第一部分
Part 1
关系数据库

Database System Concepts, 6th Ed.

- **数据模型**是描述数据、数据联系、数据语义以及一致性约束的概念工具的集合。
- **关系模型**利用表的集合来表示
  - 数据和数据间的联系
- 关系模型在逻辑层和视图层描述数据，使用户不必关注数据存储的底层细节。

- 需要解决几个问题。
  - 最重要的问题是用户如何说明对数据的检索和更新请求，为此已经开发了好几种查询语言。
  - 第二个问题是数据完整性和数据保护。
    - 无论用户有意或无意地破坏数据，数据库都要保护数据及其完整性，使其免遭破坏。

Chapter 2: Intro to Relational Model

第2章 关系模型介绍

Database System Concepts, 6th Ed.

2.1 关系数据库的结构

- 关系数据库由表(table)的集合构成，每个表有唯一的名字。
- 一般说来，表中一行代表了一组值之间的一种***联系***。
- 由于一个表就是这种联系的一个集合，表这个概念和数学上的**关系**这个概念是密切相关的

- 在数学术语中，元组(tuple)只是一组值的序列（或列表）。
- 在n个值之间的一种联系，可以在数学上用关于这些值的一个n元组(n-tuple)来表示，
- 换言之，n元组就是一个有n个值的元组，它对应于表中的一行。

- 在关系模型的术语中，
  - **关系**(relation)用来指**表**，
  - 而**元组**(tuple)用指代表中的行
  - **属性**（attribute）指代的是表中的列
- **关系实例**（relation instance）表示一个关系的特定实例，也就是所包含的一组特定的行。

Example of a Relation

attributes
(or columns)

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)

Relation Schema and Instance

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

  Example:

  $$instructor = (ID, name, dept\_name, salary)$$

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

  $$D_1 \times D_2 \times \ldots \times D_n$$

  Thus, a relation is a set of n-tuples $(a_1, a_2, \ldots, a_n)$ where each $a_i \in D_i$
- 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

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

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. Indicated that the value is "**unknown**"
- The null value causes complications in the definition of many operations

2.2 数据库模式

- 数据库模式(database schema)和数据库实例(database instance)
  - 前者是数据库的逻辑设计，
  - 后者是*给定时刻*数据库中数据的一个快照。
- **关系模式**(relation schema)的概念对应于程序设计语言中类型定义的概念。
  - 关系模式由属性序列及各属性对应域组成。

- **关系实例**的概念对应于程序设计语言中**变量的值**的概念。
- 给定变量的值可能随时间发生变化；
- 当关系被更新时，关系实例的内容也随时间发生了变化。
- 相反，关系的模式是不常变化的。
- 在关系模式中，不同的关系使用相同属性，正是将不同关系的元组联系起来的一种方法。

2.3 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*

- 一个元组的属性值必须是能够**唯一**区分元组的。
- 换句话说，一个关系中没有两个元组在所有属性上的取值**都相同**。

- One of the candidate keys is selected to be the **primary key**.
- 习惯上，把一个关系模式的主键属性列在其他属性*前面*，还加上了*下划线*。

- 一个关系模式（如r1）如在它的属性中包括另一个关系模式（如r2）的主键，那么，这个属性在r1上称作参照r2的**外键**(foreign key)。
- 关系r1，称为有外键依赖的**参照关系**(referencing relation)
- r2，叫做外键的**被参照关系**(referenced relation)。

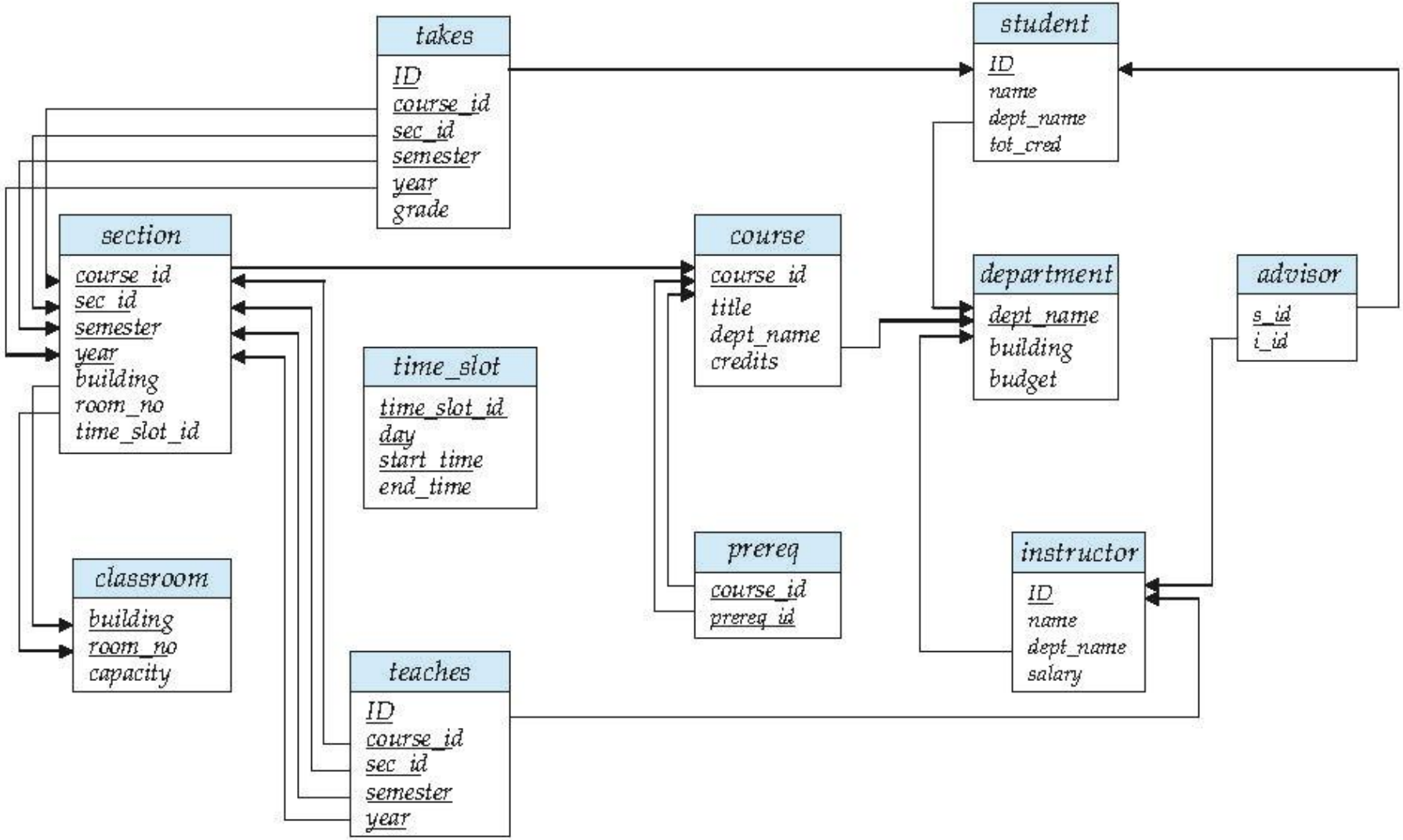
- **Foreign key** constraint: Value in one relation must appear in another
  - **Referencing** relation
  - **Referenced** relation
  - Example – *dept_name* in i*nstructor* is a foreign key from *instructor* referencing *department*

- 参照完整性约束（referential integrity constraint）要求
  - 在参照关系中，任意元组在特定属性上的取值必然等于被参照关系中某个元组在特定属性上的取值。

2.4 模式图

- 一个含有**主键**和**外键**依赖的数据库模式可以用模式图(schema diagram)来表示。
- 每一个关系用一个矩形来表示，关系的名字显示在矩形上方，矩形内列出各属性。
- 主键属性用下划线标注。
- 外键依赖用从参照关系的外码属性到被参照关系的主码属性之间的箭头来表示。

Schema Diagram for University Database



2.5 Relational Query Languages

- **查询语言**(query language)是用户用来从数据库中**请求**获取信息的语言。
- 查询语言可以分为
  - Procedural vs .non-procedural, or declarative

- **过程化语言**(procedural language)中，用户指导系统对数据库执行一系列操作以计算出所需结果。
- **非过程化语言**(nonprocedural language)中，用户只需描述所需信息，而不用给出获取该信息的具体过程。

- "Pure" Query languages:
  - Relational algebra (procedural language)
  - Tuple relational calculus (nonprocedural language)
  - Domain relational calculus (nonprocedural language)

- **关系代数**包括一个运算的**集合**（6种运算），这些运算以一个或两个关系为输入，产生一个新的关系作为结果输出。
- **关系演算**使用**谓词逻辑**来定义所需的结果，但不需给出获取结果的特定代数过程。

2.6 关系运算

- 所有的过程化关系查询语言都提供了一组运算
- 这些运算要么施加于单个关系上，要么施加于一对关系上。
- 运算结果总是单个的关系。
- 关系运算可施加到这个查询结果上
- 最常用的关系运算：select，project，product，union，difference，join

Summary of Relational Algebra 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.
x (Cartesian Product)	*instructor* x *department* Output pairs of rows from the two input relations that have the same value on all attributes that have the same name.
∪ (Union)	$\Pi_{name}(instructor) \cup \Pi_{name}(student)$ Output the union of tuples from the *two* input relations.
– (Set Difference)	$\Pi_{name}(instructor) - \Pi_{name}(student)$ Output the set difference of tuples from the two input relations.
⋈ (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.

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 ∪ 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: $r \cap s = r - (r - s)$

joining 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* x *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* x *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 refer to a relation, (say E) by more than one name.

$$\rho_x(E)$$

returns the expression E under the name X

- Relations r

A	B
α	1
β	2

r

- $r \times \rho_s(r)$

$r.A$	$r.B$	$s.A$	$s.B$
α	1	α	1
α	1	β	2
β	2	α	1
β	2	β	2

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 ⋈ 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 Turning complete
- Can we compute:
  - SUM
  - AVG
  - MAX
  - MIN

End of Chapter 2