



Chapter 2: Intro to Relational Model

These slides are a modified version of the slides provided with the book:

Database System Concepts, 6th Ed.

©Silberschatz, Korth and Sudarshan

See www.db-book.com for conditions on re-use

The original version of the slides is available at: <https://www.db-book.com/>



Example of a relation

- All the data is stored in various tables
- Example of tabular data in the relational model
- The set of allowed values for each attribute is called the **domain** of the attribute

<i>ID</i>	<i>name</i>	<i>dept_name</i>	<i>salary</i>
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

attributes
(or columns)

tuples
(or rows)



Relation schemas and instances

Let A_1, A_2, \dots, A_n be attributes and D_1, D_2, \dots, D_n be domains

- **relation schema**: describes the form of a tables (its definition, its meta-data)

Formally, the list of attributes and their domains

$R = (A_1, A_2, \dots, A_n)$ is a **relation schema** (sometimes information about domain is omitted)

Example: *instructor* = (*ID*: integer, *name*: string, *dept_name*: string, *salary*: integer)

instructor = (*ID*, *name*, *dept_name*, *salary*)

- **relation instance**: captures the actual content of a table (its values) at a particular moment

Formally, a **relation instance** r is a **finite** subset of $D_1 \times D_2 \times \dots \times D_n$

Thus, a relation instance is a **finite** set of n -tuples $\langle a_1, a_2, \dots, a_n \rangle$ where $a_i \in D_i$

Example: { $\langle 12121, "Wu", "Finance", 90000 \rangle$,
 $\langle 22222, "Einstein", "Physics", 95000 \rangle$,
 $\langle 33456, "Gold", "Physics", 87000 \rangle$ }

- Terminology: RELATIONS = TABLES – ATTRIBUTES = COLUMNS – TUPLES = ROWS

- Notation: lowercase letters for *relation instances* – r, s, r_1, r_2, \dots
uppercase letters for *relation schemas* – R, S, R_1, R_2, \dots
distinct lowercase letters for *tuples* – t, t_1, t_2, \dots

- Examples: $r(R)$ or $r \in R$ means “ r is an instance of relation with schema R ”
 $t \in r$ means “tuple t belongs to instance r ”



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
 - What does atomic means? It depends on the applications
- The special value ***null*** is a member of every domain. It indicates that the value is “unknown”, “undefined”, “optional”
 - Using a “dummy value” is not a good solution
- The ***null*** value causes complications in the definition of many operations



Relations are unordered

- Order of tuples is irrelevant (tuples may be stored in an arbitrary order)
- Same for attribute if columns have names (this is usually the case and we assume so)
- Example: *instructor* relation with unordered tuples

<i>ID</i>	<i>name</i>	<i>dept_name</i>	<i>salary</i>
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



Keys

- Let $K \subseteq R$ (R is a relation schema)
- 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?
- Terminology: by *key* we may refer to both candidate key or primary key (disambiguation by context)
- **Foreign key** constraint: Value in one relation must appear in another
 - **Referencing** relation
 - **Referenced** relation
 - Example – *dept_name* in *instructor* is a foreign key from *instructor* referencing *department*



A Sample Relational Database

<i>ID</i>	<i>name</i>	<i>dept_name</i>	<i>salary</i>
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

(a) The *instructor* table

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

(b) The *department* table



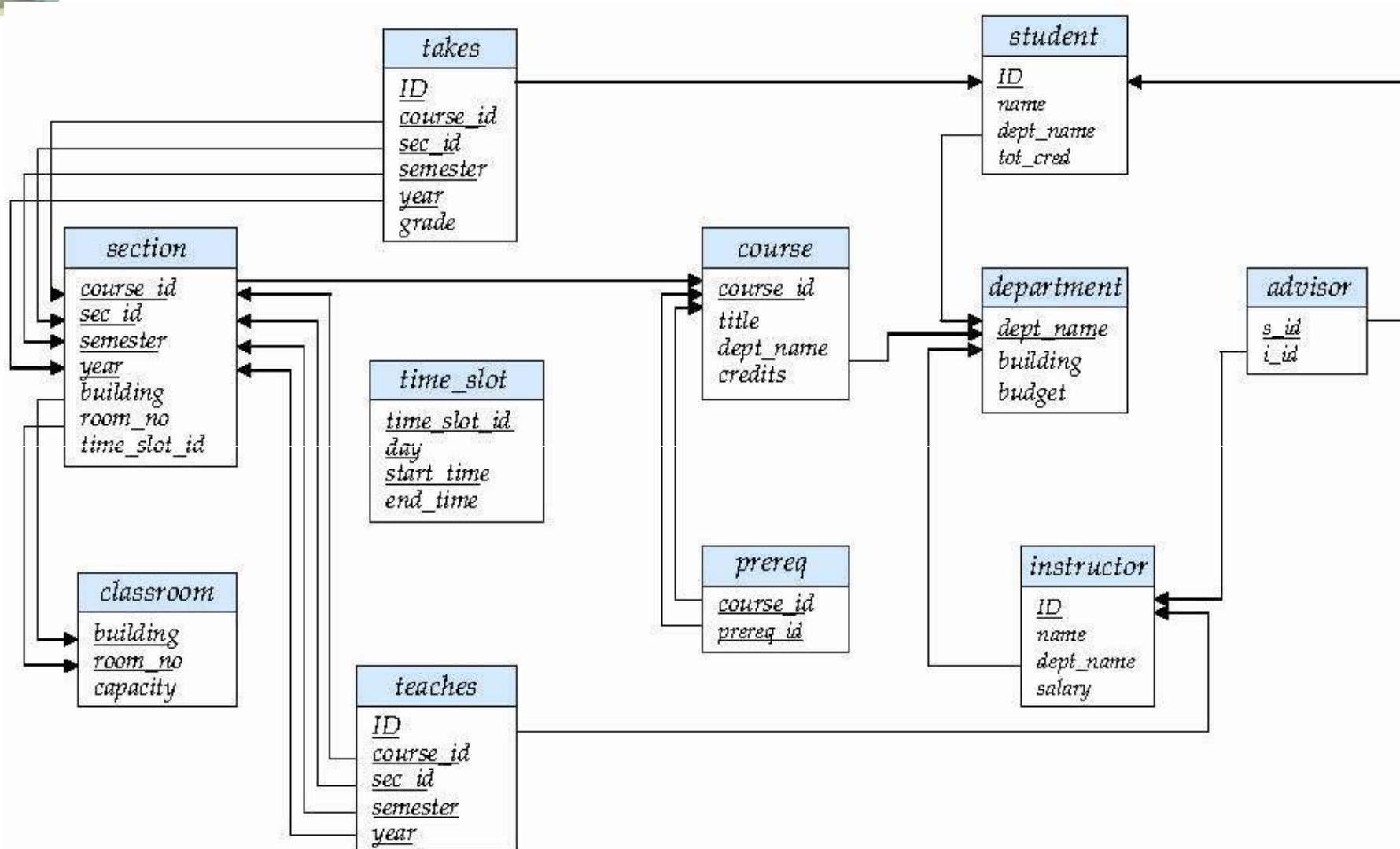
Database Design (Cont.)

- Is there any problem with this relation?

<i>ID</i>	<i>name</i>	<i>salary</i>	<i>dept_name</i>	<i>building</i>	<i>budget</i>
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000



Schema diagram for university database





Relational query languages

- Each Query input is a table (or set of tables)
- Each query output is a table.
- All data in the output table appears in one of the input tables
- Relational Algebra is not Turing complete

- Procedural vs .non-procedural, or declarative
- “Pure” languages:
 - Relational algebra
 - Tuple relational calculus
 - Domain relational calculus
- The above 3 pure languages are equivalent in computing power
- We will concentrate in this chapter on relational algebra
 - Not Turing machine equivalent
 - consists of 6 basic operations



Select operation – selection of rows (tuples)

- Relation r

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

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

A	B	C	D
α	α	1	7
β	β	23	10



Project operation – selection of columns (attributes)

- Relation r :

A	B	C
α	10	1
α	20	1
β	30	1
β	40	2

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

A	C
α	1
α	1
β	1
β	2

 $=$

A	C
α	1
β	1
β	2



Union of two relations

- 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 of two relations

- 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 of two relations

- 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

Note: $r \cap s = r - (r - s)$



Join of two relations – Cartesian product

■ Relations r, s :

A	B
α	1
β	2

r

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

s

■ $r \times s$:

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



Cartesian product – naming issue

■ Relations r, s :

A	B
α	1
β	2

r

B	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

s

■ $r \times s$:

A	$r.B$	$s.B$	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b



Renaming a Table

- Allows us to rename attributes of a relation

$$\rho_{C/B}(r)$$

where $r(R)$ and $B \in R$ (B is in is an attribute in the relation schema of r)

r

A	B
---	---

$\rho_{C/B}(r)$

A	C
---	---



Composition of operations

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

- $r \times s$

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



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 $R \cup S$ 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 $R \cap S$, 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



Natural join – Example

- Relations r, s :

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

r

B	D	E
1	a	α
3	a	β
1	a	γ
2	b	δ
3	b	ϵ

s

- Natural Join

- $r \bowtie s$

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

$$\Pi_{A, r.B, C, r.D, E} (\sigma_{r.B = s.B \wedge r.D = s.D} (r \times s))$$



Notes about Relational Languages

- Each Query input is a table (or set of tables)
- Each query output is a table.
- All data in the output table appears in one of the input tables
- Relational Algebra is not Turing complete
- Can we compute:
 - SUM
 - AVG
 - MAX
 - MIN



Summary of Relational Algebra Operators

Symbol (Name)	Example of Use
σ (Selection)	$\sigma \text{ salary} \geq 85000$ (<i>instructor</i>)
	Return rows of the input relation that satisfy the predicate.
Π (Projection)	$\Pi ID, salary$ (<i>instructor</i>)
	Output specified attributes from all rows of the input relation. Remove duplicate tuples from the output.
\times (Cartesian Product)	<i>instructor</i> \times <i>department</i>
	Output pairs of rows from the two input relations that have the same value on all attributes that have the same name.
\cup (Union)	$\Pi name$ (<i>instructor</i>) \cup $\Pi name$ (<i>student</i>)
	Output the union of tuples from the <i>two</i> input relations.
$-$ (Set Difference)	$\Pi name$ (<i>instructor</i>) $--$ $\Pi name$ (<i>student</i>)
	Output the set difference of tuples from the two input relations.
\bowtie (Natural Join)	<i>instructor</i> \bowtie <i>department</i>
	Output pairs of rows from the two input relations that have the same value on all attributes that have the same name.



End of Chapter 2

Database System Concepts, 6th Ed.

©Silberschatz, Korth and Sudarshan

See www.db-book.com for conditions on re-use