causes complications in the definition of many operations

Relation Schema and Instance

- A_1, A_2, A_3, …, A_n are **attributes**
- R = (A_1, A_2, A_3, …, A_n) is a **relation schema**
  - Example:
    Instructor = (ID, Name, Dept_Name, Salary)
- Formally, given sets D_1, D_2, D_3, …, D_n, a relation r is a subset of D_1 X D_2 X D_3 X … X D_n
  - Thus, a **relation** is a set of n-tuples (A_1, A_2, A_3, …, A_n) where each $A_i \in D_i$

Relation Schema and Instance

- The current values (**relation instance**) of a relation are specified by a **table**
- The element t, of r is a tuple and is represented by a row in a table

Relations are Unordered

- Order of tuples is irrelevant
  - Tuples may be stored in any arbitrary order
- Ex. The instructor relation with unordered tuples

ID	Name	Dept_Name	Salary
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
14061	Mozart	Music	59000
42305	Buchholz	Computer Science	78000
49675	Hemmingway	English	73000
85720	Crick	Biology	72000

Database

- A database consists of multiple relations
- Information about an enterprise is broken up into parts
  - For a university those parts might include:
    - Instructors
    - Students
    - Advisors
    - Classes
    - Buildings
    - …

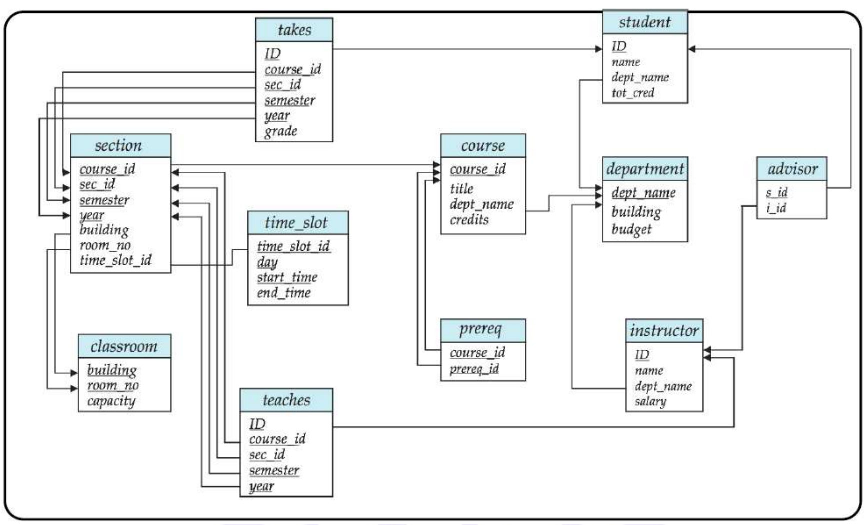
Database

- Bad Design
  - Storing everything in one record
  - Univ (Instructor_ID, Name, Dept_name, Salary, Student_ID, Building, …)
  - Results in
    - Repetition of information (e.g. two students have the same instructor)
    - The need for null values (e.g. a student with no advisor)
- Normalization Theory (Chapter 7)
  - Deals with how to design “good” relational schemas

Keys

- Let $K \subseteq R$
  - K is a **Superkey** of R if values for K are sufficient to identify a unique tuple of each possible relation r(R)
    - Example: {ID} and {ID, Name} are both Superkeys of *Instructor*
  - Superkey *K* is a **Candidate Key** if K is minimal
    - Example: {ID} is a candidate key for *Instructor*
  - One of the candidate Keys is selected to be the primary key – which one?
- Foreign Key constraint
  - Value in one relation must appear in the other
  - Referencing relation
  - Referenced relation

Schema Diagram for University Database



Relational Query Language

- Procedural (imperative) v Non-Procedural (declarative)
  - Imperative – C, Python, Java
  - Declarative – SQL, R, Ruby, Haskell
- "Pure" Languages
  - Relational Algebra
  - Tuple Relational Calculus
  - Domain Relational Calculus
- Relational Operators
  - Select
  - Project
  - Join
  - …

Selection of Tuples

- Relation, r

A	B	C	D
α	α	1	7
α	β	5	7
β	β	12	3
β	β	23	10

- Select tuples with A = B and D > 5
  - σ A = B and D > 5 (r)

A	B	C	D
α	α	1	7
β	β	23	10

Selection of Attributes

- Relation, r

A	B	C
α	10	1
α	20	1
β	30	1
β	40	2

- Select A and C
  - Projection
  - $\pi_{a,c}$ (r)

A	C
α	1
α	1
β	1
β	2

=

A	C
α	1
β	1
β	2

Joining Two Relations – Cartesian Product

- Two relations, R and S

A	B
α	1
β	2

R

C	D	E
α	10	A
β	10	A
β	20	B
ϒ	10	B

S

- R X S

A	B	C	D	E
α	1	α	10	A
α	1	β	10	A
α	1	β	20	B
α	1	ϒ	10	B
β	2	α	10	A
β	2	β	10	A
β	2	β	20	B
β	2	ϒ	10	B

Set Intersection of Two Relations

- Two relations, R and S

A	B
α	1
α	2
β	1

R

A	B
α	2
β	3

S

- R ∩ S

A	B
α	2

Joining Two Relations – Natural Join

- Let *r* and *s* be relations on schemas R and S respectively. Then the "**natural join**" of relations R and S is a relation on schema R ∪ S obtained as follows:
  - Consider each pair of tuples t_r from R and t_s from S
  - If t_r and t_s have the same value on each of the attributes in R ∩ S, add a tuple t to the result, where
    - t has the same value as t_r on r
    - t has the same value as t_s on s

Joining Two Relations – Natural Join

- Two relations, R and S

A	B	C	D
α	1	α	A
β	2	ϒ	A
ϒ	4	β	B
α	1	ϒ	A
δ	2	β	B

R

B	D	E
1	A	α
3	A	β
1	A	ϒ
2	B	δ
3	B	ε

S

- Natural Join – R ⊠ S

A	B	C	D	E
α	1	α	A	α
α	1	α	A	ϒ
α	1	ϒ	A	α
α	1	ϒ	A	ϒ
δ	2	β	B	δ

Figure in 2.1

Symbol (Name	Example of Use
σ Selection	σ salary >= 85000(Instructor) Return rows of the input relation that satisfy the predicate
⊓ Projection	$\sqcap_{ID,\ Salary}$(Instructor) Output specified attributes from all rows of the input relation. Remove duplicate tuples from the output
⊠ Natural Join	Instructor ⊠ Department Output pairs of rows from the two input relations that have the same value on all attributes that have the same name
X Cartesian Product	Instructor X Department Output all pairs of rows from the two input relations (regardless of whether or not they have the same values on common attributes
∪ Union	$\sqcap_{Name}$(Instructor) ∪ $\sqcap_{Name}$(Instructor) Output the union of tuples from the two input relations

End of Chapter 2