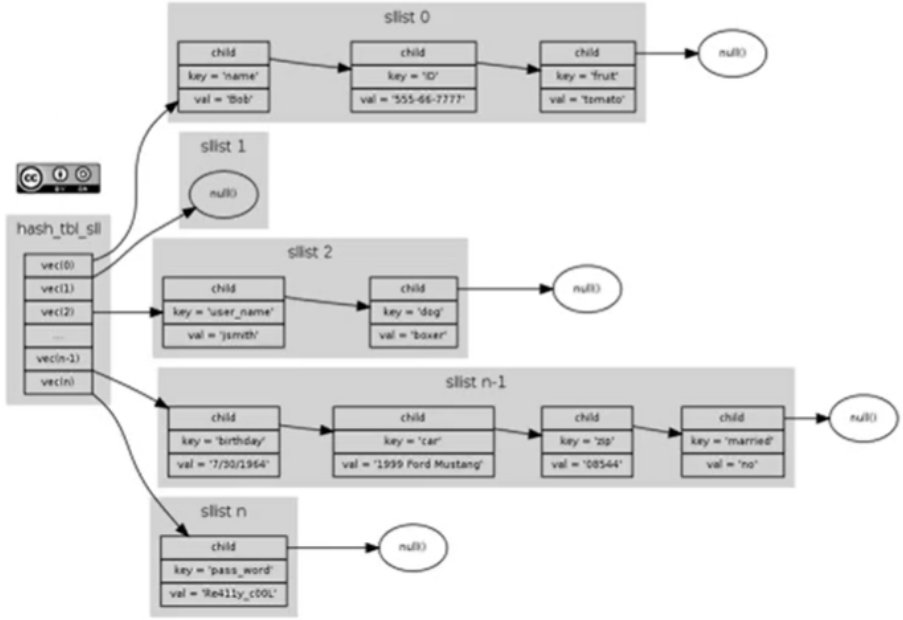
Chapter 2: Relational Model

- Structure of Relational Databases
- Database Schema
- Keys
- Schema Diagrams
- Relational Query Languages
- Relational Operations

Data Model

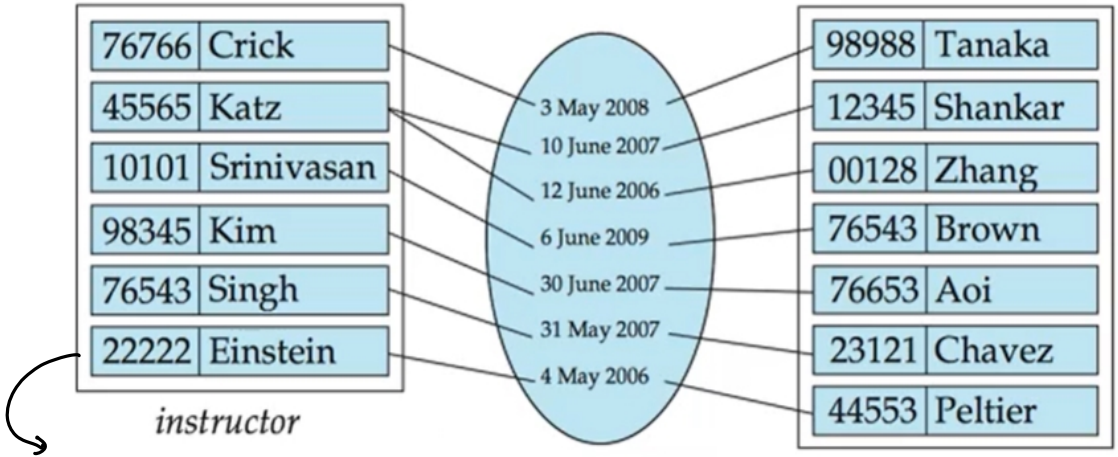
- The framework/formalism for representing data and their relationships



Data Model

Entity – Relationship

instructor가 advise함

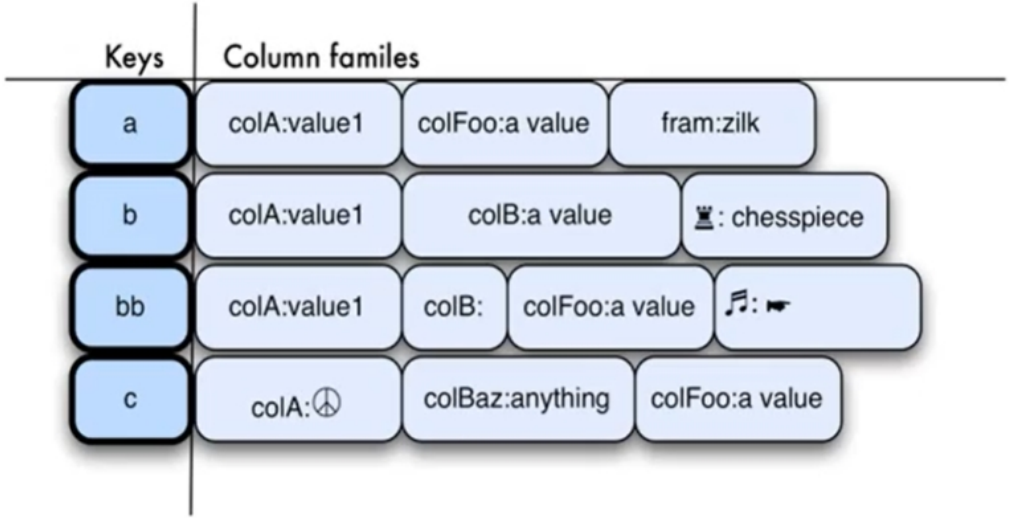


각 entity가 모여

instructor entity set 형성

Data Model

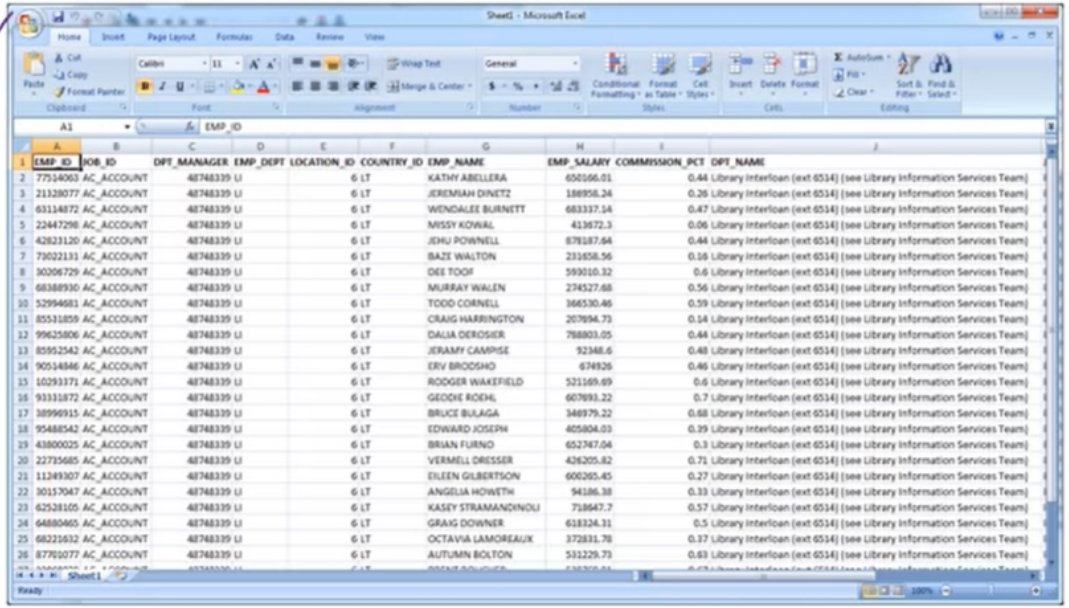
Key-Value Store : 유연한 Schema
→ big data 관리에 유용



Data Model

Excel : raw - column 으로 구성된 table형 DB

(여러보다 계산을 위한 tool)



	A	B	C	D	E	F	G	H	I	J
1	EMP_ID	JOB_ID	DPT_MANAGER	EMP_DEPT	LOCATION_ID	COUNTRY_ID	EMP_NAME	EMP_SALARY	COMMISSION_PCT	DPT_NAME
2	77514063	AC_ACCOUNT	48748339	LI		6	LT	KATHY ABELLERA	650166.01	0.44 Library Interloan (ext 6514) (see Library Information Services Team)
3	21328077	AC_ACCOUNT	48748339	LI		6	LT	JEREMIAH DINETZ	186958.24	0.26 Library Interloan (ext 6514) (see Library Information Services Team)
4	63114872	AC_ACCOUNT	48748339	LI		6	LT	WENDALEE BURNETT	683337.14	0.47 Library Interloan (ext 6514) (see Library Information Services Team)
5	22447298	AC_ACCOUNT	48748339	LI		6	LT	MISSY KOWAL	413672.3	0.06 Library Interloan (ext 6514) (see Library Information Services Team)
6	42823120	AC_ACCOUNT	48748339	LI		6	LT	JEHU POWNELL	878187.64	0.44 Library Interloan (ext 6514) (see Library Information Services Team)
7	73022131	AC_ACCOUNT	48748339	LI		6	LT	BAZE WALTON	231658.56	0.16 Library Interloan (ext 6514) (see Library Information Services Team)
8	30206729	AC_ACCOUNT	48748339	LI		6	LT	DEE TOOF	593010.32	0.6 Library Interloan (ext 6514) (see Library Information Services Team)
9	68388930	AC_ACCOUNT	48748339	LI		6	LT	MURRAY WALEN	274527.68	0.56 Library Interloan (ext 6514) (see Library Information Services Team)
10	52994681	AC_ACCOUNT	48748339	LI		6	LT	TODD CORNELL	366530.46	0.59 Library Interloan (ext 6514) (see Library Information Services Team)
11	85531859	AC_ACCOUNT	48748339	LI		6	LT	CRAIG HARRINGTON	207694.73	0.14 Library Interloan (ext 6514) (see Library Information Services Team)
12	99625806	AC_ACCOUNT	48748339	LI		6	LT	DALIA DEROSIER	788803.05	0.44 Library Interloan (ext 6514) (see Library Information Services Team)
13	85952542	AC_ACCOUNT	48748339	LI		6	LT	JERAMY CAMPISE	92348.6	0.48 Library Interloan (ext 6514) (see Library Information Services Team)
14	90514846	AC_ACCOUNT	48748339	LI		6	LT	ERV BRODSHO	674926	0.46 Library Interloan (ext 6514) (see Library Information Services Team)
15	10293171	AC_ACCOUNT	48748339	LI		6	LT	RODGER WAKEFIELD	521169.69	0.6 Library Interloan (ext 6514) (see Library Information Services Team)
16	93331872	AC_ACCOUNT	48748339	LI		6	LT	GEODIE ROEHL	607693.22	0.7 Library Interloan (ext 6514) (see Library Information Services Team)
17	38996915	AC_ACCOUNT	48748339	LI		6	LT	BRUCE BULAGA	346979.22	0.68 Library Interloan (ext 6514) (see Library Information Services Team)
18	95488542	AC_ACCOUNT	48748339	LI		6	LT	EDWARD JOSEPH	405804.03	0.39 Library Interloan (ext 6514) (see Library Information Services Team)
19	43800025	AC_ACCOUNT	48748339	LI		6	LT	BRIAN FURNO	652747.04	0.3 Library Interloan (ext 6514) (see Library Information Services Team)
20	22735685	AC_ACCOUNT	48748339	LI		6	LT	VERMELL DRESSER	426205.82	0.71 Library Interloan (ext 6514) (see Library Information Services Team)
21	11249307	AC_ACCOUNT	48748339	LI		6	LT	EILEEN GILBERTSON	600265.45	0.27 Library Interloan (ext 6514) (see Library Information Services Team)
22	30157047	AC_ACCOUNT	48748339	LI		6	LT	ANGELIA HOWETH	94186.38	0.33 Library Interloan (ext 6514) (see Library Information Services Team)
23	62528105	AC_ACCOUNT	48748339	LI		6	LT	KASEY STRAMANDINOLI	718647.7	0.57 Library Interloan (ext 6514) (see Library Information Services Team)
24	64880465	AC_ACCOUNT	48748339	LI		6	LT	GRAIG DOWNER	618324.31	0.5 Library Interloan (ext 6514) (see Library Information Services Team)
25	68221632	AC_ACCOUNT	48748339	LI		6	LT	OCTAVIA LAMOREAUX	372831.78	0.37 Library Interloan (ext 6514) (see Library Information Services Team)
26	87701077	AC_ACCOUNT	48748339	LI		6	LT	AUTUMN BOLTON	531229.73	0.63 Library Interloan (ext 6514) (see Library Information Services Team)

Data Model - Relational

관계형 DB

attributes (or columns) or field

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

tuples (or rows) or record

Relation (in Set Theory)

- **Binary relation** *R* on a set *A* 원소들 간의 관계
  is a collection of ordered pairs of elements of *A*
  - *A* = { 1, 2, 3 }
  - < = { <1, 2>, <1, 3>, <2, 3> }
  - ≤ = { <1, 2>, <1, 3>, <2, 3>, <1, 1>, <2, 2>, <3, 3> }

- $R \subseteq A \times A$ A로 만들수 있는 모든 조합 (Cartesian product)

- *(Bipartite) Relation R2* between elements of set *B* and set *C*
  - *B* = {미국, 일본, 중국, 한국}, *C* = {베이징, 서울, 워싱턴, 토쿄}
  - 수도 = { < 미국, 워싱턴>, <일본, 토쿄>, <중국, 베이징>, <한국, 서울> }

↳ 이러한 관계 정보를 담는 것이 Relational DB

Basic Structure of a Relation

- Formally, 이름 . ID . 전공 등 Domain → 모든 가능한 조합

  given sets $D_1, D_2, \ldots D_n$ a *relation* r is a subset of $D_1 \times D_2 \times \ldots \times D_n$

  ↓ 한 순서쌍이 한 개체를 표현한다

  $r \subseteq D_1, \times \ldots\ldots \times D_n$ (n: degree of r)

  or

  $r = \{ <d_1, \ldots, d_n> \mid d_1 \in D_1, \ldots, d_n \in D_n \}$ (set of tuples)

- Example: *customer-name* = {Jones, Smith, Curry, Lindsay}
  customer-street = {Main, North, Park}
  customer-city = {Harrison, Rye, Pittsfield}

  하나의 Relation

  Then r = { (Jones, Main, Harrison), (Smith, North, Rye), (Curry, North, Rye), (Lindsay, Park, Pittsfield)} → 순서쌍들의 집합

  is a relation over *customer-name x customer-street x customer-city*

Attribute Types

- Each attribute of a relation has a name
- 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
  - E.g. multivalued attribute values are not atomic
  - E.g. composite attribute values are not atomic

Relation Schema

- $A_1, A_2, \ldots, A_n$ are *attributes*

어떤 attribute로 구성되는가?

- $R = (A_1, A_2, \ldots, A_n)$ is a *relation schema*

e.g.

Customer-schema = (customer-name, customer-street, customer-city)

- $r(R)$ is a *relation* (*variable*) on the *relation schema R*

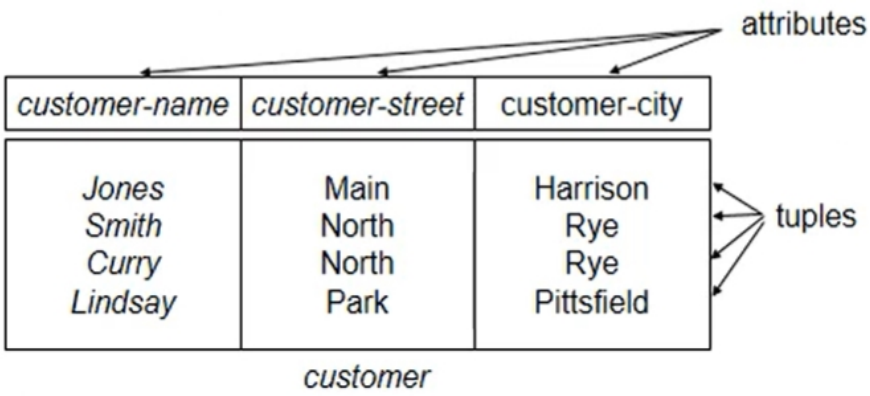
e.g. 데이터를 담는 그릇

customer(Customer-schema)

그릇의 구조

Relation Instance

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



Relations are Unordered → relation은 Set이다

- Order of tuples is irrelevant
  (tuples may be stored in an arbitrary order)

순서가 중요하지 않다 (의미 없다)
중복도 의미가 없다

- E.g. *instructor* relation with unordered tuples

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

Relational Database

relation들의 모임

- A database consists of multiple relations
- Information about an enterprise is broken up into parts, with each relation storing one part of the information

  E.g.: *instructor*: information about teachers in the university
  student: information about students
  advisor: information about which instructor advises which students

모든 속성을 하나의 큰 table로 만들기 ① 일단 구성이 어려움 ② 중복이 발생

- Storing all information as a single relation is not a good idea

  univ (*instructor -ID, name, dept_name, salary, student_Id*, ..)

- Database design (Chapter 7 & 8) deals with how to decide on the relational schemas

어떻게 효율적으로 relation들을 저장할 것인가?

A relational database for a university

ID	name	dept_name	salary
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565			
98345			
76766			
10101			
58583			
83821			
15151			
33456			
76543			

dept_name	building	budget
Biology	Watson	90000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Finance	Painter	120000
History	Painter	50000
Music	Packard	80000
Physics	Watson	70000

course_id	title	dept_name	credits
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design		
CS-315	Robotics		
CS-319	Image Proc		
CS-347	Database Sy		
EE-181	Intro. to Dig		
FIN-201	Investment		
HIS-351	World Histo		
MU-199	Music Video		
PHY-101	Physical Pri		

course_id	sec_id	semester	year	building	room_number	time_slot_id
BIO-101	1	Summer	2009	Painter	514	B
BIO-301	1	Summer	2010	Painter	514	A
CS-101	1	Fall	2009	Packard	101	H
CS-101	1	Spring	2010	Packard	101	F
CS-190	1	Spring	2009	Taylor	3128	
CS-190	2	Spring	2009	Taylor	3128	
CS-315	1	Spring	2010	Watson	120	
CS-319	1	Spring	2010	Watson	100	
CS-319	2	Spring	2010	Taylor	3128	
CS-347	1	Fall	2009	Taylor	3128	
EE-181	1	Spring	2009	Taylor	3128	
FIN-201	1	Spring	2010	Packard	101	
HIS-351	1	Spring	2010	Painter	514	
MU-199	1	Spring	2010	Packard	101	
PHY-101	1	Fall	2009	Watson	100	

course_id	prereq_id
BIO-301	BIO-101
BIO-399	BIO-101
CS-190	CS-101
CS-315	CS-101
CS-319	CS-101
CS-347	CS-101
EE-181	PHY-101

ID	course_id	sec_id	semester	year
10101	CS-101	1	Fall	2009
10101	CS-315	1	Spring	2010
10101	CS-347	1	Fall	2009
12121	FIN-201	1	Spring	2010
15151	MU-199	1	Spring	2010
22222	PHY-101	1	Fall	2009
32343	HIS-351	1	Spring	2010
45565	CS-101	1	Spring	2010
45565	CS-319	1	Spring	2010
76766	BIO-101	1	Summer	2009
76766	BIO-301	1	Summer	2010
83821	CS-190	1	Spring	2009
83821	CS-190	2	Spring	2009
83821	CS-319	2	Spring	2010
98345	EE-181	1	Spring	2009

Keys

Table에서 특정 record를 어떻게 identify 하는가?

unique해야 identify 가능
학번 등

각 column의 값을 기준으로 봄음

Set of Attributes

- Let $K \subseteq R$
- *K* is a *superkey* of *R*

  학번 or (이름, 학과, 주소)

  if values for *K* are sufficient to identify a unique tuple of each possible relation *r(R)*.
- By "possible *r*" we mean a relation *r* that could exist in the enterprise we are modeling.
  - Example: {*customer-name*, *customer-street*} and {*customer-name*}

    are both superkeys of *Customer*, if no two customers can possibly have the same name.
- *K* is a *candidate key* if *K* is minimal
  - Example: {*customer-name*} is a candidate key for *Customer*,

    since it is a superkey, and no subset of it is a superkey.

일반적으로 key라고 하면 candidate key

Keys

- **Primary key**: one of the candidate keys is selected to be the primary key of the table
  - which one?

DBA가 해당 domain에서 대표가 될 만한 attribute를 지정
Application에서 다른 곳에 제공하는 key를 선언하는 의미도 있음 : foreign key

- **Foreign key**: Value in one relation must appear in another
  - **Referencing** relation
  - **Referenced** relation

다른 테이블의 record와의 관계를 표현
(같은 table도 가능)

course_id	prereq_id
BIO-301	BIO-101
BIO-399	BIO-101
CS-190	CS-101
CS-315	CS-101
CS-319	CS-101
CS-347	CS-101
EE-181	PHY-101

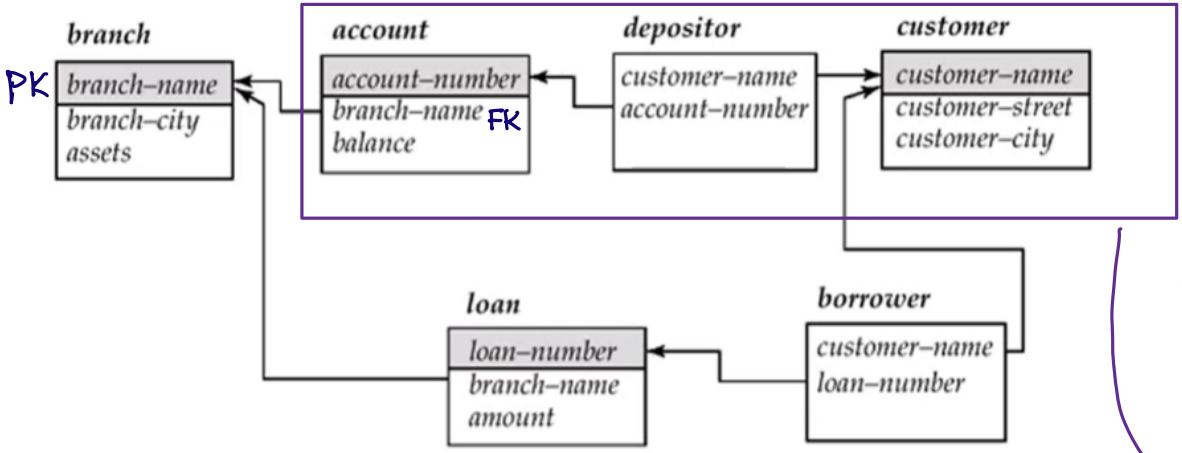
course relation

course_id	name	dept
⋮	⋮	⋮

foreign key → primary key

Schema Diagram for a Banking Enterprise

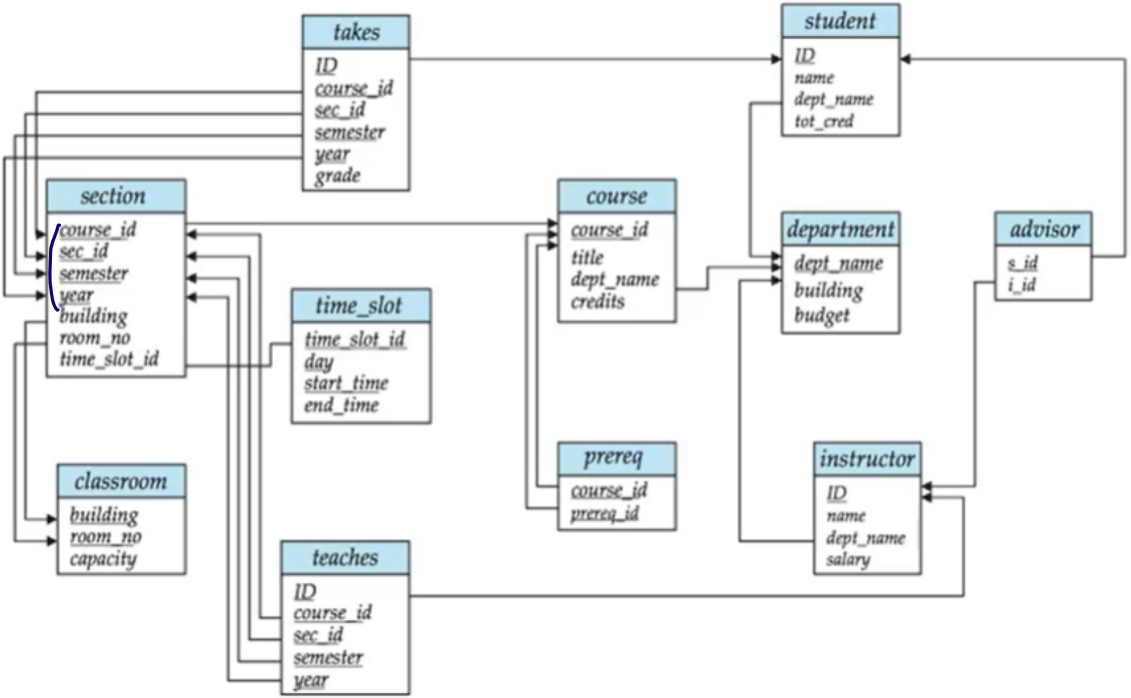
화살표 : Foreign Key reference를 표현



"서울에 사는 고객들의 balance를 표시해라"

customer-city로 검색 → customer-name → account-number → balance

Schema Diagram for the University



unique한 강좌를 만드는 조합 "1학년 2학기 컴구 002"

Query Languages

retrieve

- Language in which user requests information from the database.
  - procedural
    - specify what data are needed and how to get those data
  - nonprocedural
    - declarative
    - specify only what data are needed
- "Pure" languages:
  - Relational Algebra
  - Tuple Relational Calculus
  - Domain Relational Calculus
- Pure languages form underlying basis of query languages that people use.

정보를 가져오는 '절차'를 specify 함 ex) Relational Algebra

서울에 있는 Customer의 account number를 이용해 account의 balance를 가져와라

'조건'만 명시

ex) SQL

DB를 조작하고 정보를 뽑아내는 데 필요한 기능을 이론적으로 정의

내부적으로는 Relational Algebra가 쓰여도 우리는 SQL을 이용

Relational Algebra

Relation을 조작하는 operator들을 가진 System이다

- Algebra : operators and operands
  - Relational algebra
    - operands : relations
    - operators : basic operators (+ additional operations)
      - take two or more relations as inputs and give a new relation as a result.
- Procedural language
- Operators
  - select
  - project
  - union
  - set difference
  - Cartesian product
  - rename
  - join
  - division
  - assignment
  - …

자세한 내용은 6장 에서

Examples of Relational Operators

Symbol (Name)	Example of Use
σ (Selection)	$\sigma_{salary>=85000}(instructor)$
	Return rows of the input relation that satisfy the predicate.
Π (Projection)	$\Pi_{ID, salary}(instructor)$
	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.
× (Cartesian Product)	*instructor* × *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)	$\Pi_{name}(instructor) \cup \Pi_{name}(student)$
	Output the union of tuples from the two input relations.

Select Operation – selection of tuples

table의 record 들을 선별적으로 찾아냄

- Relation r :

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

Select from r

- $\sigma_{A=B \wedge D>5}(r)$:

satisfies condition

A	B	C	D
α	α	1	7
β	β	23	10

Project Operation – selection of columns

- Relation *r*:

A	B	C
α	10	1
α	20	1
β	30	1
β	40	2

selection 은 record 의 filter
projection 은 column "
orthogonal

- $\Pi_{A,C}(r)$

A	C
α	1
α	1
β	1
β	2

=

A	C
α	1
β	1
β	2

중복은 relation의 성질 (set) 에 의해 제거

Union Operation – merging two relations

합집합

- Relations *r, s:*

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

- $r \cup s$:

A	B
α	1
α	2
β	1
β	3

Set Difference Operation

차집합

- Relations *r*, *s*:

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

- *r* – *s*:

A	B
α	1
β	1

Set-Intersection Operation

교집합

- Relation *r*, *s*:

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

- $r \cap s$

A	B
α	2

Joining two relations: Cartesian-Product Op.

- Relations *r, s:*

r

A	B
α	1
β	2

2

s

C	D	E
α	10	a
β	19	a
β	20	b
γ	10	b

4

r의 모든 record와
s의 모든 record의 조합

- *r* x *s:*

A	B	C	D	E
α	1	α	10	a
α	1	β	19	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	19	a
β	2	β	20	b
β	2	γ	10	b

2x4 = 8

Joining two relations: Natural-Join Operation

- Relations *r*, *s*:

r

A	B	C	D
α	1	α	a
β	2	γ	a
γ	4	β	b
α	1	γ	a
δ	2	β	b

s

B	D	E
1	a	α
3	a	β
1	a	γ
2	b	δ
3	b	∈

- $r \bowtie s$

A	B	C	D	E
α	1	α	a	α
α	1	α	a	γ
α	1	γ	a	α
α	1	γ	a	γ
δ	2	β	b	δ

2x2 = 4

1x1 = 1

공유하는 attribute 에서 같은 값을 갖는 record 만 남김

Cartesian Product

Composition of Operations

- Can build expressions using multiple operations
- Example: $\sigma_{A=C}(r \times s)$
- $r \times s$

r

A	B
α	1
β	2

s

C	D	E
α	10	a
β	19	a
β	20	b
γ	10	b

A	B	C	D	E
α	1	α	10	a
α	1	β	19	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	19	a
β	2	β	20	b
β	2	γ	10	b

- $\sigma_{A=C}(r \times s)$

A	B	C	D	E
α	1	α	10	a
β	2	β	19	a
β	2	β	20	b

각 operator를 적용한 결과도 relation

→ operand로써 다시 operator 적용 가능

Natural Join

→ r×s 한 후 공유하는 attribute를 같게하는 조건으로 selection 한 후 공유되는 attribute 중 하나만 남기고 나머지를 projection 하는 것

END OF CHAPTER 2