



北京航空航天大学
BEIHANG UNIVERSITY

数据管理技术

北京航空航天大学

周号益

2024年



第二章 数据模型

1 什么是数据模型

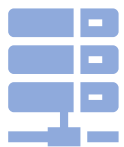
2 概念模型

3 逻辑模型

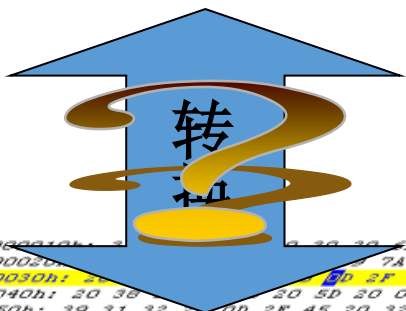
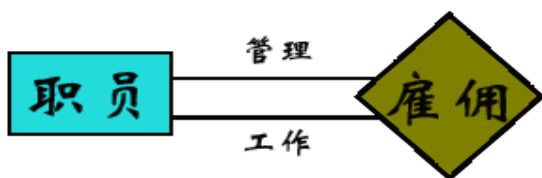


逻辑模型

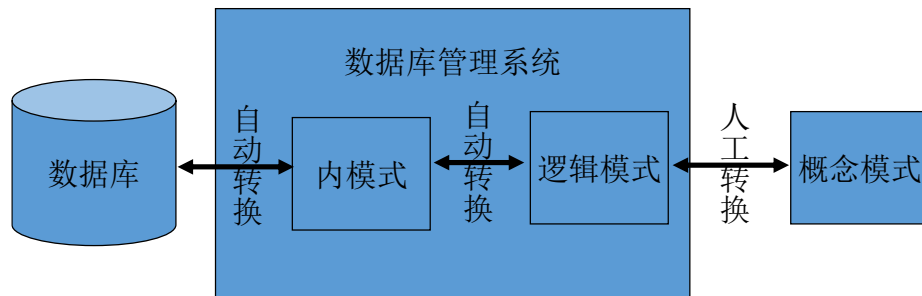
- 逻辑数据模型是对现实世界的第二层抽象。负责将概念数据模式映射为数据库的逻辑结构。
- 直接与DBMS有关,有严格的形式化定义,以便在计算机系统中实现。
- 它通常有一组严格定义的无二义性语法和语义的DB语言,人们可以用这种语言来定义、操纵DB中的数据。
- 逻辑数据模型的三要素:
 - **数据结构、数据操作和数据约束**



逻辑模型



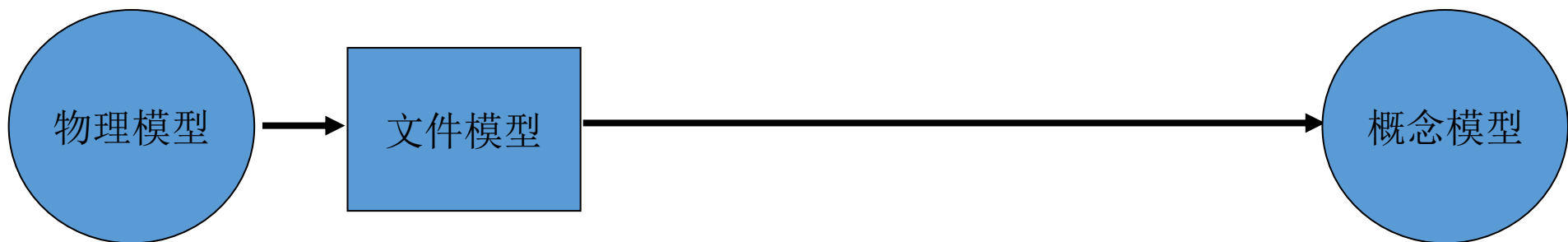
00000000h: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
00000001h: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
00000002h: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
00000003h: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
00000004h: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
00000005h: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
00000006h: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
00000007h: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
00000008h: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
00000009h: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
0000000ah: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
0000000bh: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
0000000ch: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
0000000dh: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
0000000eh: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
0000000fh: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
00000010h: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
00000011h: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
00000012h: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
00000013h: 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30





逻辑模型

■ 逻辑数据模型的发展



计算机处理能力:

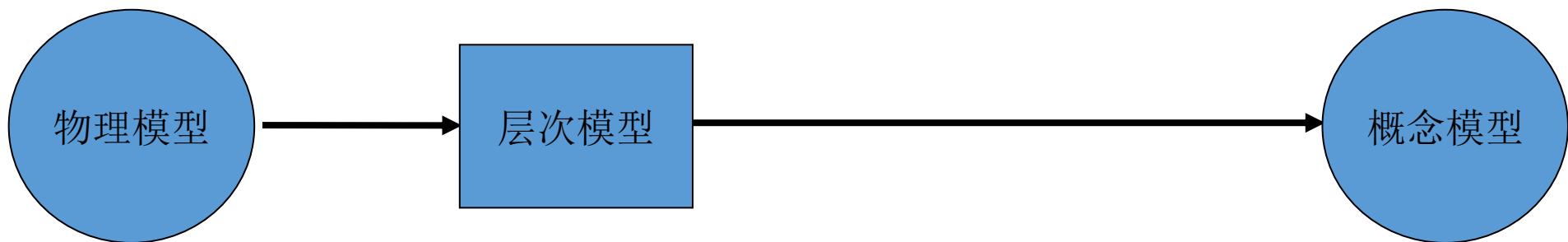


记录1	字段1	字段2
记录2	字段1	字段2
....			

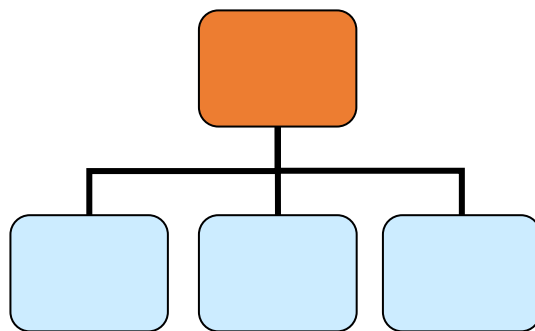


逻辑模型

■ 逻辑数据模型的发展



计算机处理能力

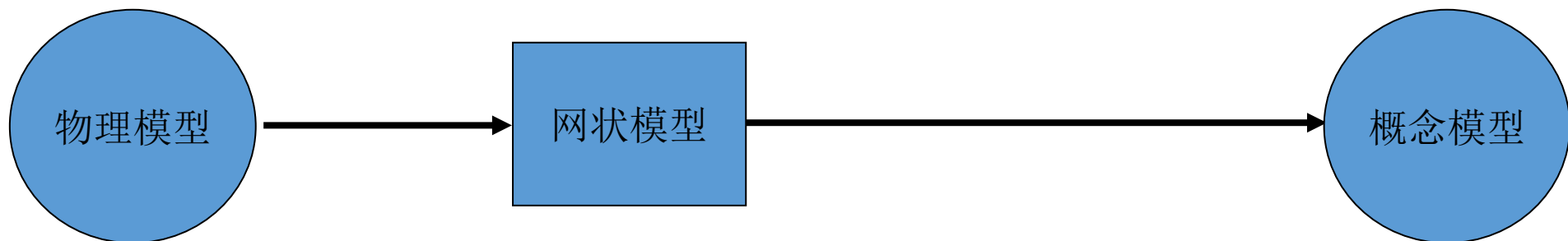


较复杂的数据结构
如树形结构

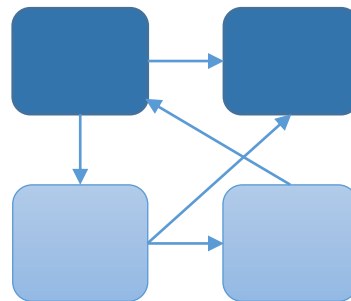


逻辑模型

■ 逻辑数据模型的发展



计算机处理能力

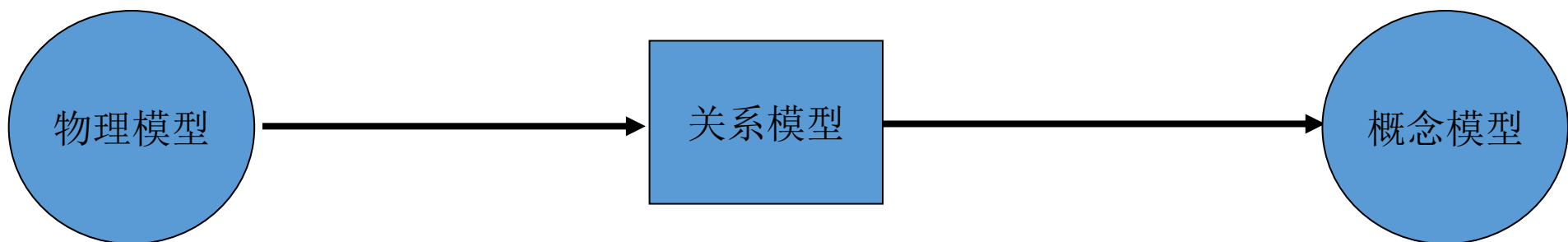


网状结构



逻辑模型

■ 逻辑数据模型的发展



计算机处理能力



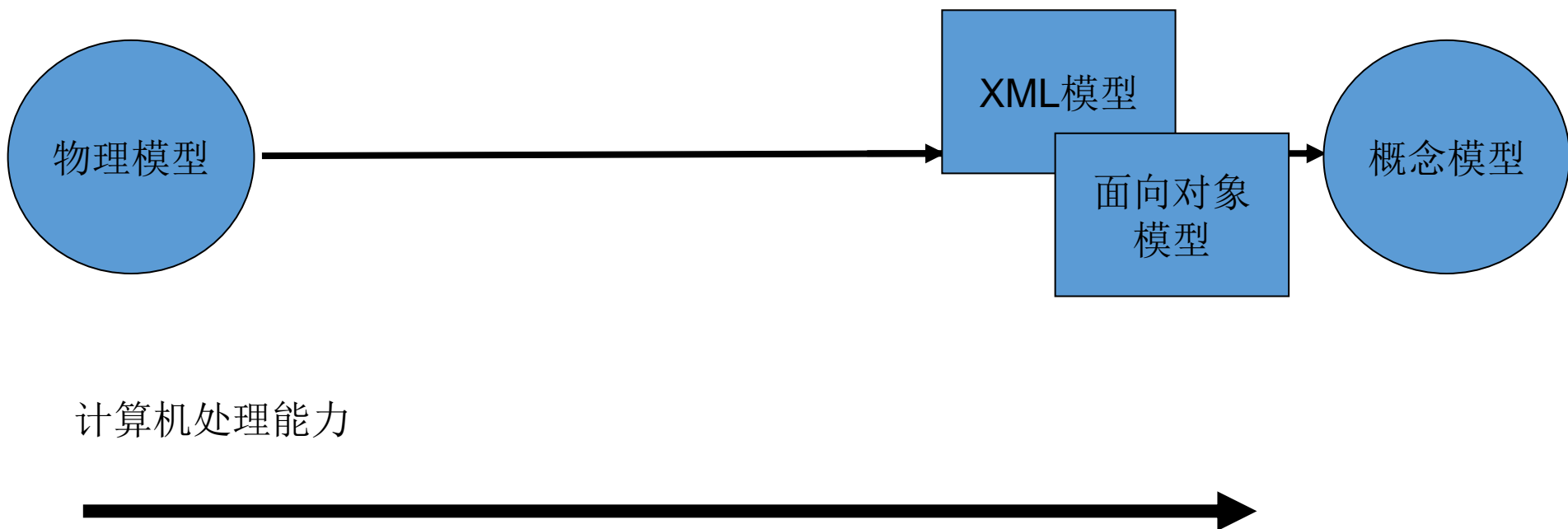
A	B	C
a	b	c
c	b	d

二维表结构



逻辑模型

■ 逻辑数据模型的发展





关系模型

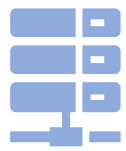
- 1 关系模型概述
- 2 关系数据结构 (结构)
- 3 关系的完整性 (约束)
- 4 关系代数 (操作)



关系模型概述

■关系模型发展历史

- 系统而严格地提出关系模型的是IBM公司的E.F.Codd
- 1970年提出关系数据模型
- E.F.Codd, "A Relational Model of Data for Large Shared Data Banks" , 《Communication of the ACM》,1970
- 之后, 提出了关系代数和关系演算的概念
- 1972年提出了关系的第一、第二、第三范式
- 1974年提出了关系的BC范式



关系模型概述

■ ER模型 vs. 关系模型

- 都用于数据建模
- ER模型有很多概念
 - ✓ 实体、关系、属性等
 - ✓ 适用于描述应用需求
 - ✓ 不适合在计算机上实现

(只有数据结构, 没有定义数据操作)
- 关系模型
 - ✓ 只有一个概念: 关系 (relation)
 - ✓ 用一组表的集合来描述世界
 - ✓ 适合在计算机上实现, 能进行高效的数据操作



关系模型概述

■ 关系数据库

- 关系数据库系统是支持关系模型的数据库系统
- 关系数据库应用严格的数学方法来处理数据库中的数据
- 80年代后，关系数据库系统成为最重要、最流行的数据库系统
- 典型的关系数据库产品:

ORACLE

PolarDB

SQL Server

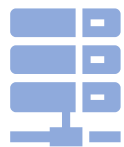
OpenGauss

MySQL

TDSQL-C

PostgreSQL

人大金仓



关系数据结构

- 关系模型就是用二维表格结构来表示实体及实体之间联系的模型
- 关系模型是各个关系的框架的集合，即关系模型是一些表格的格式，其中包括关系名、属性名、关键字等

- 关系的框架称为关系模式(relation schema)
- 关系框架和符合该框架的关系值称为关系实例(relation instance)

TNO 教师号	TN 姓名	SEX 性别	AGE 年龄	PROF 职称	SAL 工资	COMM 岗位津贴	DEPT 系别
T1	李力	男	47	教授	1500	3000	计算机
T2	王平	女	28	讲师	800	1200	信息
T3	刘伟	男	30	讲师	900	1200	计算机
T4	张雪	女	51	教授	1600	3000	自动化
T5	张兰	女	39	副教授	1300	2000	信息

表格格式

表格内容



关系数据结构

- 关系模型就是用二维表格结构来表示实体及实体之间联系的模型
- 关系模型是各个关系的框架的集合，即关系模型是一些表格的格式，其中包括关系名、属性名、关键字等

Table name
↓
Products:

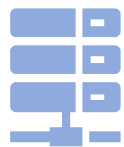
Fields Attribute names 属性

Name	Price	Category	Manufacturer
α550	¥ 5000	photography	Sony
TP T420	¥ 9000	notebook	Lenovo
I9100	¥ 3000	smartphone	samsung
IPAD	¥ 4000	tablet pc	Apple

关系术语

tuples 元组
Rows 行

columns 列



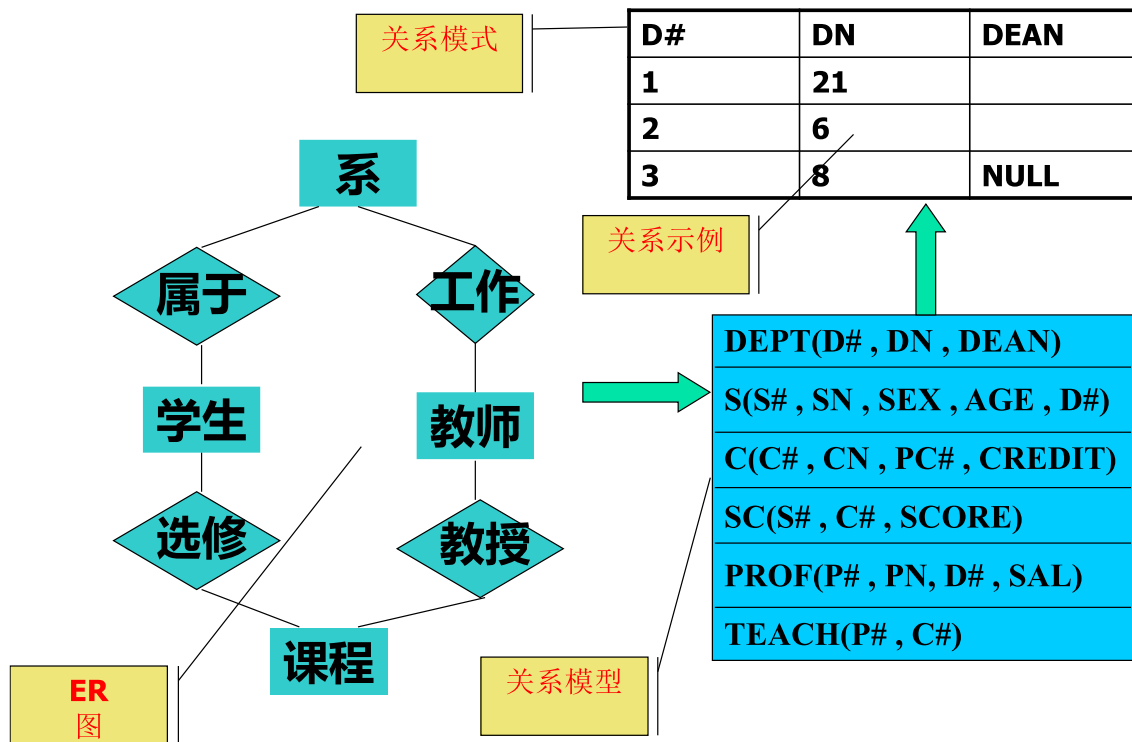
关系数据结构

ER模型与关系模型对比

- 注意关系模型和关系模式的区别
- 从各个关系的框架中，我们可以看出哪两个关系之间有联系。

例如：

- ✓ 系关系和学生关系有公共的属性“D#”，则表明这两个关系有联系。
- ✓ 而学生关系和选课关系有公共的属性“S#”，则表明这两个关系也有联系。





关系数据结构

■由上例可以看出，在一个关系中可以存放两类信息：

- 一类是描述实体本身的信息
- 一类是描述实体（关系）之间的联系的信息

■在层次模型和网状模型中，把有联系的实体（元组）用指针链接起来，实体之间的联系是通过指针来实现的。

■而关系模型则采用不同的思想，即用二维表来表示实体与实体之间的联系，这就是关系模型的本质所在。

■所以，在建立关系模型时，只要把的所有的实体及其属性用关系框架来表示，同时把实体之间的关系也用关系框架来表示，就可以得到一个关系模型。



关系数据结构

■关系的定义

- 在关系模型中，数据是以二维表的形式存在的，这个二维表就叫做关系。
- 关系理论是以集合代数理论为基础的，因此，我们可以用集合代数给出二维表的“关系”定义。
- 为了从集合论的角度给出关系的定义，我们先引入域和笛卡尔积的概念。



关系数据结构

■域 (Domain)

- 域是一组具有相同数据类型的值的集合，又称为值域。（用D表示）

✓ 例如整数、实数、字符串的集合。

- 域中所包含的值的个数称为域的基数（用m表示）。
- 关系中用域表示属性的取值范围。例如：

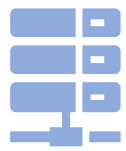
$D1 = \{\text{李力, 王平, 刘伟}\}$ $m1 = 3$

$D2 = \{\text{男, 女}\}$ $m2 = 2$

$D3 = \{47, 28, 30\}$ $m3 = 3$

其中，D1，D2，D3为域名，分别表示教师关系中姓名、性别、年龄的集合。

- 域值无排列次序，如 $D2 = \{\text{男, 女}\} = \{\text{女, 男}\}$



关系数据结构

■笛卡尔积 (Cartesian product)

- 给定一组域 D_1, D_2, \dots, D_n (它们可以有相同的元素, 即可以完全不同, 也可以部分或全部相同)
- D_1, D_2, \dots, D_n 的笛卡尔积为 $D_1 \times D_2 \times \dots \times D_n = \{ (d_1, d_2, \dots, d_n) \mid d_i \in D_i, i=1, 2, \dots, n \}$ 。
- 笛卡尔积每一个元素 $(d_1, d_2, d_3, \dots, d_n)$ 叫做一个n元组 (n-tuple) , 简称元组 (Tuple) 。
- 元素中的每一个 d_i 叫做一个分量(Component), 来自相应的域 ($d_i \in D_i$)
- 但元组不是 d_i 的集合, 元组的每个分量 (d_i) 是按序排列的
 - ✓ $(1, 2, 3) \neq (2, 3, 1) \neq (1, 3, 2)$;
 - ✓ 而集合中的元素是没有排序次序的, 如 $(1, 2, 3) = (2, 3, 1) = (1, 3, 2)$



关系数据结构

■笛卡尔积 (Cartesian product)

例 给出三个域:

$D_1 = \text{SUPERVISOR} = \{ \text{张老师, 刘老师} \}$

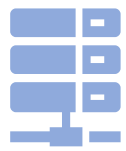
$D_2 = \text{SPECIALITY} = \{ \text{电子信息, 软件工程} \}$

$D_3 = \text{POSTGRADUATE} = \{ \text{李勇, 刘晨, 王敏} \}$

则 D_1, D_2, D_3 的笛卡尔积为:

$D_1 \times D_2 \times D_3 =$

{(张老师, 电子信息, 李勇), (张老师, 电子信息, 刘晨),
(张老师, 电子信息, 王敏), (张老师, 软件工程, 李勇),
(张老师, 信息专业, 刘晨), (张老师, 软件工程, 王敏),
(刘逸, 电子信息, 李勇), (刘逸, 电子信息, 刘晨),
(刘逸, 电子信息, 王敏), (刘逸, 软件工程, 李勇),
(刘逸, 软件工程, 刘晨), (刘逸, 软件工程, 王敏) }



关系数据结构

■笛卡尔积的基数

若 D_i ($i=1, 2, \dots, n$) 为有限集, D_i 中的集合元素个数称为 D_i 的基数, 用 m_i ($i=1, 2, \dots, n$) 表示, 则笛卡尔积 $D_1 \times D_2 \times \dots \times D_n$ 的基数 M (即元素 (d_1, d_2, \dots, d_n) 的个数) 为所有域的基数的累乘之积, 即

$$M = \prod_{i=1}^n m_i$$

例 $D_1 = \{\text{李力, 王平, 刘伟}\}$, $D_2 = \{\text{男, 女}\}$

则 基数 $M = m_1 \times m_2 = 3 \times 2 = 6$

$D_1 \times D_2 = \{(\text{李力, 男}), (\text{李力, 女}), (\text{王平, 男}), (\text{王平, 女}), (\text{刘伟, 男}), (\text{刘伟, 女})\}$,



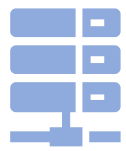
关系数据结构

■笛卡尔积的二维表表示

笛卡尔积可用二维表的形式表示。例如，前例的6个元组可表示成如下二维表：

姓名	性别
李力	男
李力	女
王平	男
王平	女
刘伟	男
刘伟	女

可见，笛卡尔积实际是一个二维表，表的框架由域构成，表的任意一行就是一个元组，表中的每一列来自同一域，如第一个分量来自D1，第二个分量来自D2。



关系数据结构

■关系 (Relation) 的定义

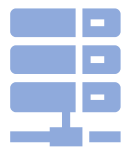
笛卡尔积 $D_1 \times D_2 \times \dots \times D_n$ 的任一子集称为定义在域 D_1, D_2, \dots, D_n 上的 n 元关系 (Relation), 可用 $R(D_1, D_2, \dots, D_n)$ 表示

✓ 如上例 $D_1 \times D_2$ 笛卡尔积的子集可以构成教师关系 T_1 :

✓ R 为关系名, n 称为关系的元(目或度) (Degree)。
当 $n=1$ 时, 称为单元关系。
当 $n=2$ 时, 称为二元关系。
...
当 $n=n$ 时, 称为 n 元关系。
如上例为二元关系, 关系名为 T_1 。

T_1

姓名	性别
李力	男
王平	女
刘伟	男



关系数据结构

■关系 (Relation) 的定义

数学上关系是笛卡尔积的任意子集，但在实际应用中关系是笛卡尔积中所取的**有意义的**子集。

例如从笛卡尔积中选取子集构成如下关系，显然不符合实际情况：

姓名	性别
李力	男
李力	女
王平	男
王平	女
刘伟	男
刘伟	女

笛卡尔积

姓名	性别
李力	男
李力	女

错误的关系



关系数据结构

■关系的二维表表示

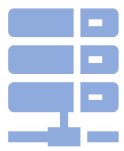
同样可以把关系看成一个二维表。其中，

- (1) 表的框架由域 D_i ($i=1, 2, \dots, n$) 构成;
- (2) 表的任意一行对应一个元组;
- (3) 表的每一列来自同一域;
- (4) 域可以相同, 为了加以区别, 每列起一个名字, 称为属性, n 目关系有 n 个属性, 属性的名字唯一, 属性的取值范围 D_i ($i=1, 2, \dots, n$) 称为值域
- (5) 具有相同关系框架的关系成为同类关系, 如:

姓名	性别
李力	男
王平	女
刘伟	男

姓名	性别
马华	男
赵新	女

同类关系



关系数据结构

■关系的术语

Relation as table

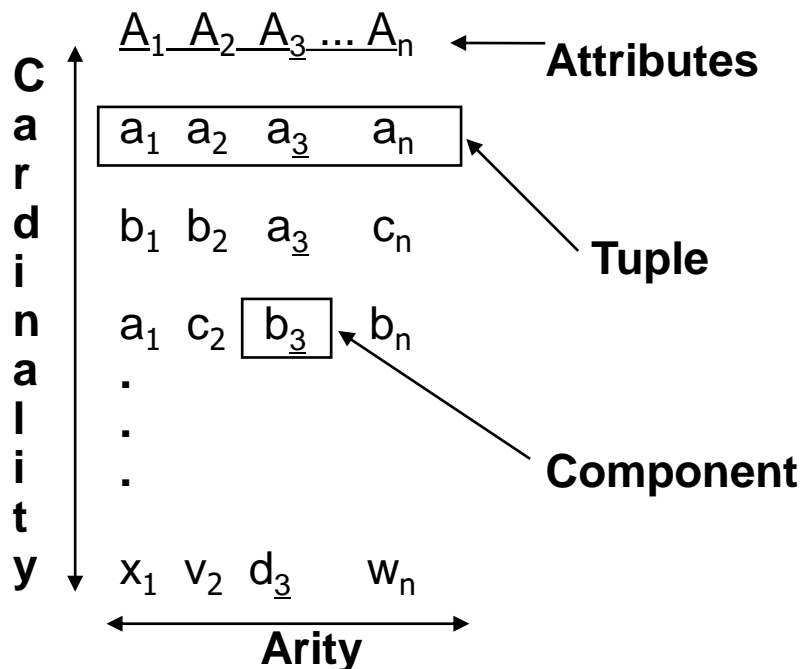
Rows = tuples

Columns = components

Names of columns = attributes

Relation name + set of attribute names = schema (关系模式)

$REL(A_1, A_2, \dots, A_n)$



- Set theoretic
- Domain — set of values
 - like a data type
- Cartesian product
 - $D_1 \times D_2 \times \dots \times D_n$
 - n-tuples (V_1, V_2, \dots, V_n)
- Relation = subset of cartesian product of one or more domains
 - FINITE only; empty set allowed
- Tuples(元组) = members of a relation inst.
- Arity (元) = number of domains
- Components (分量) = values in a tuple
- Domains(域) — corresp. with attributes
- Cardinality (基数) = max number of tuples



关系数据结构

■关系的性质

- 尽管关系与二维表格、传统的数据文件是非常类似的，但它们之间又有重要的区别。
- 严格地说，关系是种规范化了的二维表中行的集合，为了使相应的数据操作简化，在关系模型中，对关系作了种种限制，关系具有如下特性：
 1. 关系中不允许出现相同的元组。因为数学上集合中没有相同的元素，而关系是元组的集合，所以作为集合元素的元组应该是唯一的。

<i>account_number</i>	<i>branch_name</i>	<i>balance</i>
A-215	Mianus	700
A-215	Mianus	700
A-102	Perryridge	400
A-305	Round Hill	350
A-201	Brighton	900
A-222	Redwood	700
A-217	Brighton	750



关系数据结构

■关系的性质

2. 关系中元组的顺序（即行序）是无关紧要的，在一个关系中可以任意交换两行的次序。因为集合中的元素是无序的，所以作为集合元素的元组也是无序的。

3. 关系中属性的顺序也是无关紧要的，即列的顺序可以任意交换。交换时，应连同属性名一起交换，否则将得到不同的关系。

性别	姓名	=	姓名	性别	<>	姓名	性别
男	李力		李力	男		男	李力
女	王平		王平	女		女	王平
男	刘伟		刘伟	男		男	刘伟



关系数据结构

■关系的性质

4.同一属性名下的各个属性值必须来自同一个域，是同一类型的数据

5.关系中各个属性必须有不同的名字，不同的属性可来自同一个域，即它们的分量可以取自同一个域。

例如，有如下表中关系，职业与兼职是两个不同的属性，但它们取自同一个域职业 = {教师，工人，辅导员}。

姓名	职业	兼职
张强	教师	辅导员
王丽	工人	教师
刘宁	教师	辅导员



关系数据结构

■关系的性质

6.关系中每一分量必须是不可分的数据项。所有属性值都是原子的，是一个确定的值，而不是值的集合。不可“表中有表”。满足此条件的关系称为**规范化关系**，否则称为**非规范化关系**。

如下表，左边的关系中籍贯含有省、市 / 县两项，出现了“表中有表”的现象，则为非规范化关系，而把籍贯分成省、市 / 县两列，将其规范化，得到右边的规范化表

姓名	籍贯	
	省	市 / 县
张强	吉林	长春
王丽	山西	大同

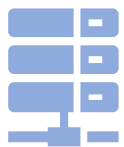
姓名	省	市 / 县
张强	吉林	长春
王丽	山西	大同



ER模型向关系模型转化

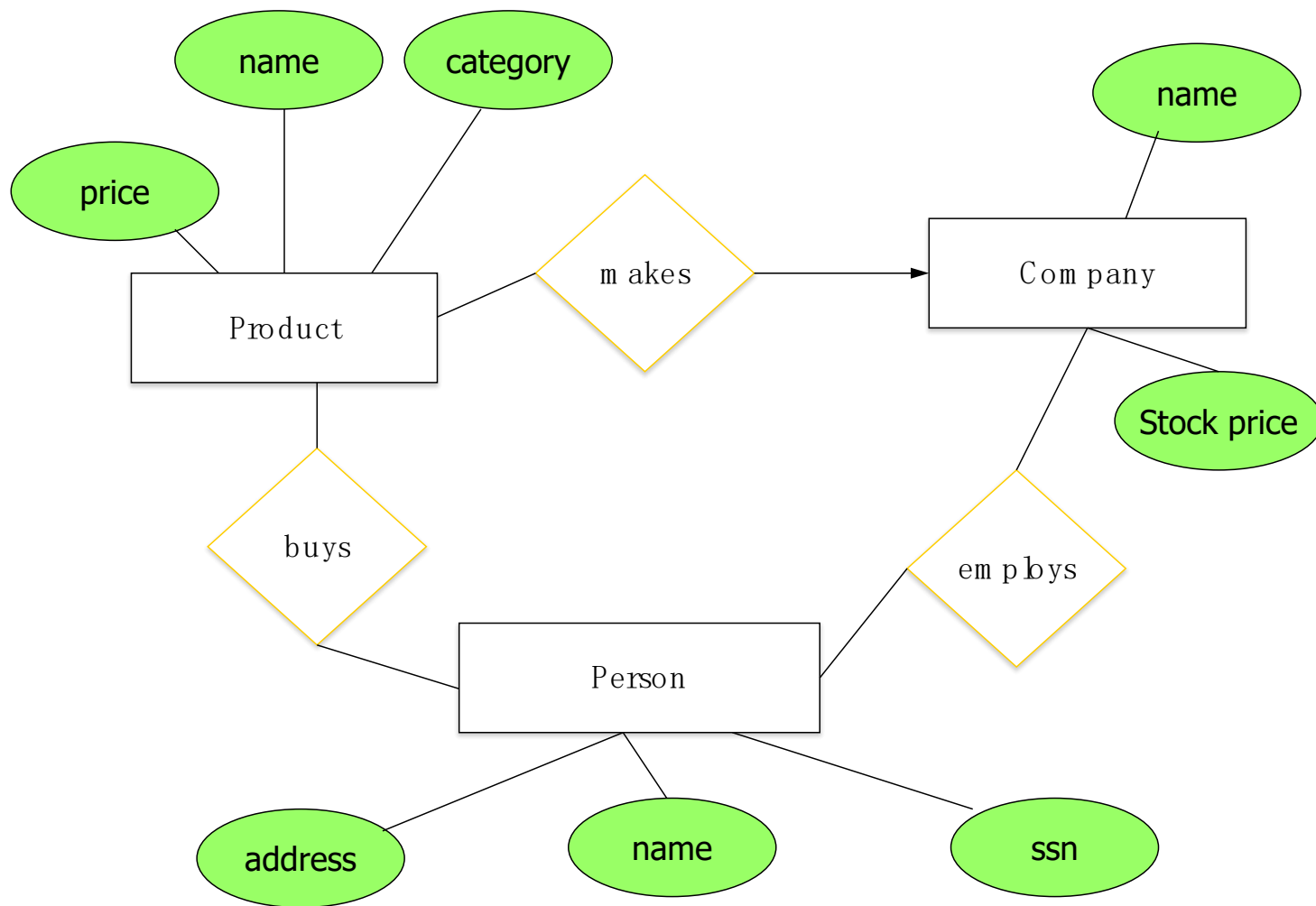
■ER模型向关系模型的转化（初步）

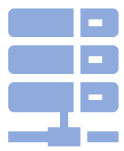
- ER模型向关系模型的转换，实际上就是把ER图转换成关系模式的集合。
- **规则1**（实体类型的转换）：将每个实体类型转换成一个关系模式，实体的属性即为关系模式的属性，实体标识符即为关系模式的键。
- **规则2**（二元联系类型的转换）
 - ① 若实体间联系是1:1。
 - ② 若实体间联系是1:N。
 - ③ 若实体间联系是M:N。



ER模型向关系模型转化

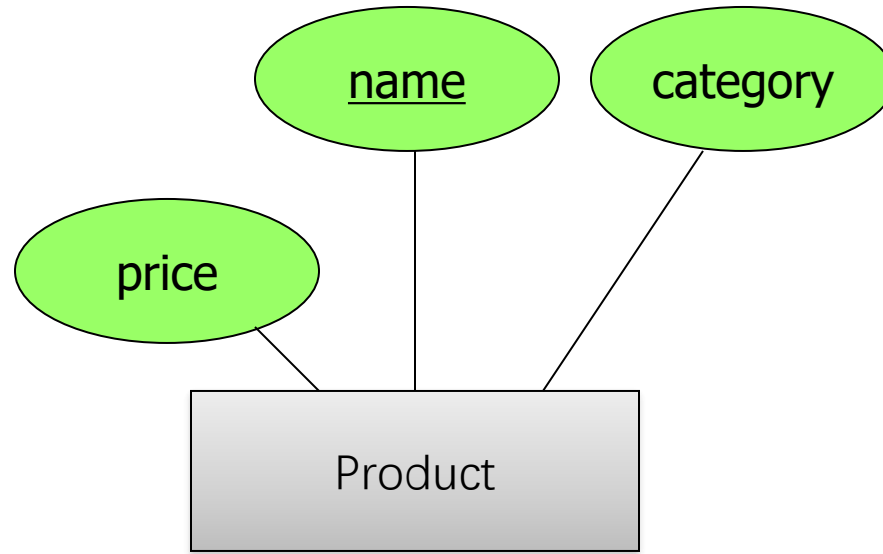
待转化的ER模型





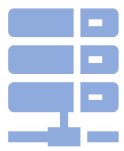
ER模型向关系模型转化

1. 把实体转化为关系



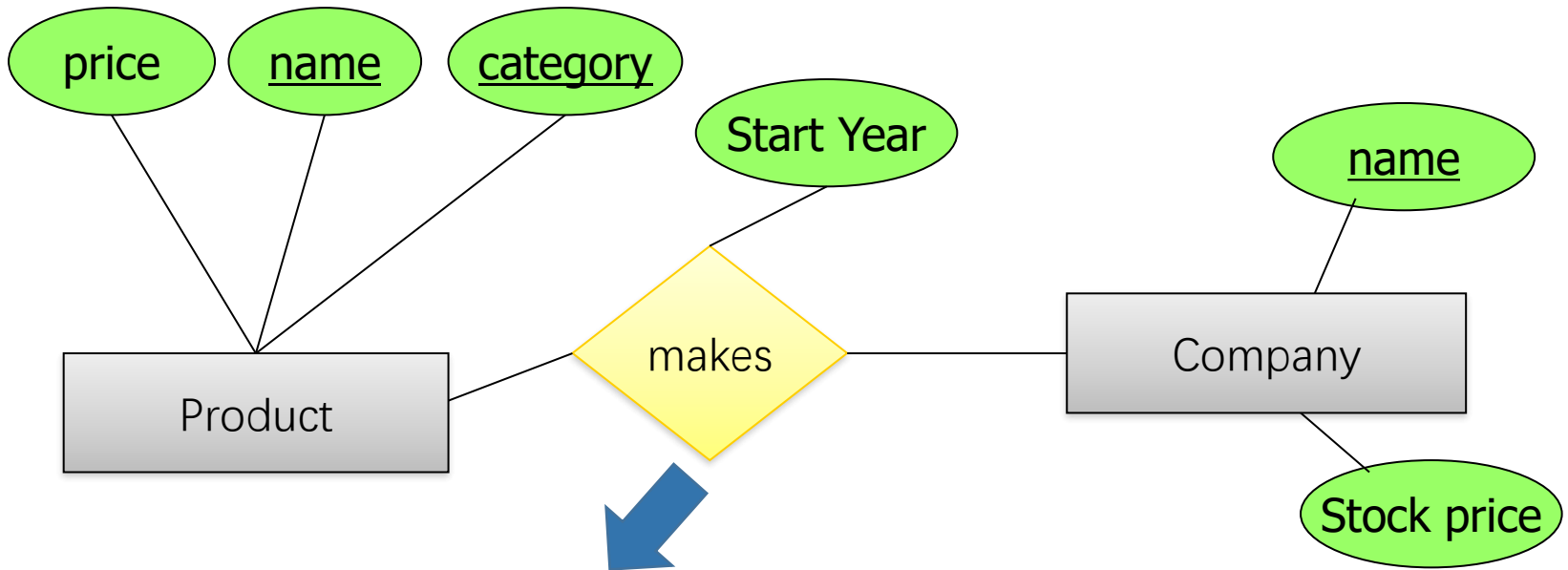
Product:

<u>Name</u>	Category	Price
ipad1	tabletpc	¥ 1999



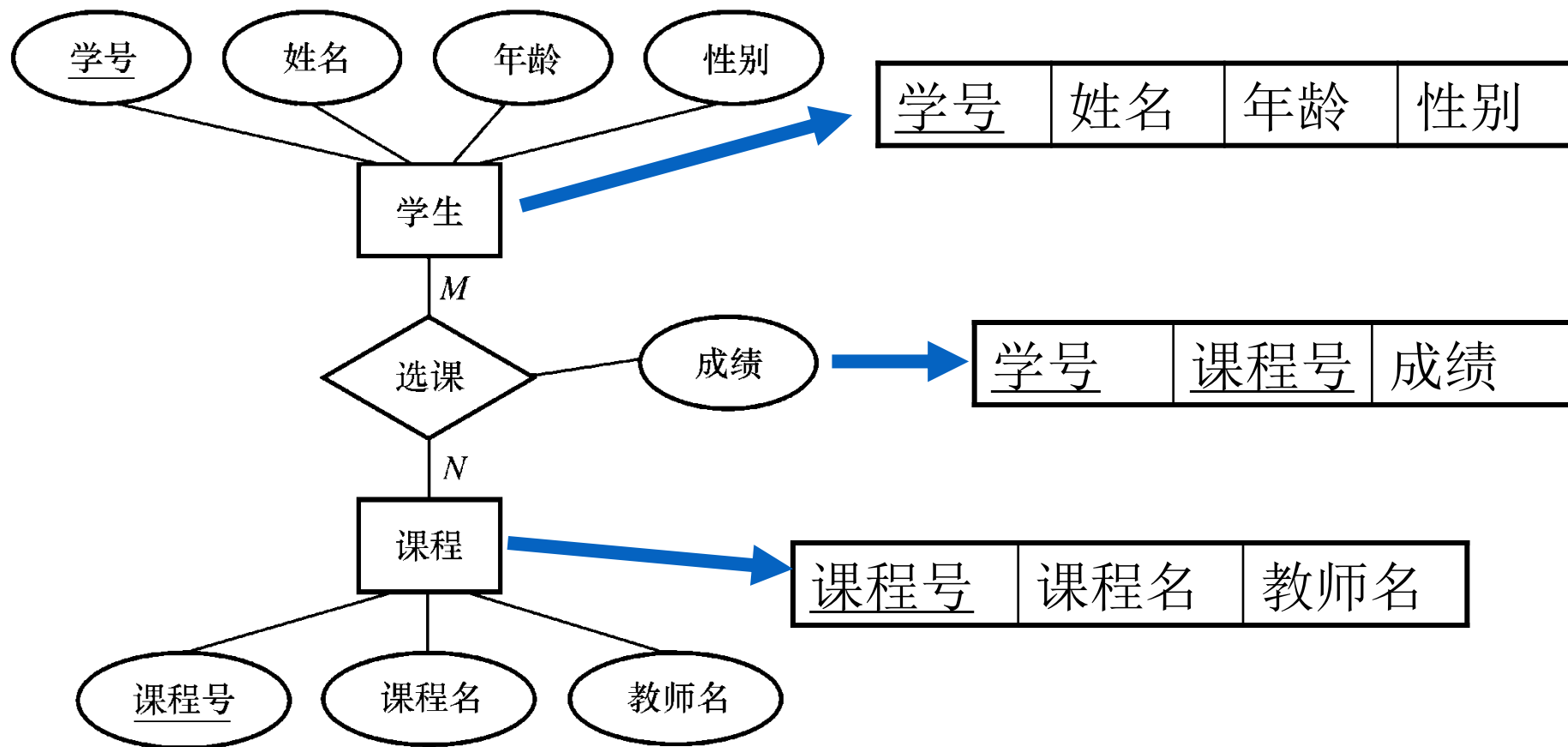
ER模型向关系模型转化

2. 把m:n联系转化为关系



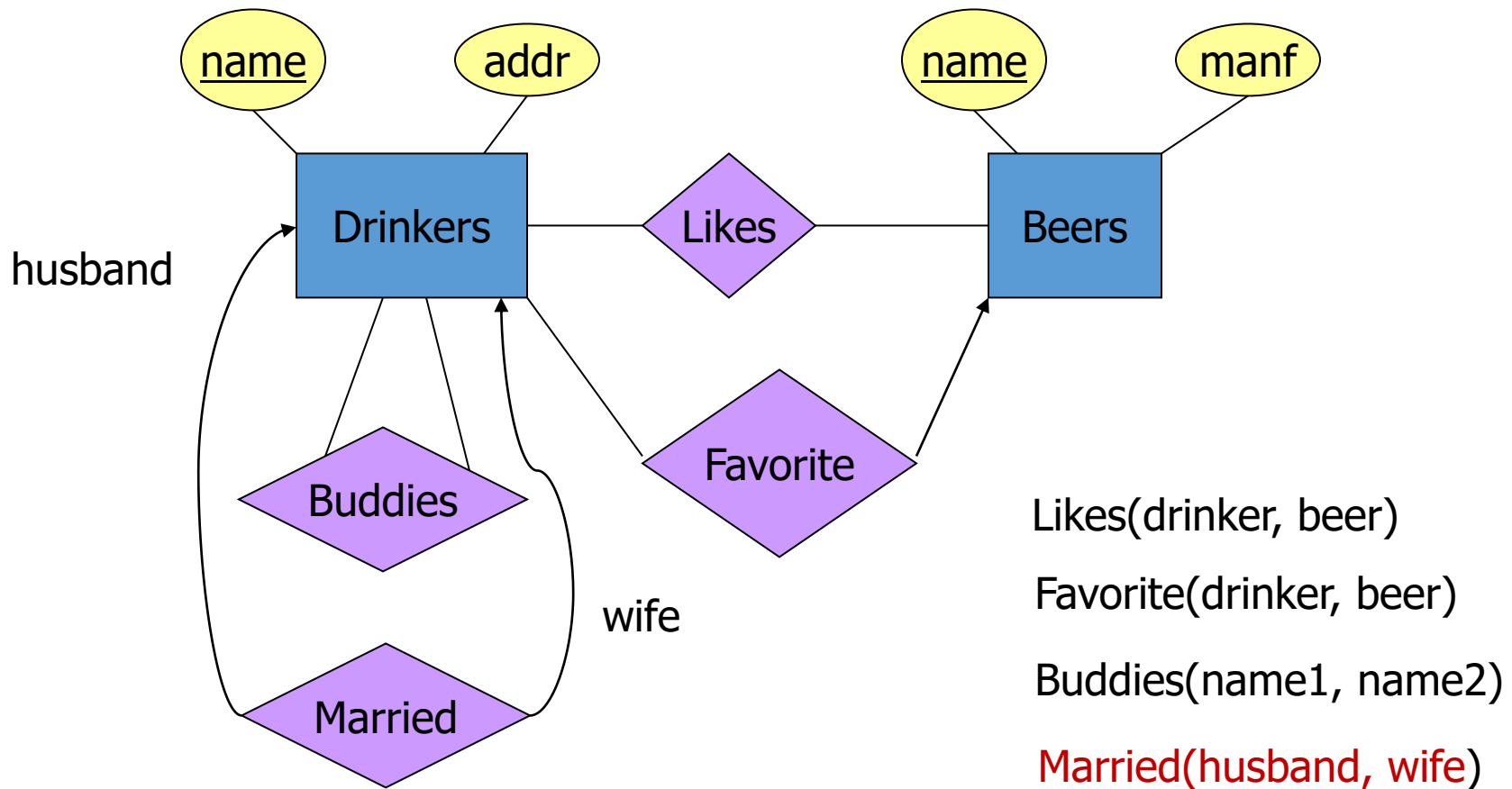
Makes

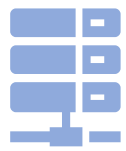
<u>Product-name</u>	<u>Product-Category</u>	<u>Company-name</u>	Starting-year
ipad1	tabletPC	Apple	2010



课程号	学号	成绩
111	3221	80
111	3222	79
112	3221	86

更多例子

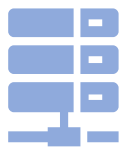




ER模型向关系模型转化

3. 对于1：n联系和1：1联系的处理

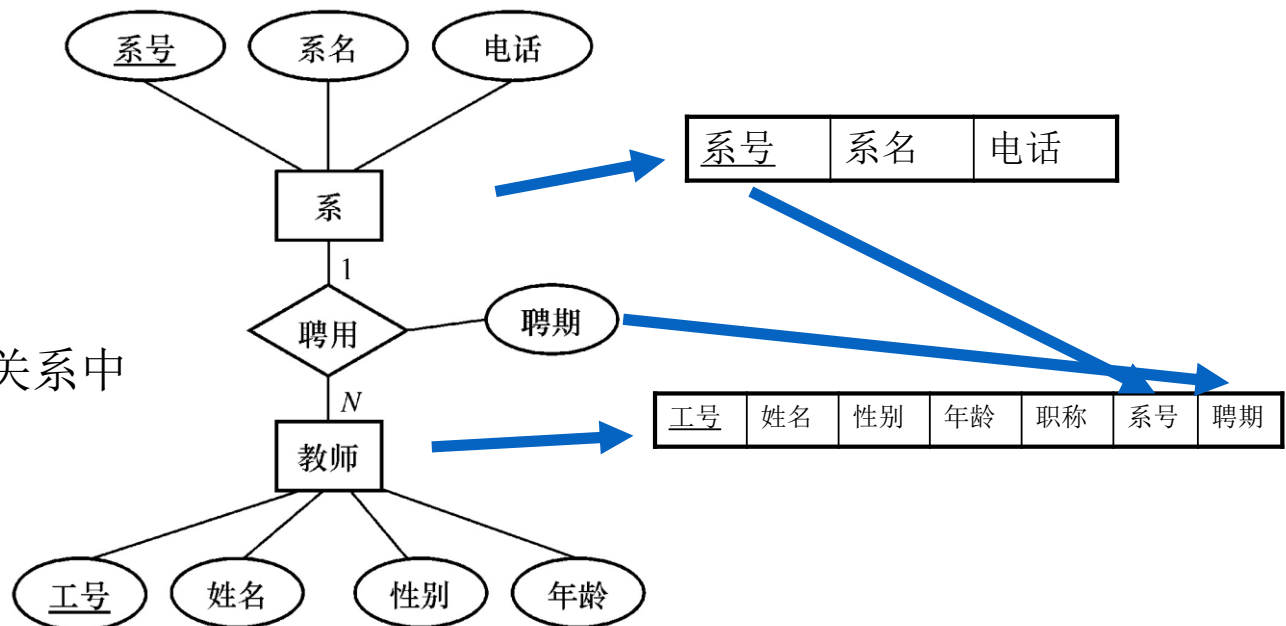
- ✓ 可以像m：n联系一样，直接转换为一个关系
- ✓ 但更好的处理方式是将1：n或1：1联系合并到相关联的实体转换出来的关系中
- ✓ 这样的好处是可以减少表的数量，更方便查询并加快查询速度



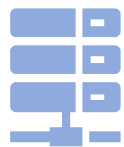
ER模型向关系模型转化

3. 对于1:n联系的处理

➤ 1:n联系合并方法：
合并到n段对应的实体关系中



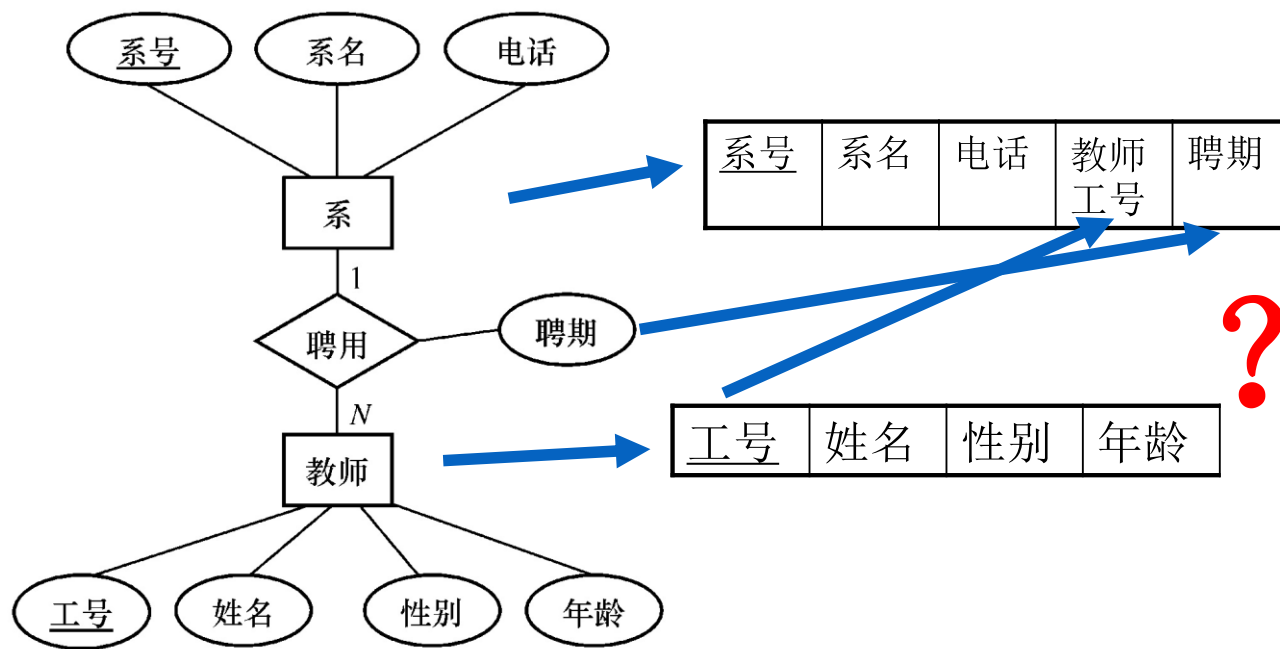
工号	教师姓名	...	系号
7579	张老师	...	21
7578	李老师	...	21
7577	马老师	...	21

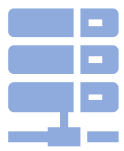


ER模型向关系模型转化

3. 对于1:n联系的处理

1:n联系合并到1端对应的实体关系中，会怎样？

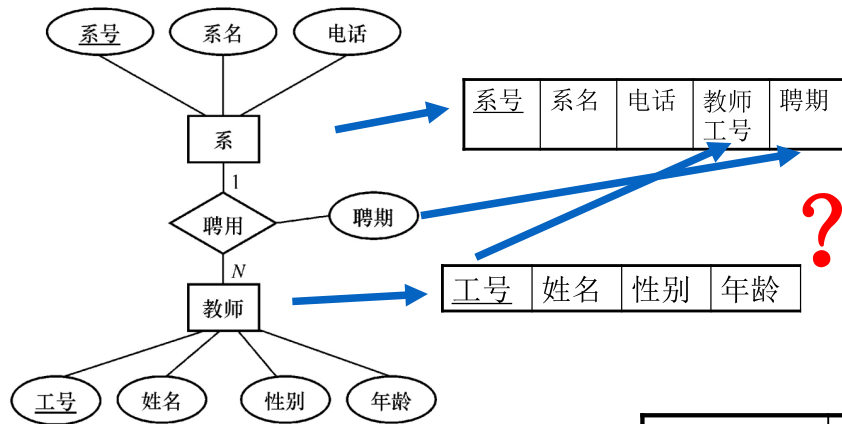




ER模型向关系模型转化

3. 对于1:n联系的处理

1:n联系合并到1端对于的实体关系中，会怎样？



系号	系名	...	教师号
21	软件学院	...	7579
21	软件学院	...	7578
21	软件学院	...	7577



出现数据冗余
且系号违反了
主键约束

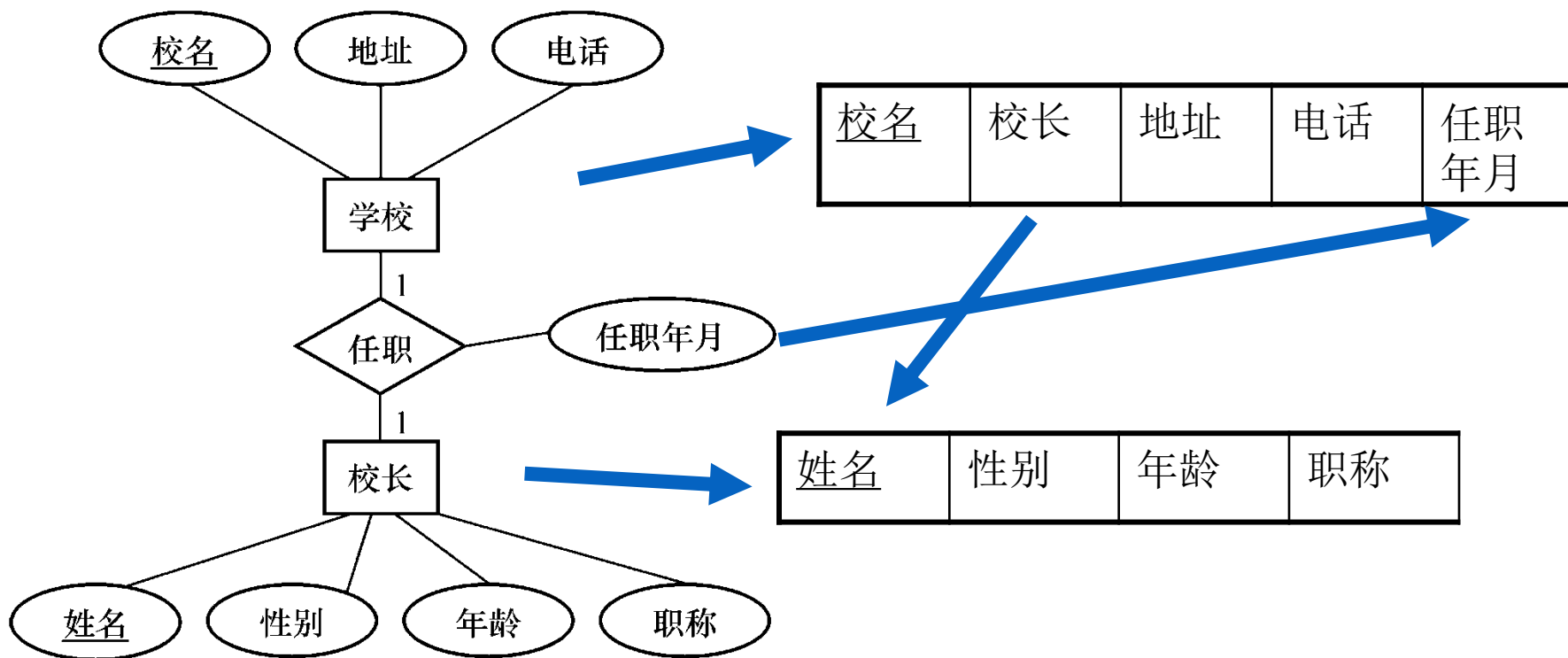


ER模型向关系模型转化

4. 对于1:1联系的处理

1:1联系可以合并到任意端对于的实体关系中

但应考虑哪个实体是被经常使用的。1:1联系应与常用实体相合并

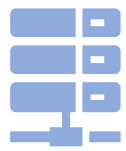


校名	地址	...	校长
A大学	文一路	...	张校长
B大学	学院路	...	李校长
C大学	复兴路	...	马校长



姓名	性别	年龄	...	校名
张校长	女	53	...	A大学
李校长	男	46	...	B大学
马校长	男	60	...	C大学





ER模型向关系模型转化

5. 对于三元联系的处理

延伸阅读: Transforming *N*-ary
Relationships to Database Schemas:
An Old and Forgotten Problem
--- Rafael Camps

常见转换方法:

1:1:1 联系与其中一个实体集合并

1:1:N 联系与N端实体集合合并

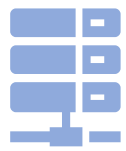
1:M:N 联系转换为独立的关系模式, 主键为**1**端实体集的主键

M:N:P 联系转换为独立的关系模式, 主键为三个实体集的主键的组合



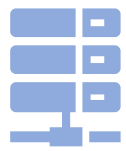
关系模型

- 1 关系模型概述
- 2 关系数据结构 (结构)
- 3 关系的完整性 (约束)
- 4 关系代数 (操作)



关系的完整性

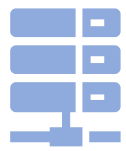
- 关系模型的完整性规则是对关系的某种约束条件
 - 对数据库状态的预测或断言
 - 总是为真（数据库更新的时候会确认状态是否符合完整性约束）
- 关系模型中的完整性约束：
 1. 域完整性 (Domain Integrity)
 2. 实体完整性 (Entity Integrity)
 3. 参照完整性 (Referential Integrity)
 4. 用户定义的完整性 (User-defined Integrity)



关系的完整性

■域完整性 (Domain Integrity)

- 属性值应符合域的取值范围
- 可以用于增强数据类型
 - 例如：为年龄属性设定 ≤ 150 的域完整性约束
- 对属性值能否为空 (NULL) 的检查也是域完整性约束的一部分
 - 空值 (NULL) 是值的一种，代表 ‘值未知’ 或 ‘值不存在’



关系的完整性

- 实体完整性（Entity Integrity）
 - 实体完整性用主键来约束
 - 实施了实体完整性约束的关系，其主属性（构成主键的属性）应同时满足以下条件：
 - ✓ 主属性不能为空
 - ✓ 主属性取值唯一



关系的完整性

■ 实体完整性（Entity Integrity）

- 实体完整性用主键来约束
- 实施了实体完整性约束的关系，其主属性（构成主键的属性）应同时满足以下条件：

✓ 主属性不能为空

✓ 主属性取值唯一

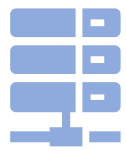
选课关系

学号	课程号	选课时间
S001	C001	2022/2/1
S001	C002	2022/2/2
S001	C001	2022/2/1
S001	C001	2022/2/2
S001		2022/2/2

X

X

X

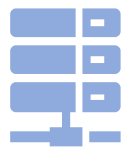


关系的完整性

■ 实体完整性（Entity Integrity）

● 关系模型必须遵守实体完整性规则的原因

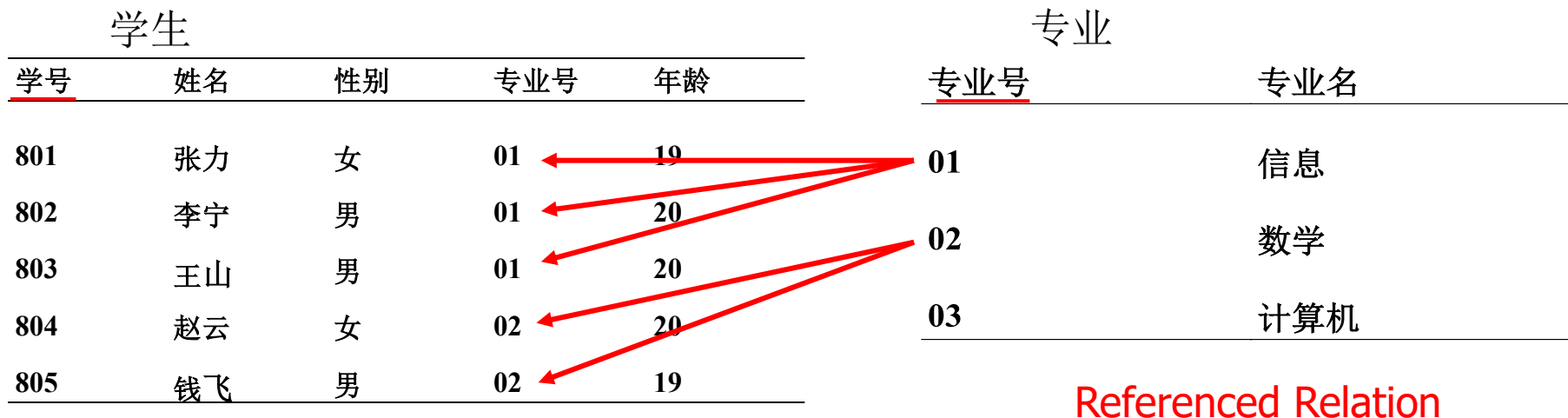
- ✓ 一个关系（二维表）通常对应现实世界的一个实体集或多对多联系。表中每条记录对应一个实体或联系。
- ✓ 现实世界中的实体和实体间的联系都是可区分的。
- ✓ 因此，关系中应设定主键来作为记录的唯一性标识，以此区分实体或联系
- ✓ 同时主键中的属性即主属性不能取空值。
 - ✓ 空值就是“不知道”或“无意义”的值。
 - ✓ 主属性取空值，就说明存在某个不可标识的实体，即存在不可区分的实体，这与上面的第二条论断相违背。



关系的完整性

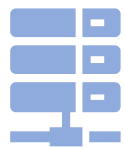
■ 参照完整性（Referential Integrity）

- 在关系模型中实体及实体间的联系都是用关系来描述的，因此可能存在着关系与关系间的引用（reference）。



Referencing Relation

Referenced Relation



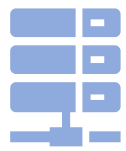
关系的完整性

■ 参照完整性（Referential Integrity）

- 在关系模型中实体及实体间的联系都是用关系来描述的，因此可能存在着关系与关系间的引用（reference）。

学生

<u>学号</u>	姓名	性别	专业号	年龄	班长
801	张力	女	01	19	802
802	李宁	男	01	20	
803	王山	男	01	20	802
804	赵云	女	02	20	805
805	钱飞	男	02	19	



关系的完整性

■ 参照完整性（Referential Integrity）

- 关系模型中用**外键（Foreign Key）**来表示这种关系与关系之间的引用。

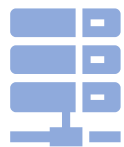
设F是关系R的一个或一组属性，但不是关系R的键。

如果F与关系S的主键K相对应，则称F是关系R的**外键**

- ✓ 关系R称为参照关系（Referencing Relation）；
- ✓ 关系S称为被参照关系（Referenced Relation）或目标关系（Target Relation）。

● 参照完整性规则：

- 外键上的取值只能是空值，或等于被参照关系中某个元组的主键值



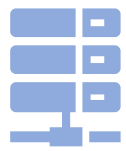
关系的完整性

■ 参照完整性（Referential Integrity）

学生关系中每个元组的“专业号”属性只取下面两类值：

- (1) 空值，表示尚未给该学生分配专业
- (2) 非空值，这时该值必须是专业关系中某个元组的“专业号”值，表示该学生不可能分配到一个不存在的专业中

学生					专业	
学号	姓名	性别	专业号	年龄	专业号	专业名
801	张力	女	01	19	01	信息
802	李宁	男	01	20	02	数学
803	王山	男	01	20	03	计算机
804	赵云	女	02	20		
805	钱飞	男	02	19		



关系的完整性

- 参照完整性 (Referential Integrity)
 - 关系R和S不一定是不同的关系
 - 目标关系S的主键K 和参照关系的外键F必须定义在同一个 (或一组) 域上
 - 外键并不一定要与相应的主键同名
 - ✓ 当外键与相应的主键属于不同关系时, 往往取相同的名字, 以便于识别



关系的完整性

■ 用户定义的完整性（ Referential Integrity）

- 用户定义的完整性是针对某一具体关系数据库的约束条件，反映某一具体应用所涉及的数据必须满足的语义要求。
- 关系模型应提供定义和检验这类完整性的机制，以使用统一的方法处理它们，而不要由应用程序承担这一功能。

用户定义完整性示例：

课程(课程号， 课程名， 学分， 选课人数)

“课程名” 属性必须取唯一值

非主属性“课程名” 也不能取空值

“学分” 属性只能取值{1, 2, 3, 4}



本章习题

- 课本第70页：1、2、3、4、5题
- ER转换关系表作业.doc
- 提交时间：3月18日前