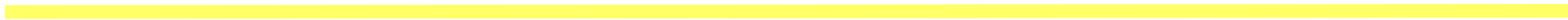


# 数据库系统概论






## An Introduction to Database Systems

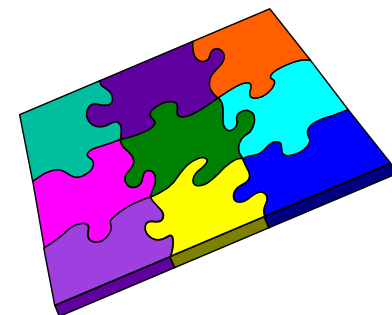
### 第六章 关系数据理论



# 第六章 关系数据理论

## 本章主要内容

-  问题的提出
-  规范化
-  数据依赖的公理系统
-  模式的分解
-  小结



➡ 例： 有三个属性的工资表（姓名,级别,工资）关系模式。  
对应此模式建立的表如表6-1所示。

姓 名	级 别	工 资
A	10	650
B	10	650
C	7	680
D	8	665
E	11	630
F	11	630

表 6-1

表 6 — 1 存在的问题：

➤ 数据冗余度大

✓ 浪费存储空间

✓ 造成数据的不一致

➤ 插入与删除异常

✓ 无法插入某部分信息

✓ 删除掉不应删除的信息

#### ➔ 解决方法

■ 将表6-1分解为两个模式表达：职工级别（姓名,级别），级别工资（级别,工资), 如表 6-2、表 6-3所示。

姓名	级别
A	10
B	10
C	7
D	8
E	11
F	11

表 6-2

级别	工资
7	680
8	665
9	660
10	650
11	630

表 6-3

改进后,

- ✓ 数据量减少
- ✓ 表达能力强
- ✓ 修改方便

### ➡ 关系数据库逻辑设计

- ▮ 针对具体问题，如何构造一个适合于它的数据模式
- ▮ 数据库逻辑设计的工具——关系数据库的规范化理论

➤ 关系模式由五部分组成，即它是一个五元组：

**$R(U, D, DOM, F)$**

**R:** 关系名

**U:** 组成该关系的属性名集合

**D:** 属性组U中属性所来自的域

**DOM:** 属性向域的映象集合

**F:** 属性间数据的依赖关系集合

➤ 关系模式  **$R(U, D, DOM, F)$**

简化为一个三元组：

**$R(U, F)$**

当且仅当U上的一个关系r  
满足F时，r称为关系模式  
 **$R(U, F)$**  的一个关系

什么是数据依赖？

- 定义属性值间的相互关连（主要体现于值的相等与否）
- 数据库模式设计的关键
- 一个关系内部属性与属性之间的约束关系
- 现实世界属性间相互联系的抽象
- 数据内在的性质
- 语义的体现
- 数据依赖的类型

函数依赖 (FD)

多值依赖 (MVF)

[例1]建立一个描述学校教务的数据库:

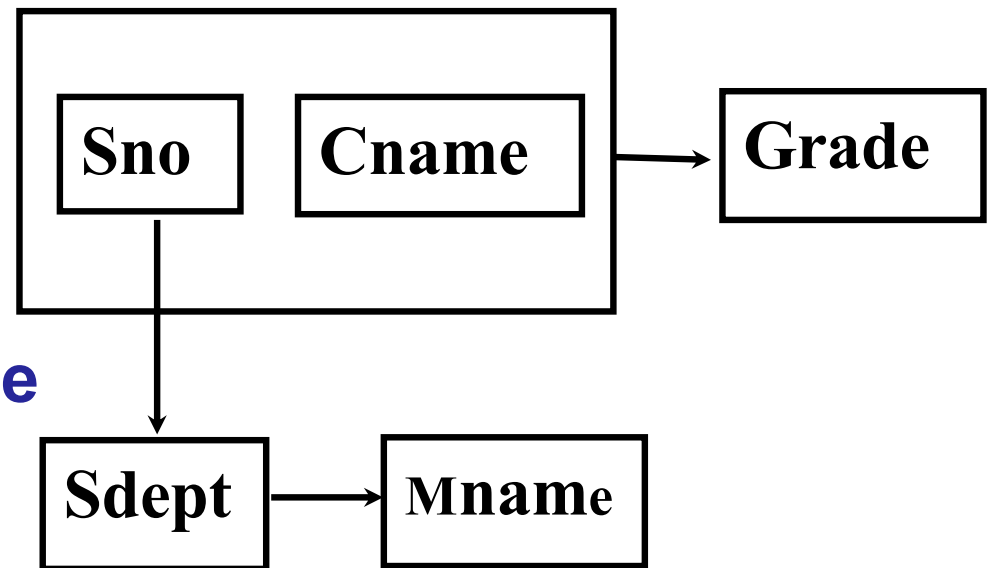
学生的学号 (**Sno**)、所在系 (**Sdept**)、系主任姓名 (**Mname**)、课程名 (**Cname**)、成绩 (**Grade**)

单一的关系模式: **Student <U、F>**

**$U = \{ Sno, Sdept, Mname, Cname, Grade \}$**

属性组**U**上的一组函数依赖**F**:

**$F = \{ Sno \rightarrow Sdept,$   
 **$Sdept \rightarrow Mname,$**   
 **$(Sno, Cname) \rightarrow Grade$**   
 **$\}$****



➔ 关系模式 **Student** $\langle U, F \rangle$  其中, ( $U = \{ Sno, Sdept, Mname, Cname, Grade \}$ ) 中存在的问题

- 数据冗余太大
- 更新异常 (Update Anomalies)
- 插入异常 (Insertion Anomalies)
- 删除异常 (Deletion Anomalies)

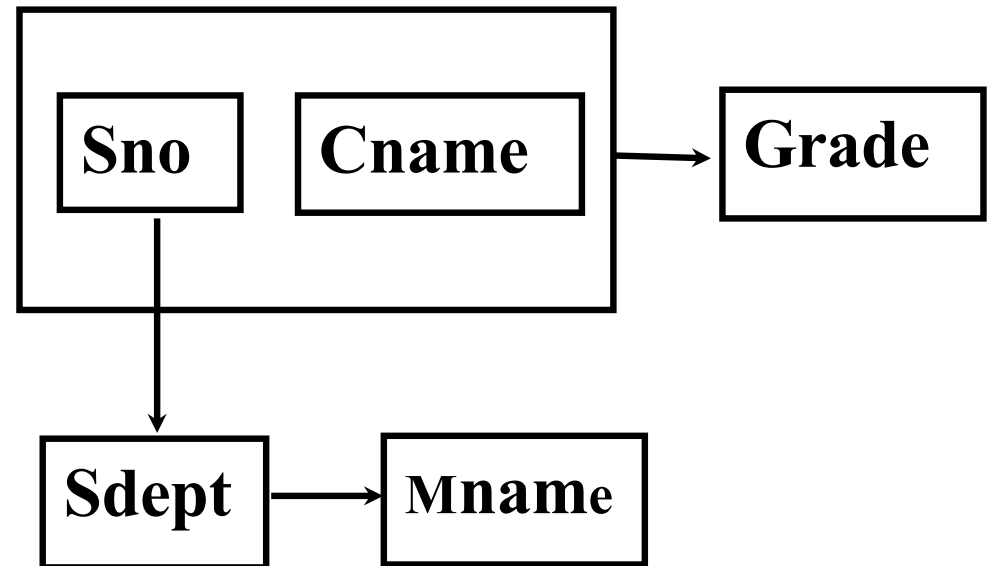
**Student**  
不是一个  
好的  
关系模式

**“好”的模式:**

不会发生插入异常、删除异常、更新异常,  
数据冗余应尽可能少



- ➡ **原因：** 由存在于模式中的某些数据依赖引起的
- ➡ **解决方法：** 通过分解关系模式来消除其中不合适的数据依赖
- ➡ 分解**Student**关系模式



把这个单一模式分成**3**个关系模式：

**S (Sno, Sdept, Sno → Sdept) ;**

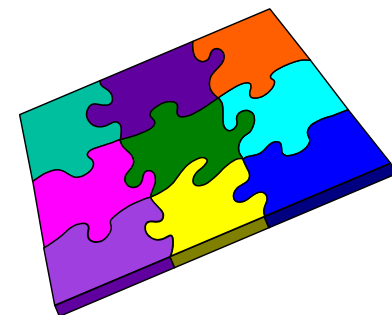
**SC (Sno, Cname, Grade, (Sno, Cno) → Grade) ;**

**DEPT (Sdept, Mname, Sdept→ Mname)**

# 第六章 关系数据理论

## 本章主要内容

- 问题的提出
- 规范化
- 数据依赖的公理系统
- 模式的分解
- 小结



规范化理论正是用来改造关系模式，通过分解关系模式来消除其中不合适的数据依赖，以解决插入异常、删除异常、更新异常和数据冗余问题。



## 6.2 规范化

6.2.1

函数依赖

6.2.2

码

6.2.3

范式

6.2.4

2NF

6.2.5

3NF

6.2.6

BCNF

6.2.7

多值依赖

6.2.8

4NF

6.2.9

规范化小结

➡ **定义6.1** 设 $R(U)$ 是一个属性集 $U$ 上的关系模式， $X$ 和 $Y$ 是 $U$ 的子集。

若对于 $R(U)$ 的任意一个可能的关系 $r$ ， $r$ 中不可能存在两个元组在 $X$ 上的属性值相等，而在 $Y$ 上的属性值不等，则称“ $X$ 函数确定 $Y$ ”或“ $Y$ 函数依赖于 $X$ ”，记作 $X \rightarrow Y$ 。

➡ 函数依赖说明

- ▣ 所有关系实例均要满足
- ▣ 语义范畴的概念
- ▣ 数据库设计者可以对现实世界作强制的规定

➡ 在关系模式  $R(U)$  中，对于  $U$  的子集  $X$  和  $Y$ ，  
如果  $X \rightarrow Y$ ，但  $Y \subsetneq X$ ，则称  $X \rightarrow Y$  是非平凡的函数依赖  
若  $X \rightarrow Y$ ，但  $Y \subseteq X$ ，则称  $X \rightarrow Y$  是平凡的函数依赖

例：在关系  $SC(Sno, Cno, Grade)$  中，

非平凡函数依赖：  $(Sno, Cno) \rightarrow Grade$

平凡函数依赖：  $(Sno, Cno) \rightarrow Sno$ 、 $(Sno, Cno) \rightarrow Cno$

- ▮ 若  $X \rightarrow Y$ ，则  $X$  称为这个函数依赖的决定属性组，也称为决定因素（**Determinant**）。
- ▮ 若  $X \rightarrow Y$ ， $Y \rightarrow X$ ，则记作  $X \longleftrightarrow Y$ 。
- ▮ 若  $Y$  不函数依赖于  $X$ ，则记作  $X \nrightarrow Y$ 。

⇒ **定义6.2** 在 $R(U)$ 中,

▮ 如果 $X \rightarrow Y$ , 并且对于 $X$ 的任何一个真子集 $X'$ , 都有 $X' \not\rightarrow Y$ , 则称 $Y$ 对 $X$ **完全函数依赖**, 记作 $X \xrightarrow{F} Y$ 。

▮ 若 $X \rightarrow Y$ , 但 $Y$ 不完全函数依赖于 $X$ , 则称 $Y$ 对 $X$ **部分函数依赖**, 记作 $X \xrightarrow{P} Y$ 。

[例1] 中 $(Sno, Cno) \rightarrow Grade$ ,  $Sno \rightarrow Sdept$ 成立, 是完全函数依赖,

而  $(Sno, Cno) \rightarrow Sdept$ 是部分函数依赖

➡ **定义6.3** 在 $R(U)$ 中, 如果 $X \rightarrow Y$ ,  $(Y \not\subseteq X)$ ,  $Y \not\rightarrow X$ ,  $Y \rightarrow Z$ , 则称 $Z$ 对 $X$ **传递函数依赖**。

记为:  $X \xrightarrow{\text{传递}} Z$

注: 如果 $Y \rightarrow X$ , 即 $X \longleftrightarrow Y$ , 则 $Z$ **直接依赖于** $X$ 。

例: 在关系 $Std(Sno, Sdept, Mname)$ 中, 有:

$Sno \rightarrow Sdept, Sdept \rightarrow Mname$

$Mname$ 传递函数依赖于 $Sno$



## 6.2 规范化

6.2.1

函数依赖

6.2.2

码

6.2.3

范式

6.2.4

2NF

6.2.5

3NF

6.2.6

BCNF

6.2.7

多值依赖

6.2.8

4NF

6.2.9

规范化小结

若关系中的某一属性组的值能唯一地标识一个元组，则称该属性组为候选码

➡ **定义6.4** 设 $K$ 为 $R\langle U, F \rangle$ 中的属性或属性组合。

若 $K \xrightarrow{F} U$ ，则 $K$ 称为 $R$ 的**侯选码**（**Candidate Key**）。

若候选码多于一个，则选定其中的一个做为**主码**（**Primary Key**）。

### ➡ 主属性与非主属性

☐ 包含在任何一个候选码中的属性，称为**主属性**（**Prime attribute**）

☐ 不包含在任何码中的属性称为**非主属性**（**Nonprime attribute**）或非码属性（**Non-key attribute**）

### ➡ 全码

☐ 整个属性组是码，称为**全码**（**All-key**）

### [例2]

关系模式**S(Sno, Sdept, Sage)**, 单个属性**Sno**是码,  
**SC (Sno, Cno, Grade)** 中, (**Sno, Cno**) 是码

### [例3]

关系模式**R (P, W, A)**

**P**: 演奏者    **W**: 作品    **A**: 听众

- ❑ 一个演奏者可以演奏多个作品
- ❑ 某一作品可被多个演奏者演奏
- ❑ 听众可以欣赏不同演奏者的不同作品

码为**(P, W, A)**, 即**All-Key**

➡ **定义6.5** 关系模式  $R$  中属性或属性组  $X$  并非  $R$  的码，但  $X$  是另一个关系模式的码，则称  $X$  是  $R$  的**外部码**

(**Foreign key**) 也称外码

如在  $SC$  ( $Sno$ ,  $Cno$ ,  $Grade$ ) 中，

$Sno$  不是码，但  $Sno$  是关系模式  $S$  ( $Sno$ ,  $Sdept$ ,  $Sage$ ) 的码，则  $Sno$  是关系模式  $SC$  的外部码

➡ 主码与外部码一起提供了表示关系间联系的手段

## 6.2 规范化

6.2.1

函数依赖

6.2.2

码

6.2.3

范式

6.2.4

2NF

6.2.5

3NF

6.2.6

BCNF

6.2.7

多值依赖

6.2.8

4NF

6.2.9

规范化小结

- ➡ **范式**是符合某一种级别的关系模式的集合
- ➡ 关系数据库中的关系必须满足一定的要求。满足不同程度要求的为不同范式
- ➡ 范式的种类及联系：

$$1NF \supset 2NF \supset 3NF \supset BCNF \supset 4NF \supset 5NF$$

- ➡ 某一关系模式**R**为第**n**范式，可简记为**R∈nNF**。
- ➡ 一个低一级范式的关系模式，通过**模式分解**可以转换为若干个高一级范式的关系模式的集合，这种过程就叫**规范化**



## 6.2 规范化

6.2.1

函数依赖

6.2.2

码

6.2.3

范式

6.2.4

**2NF**

6.2.5

**3NF**

6.2.6

**BCNF**

6.2.7

多值依赖

6.2.8

**4NF**

6.2.9

规范化小结



### ➡ 1NF的定义

如果一个关系模式**R**的所有属性都是不可分的基本数据项，则 **$R \in 1NF$**

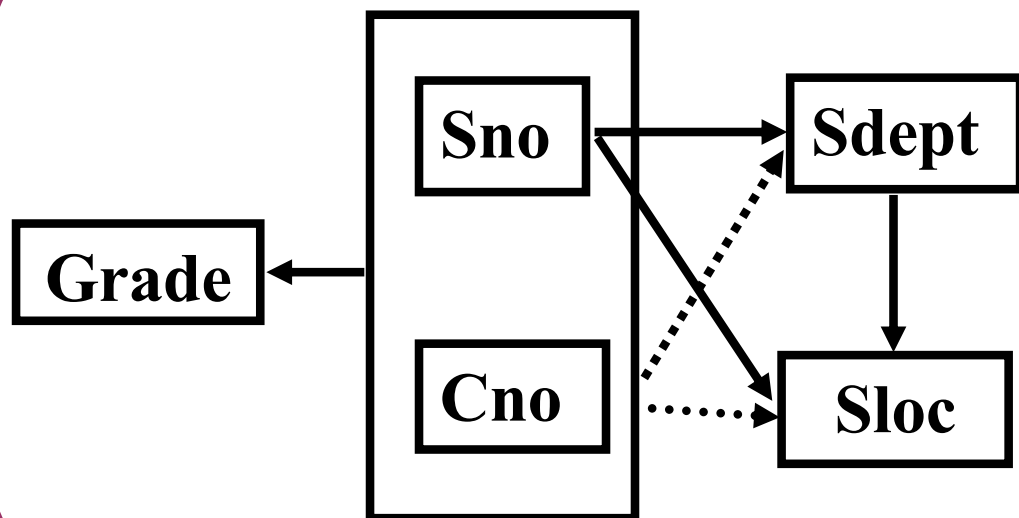
➡ 第一范式是对关系模式的最起码的要求。不满足第一范式的数据库模式不能称为关系数据库

➡ 但是满足第一范式的关系模式并不一定是一个好的关系模式

## 6.2.4 2NF

**[例4] 关系模式 S-L-C(Sno, Sdept, Sloc, Cno, Grade)**

**Sloc**为学生住处，假设每个系的学生住在同一个地方



函数依赖包括:

$(Sno, Cno) \xrightarrow{F} Grade$

$Sno \rightarrow Sdept$

$(Sno, Cno) \xrightarrow{P} Sdept$

$Sno \rightarrow Sloc$

$(Sno, Cno) \xrightarrow{P} Sloc$

$Sdept \rightarrow Sloc$

- S-L-C的码为(Sno, Cno)
- S-L-C满足第一范式。
- 非主属性Sdept和Sloc部分函数依赖于码(Sno, Cno)

➔ **S-L-C(Sno, Sdept, Sloc, Cno, Grade)**不是一个好的关系模式

❏ 插入异常

❏ 删除异常

❏ 数据冗余度大

❏ 修改复杂

函数依赖包括:

$(Sno, Cno) \xrightarrow{F} Grade$

$Sno \rightarrow Sdept$

$(Sno, Cno) \xrightarrow{P} Sdept$

$Sno \rightarrow Sloc$

$(Sno, Cno) \xrightarrow{P} Sloc$

$Sdept \rightarrow Sloc$

原因:

**Sdept、Sloc**部分函数依赖于码

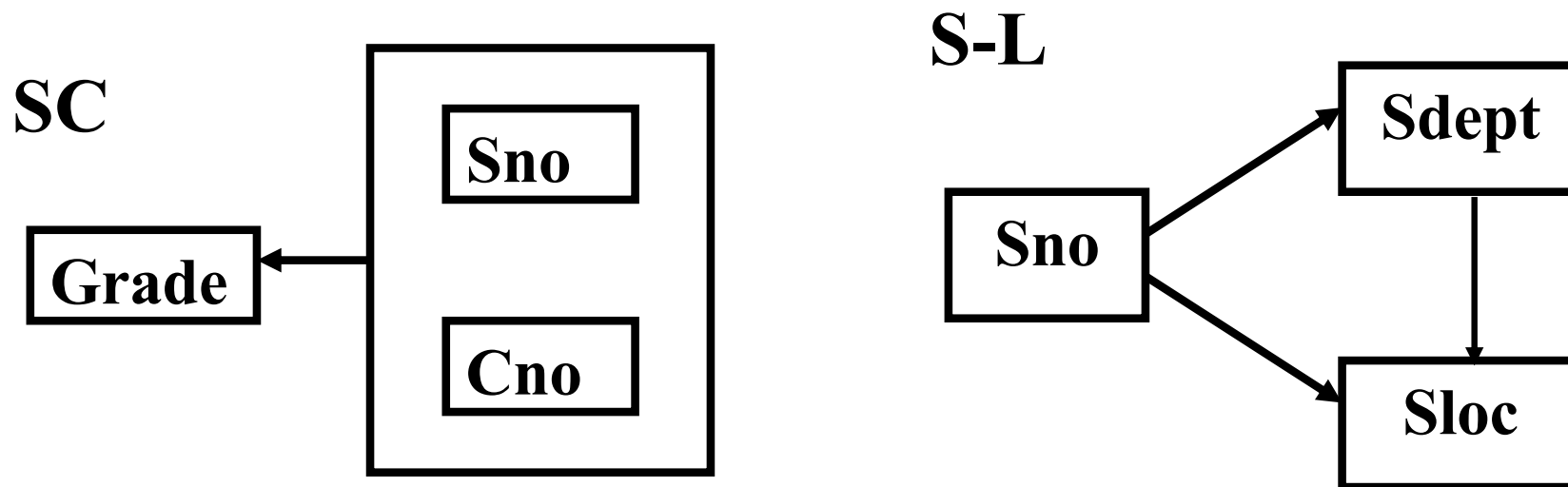
➔ 解决方法

**S-L-C**分解为两个关系模式，以消除这些部分函数依赖

**SC (Sno, Cno, Grade)**

**S-L (Sno, Sdept, Sloc)**

### ➡ 分解后的函数依赖图



- ❖ 关系模式SC的码为 (Sno, Cno)
- ❖ 关系模式S-L的码为Sno
- ❖ 这样非主属性对码都是完全函数依赖

## 6.2.4 2NF

### ➔ 2NF的定义

**定义6.6** 若 $R \in 1NF$ ，且每一个非主属性完全函数依赖于码，则 $R \in 2NF$ 。

**[说明]** 若一个数据库模式是二范式，则它的每一个关系模式都是二范式。

例：设 $U=\{S\#, C\#, SG, SD, CC, CN\}$ ， $U$ 上的函数依赖  
 $F=\{S\# \rightarrow SD, C\# \rightarrow CC, S\#, C\# \rightarrow SG, C\# \rightarrow CN, CN \rightarrow C\#\}$ 。

■ 令数据库模式 $R=\{R1, R2, R3\}$ ，其中

■  $R1=\{S\#, SD\}$

■  $R2=\{S\#, C\#, SG\}$

■  $R3=\{C\#, CN, CC\}$

- 可以看出， $R1$ 中 $S\#$ 是码， $SD$ 属性完全依赖于码，故 $R1$ 是二范式。
- 在 $R2$ 中， $S\#$ ， $C\#$ 是码，而非主属性 $SG$ 完全依赖于码。
- 在 $R3$ 中， $C\#$ 或 $CN$ 是码，而非主属性 $CC$ 完全依赖于码，故 $R1$ ， $R2$ ， $R3$ 均为二范式， $R$ 为二范式的数据库模式。

- ➡ 采用投影分解法将一个**1NF**的关系分解为多个**2NF**的关系，可以在一定程度上减轻原**1NF**关系中存在的插入异常、删除异常、数据冗余度大、修改复杂等问题。

例：S-L-C(Sno, Sdept, Sloc, Cno, Grade) ∈ 1NF

S-L-C(Sno, Sdept, Sloc, Cno, Grade) ∉ 2NF

SC (Sno, Cno, Grade) ∈ 2NF

S-L (Sno, Sdept, Sloc) ∈ 2NF ?

- ➡ 将一个**1NF**关系分解为多个**2NF**的关系，并不能完全消除关系模式中的各种异常情况和数据冗余。

## 6.2 规范化

6.2.1

函数依赖

6.2.2

码

6.2.3

范式

6.2.4

2NF

6.2.5

3NF

6.2.6

BCNF

6.2.7

多值依赖

6.2.8

4NF

6.2.9

规范化小结

### ➡ 3NF的定义

**定义6.7** 关系模式 $R\langle U, F \rangle$ 中若不存在这样的码 $X$ 、属性组 $Y$ 及非主属性 $Z$  ( $Z \not\subseteq Y$ ), 使得 $X \rightarrow Y$ ,  $Y \rightarrow Z$ 成立,  $Y \not\rightarrow X$ , 则称 $R\langle U, F \rangle \in 3NF$ 。

■ 若 $R \in 3NF$ , 则每一个非主属性既不部分依赖于码也不传递依赖于码。



例：2NF关系模式S-L(Sno, Sdept, Sloc)中

函数依赖：

$Sno \rightarrow Sdept$

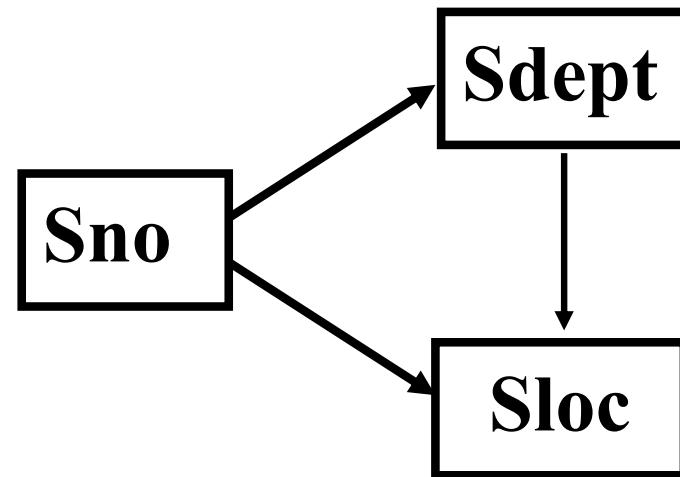
$Sdept \not\rightarrow Sno$

$Sdept \rightarrow Sloc$

可得：

$Sno \xrightarrow{\text{传递}} Sloc$ ，即S-L中存在非主属性对码的传递函数依赖， $S-L \notin 3NF$

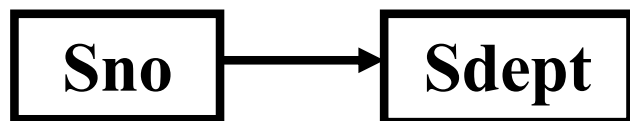
S-L函数依赖图：



### ➡ 解决方法

采用投影分解法，把**S-L**分解为两个关系模式，以消除传递函数依赖：

**S-D** (**Sno**, **Sdept**)



**D-L** (**Sdept**, **Sloc**)



▣ **S-D**的码为**Sno**， **D-L**的码为**Sdept**

▣ 分解后的关系模式**S-D**与**D-L**中不再存在传递依赖

❖ **S-L(Sno, Sdept, Sloc) ∈ 2NF**  
**S-L(Sno, Sdept, Sloc) ∉ 3NF**  
**S-D(Sno, Sdept) ∈ 3NF**  
**D-L(Sdept, Sloc) ∈ 3NF**

- ➡ 采用投影分解法将一个**2NF**的关系分解为多个**3NF**的关系，可以在一定程度上解决原**2NF**关系中存在的插入异常、删除异常、数据冗余度大、修改复杂等问题。
- ➡ 将一个**2NF**关系分解为多个**3NF**的关系后，仍然不能完全消除关系模式中的各种异常情况和数据冗余。

## 6.2.5 3NF

例：配件管理关系模式 **WPE** (**WNO**, **PNO**, **ENO**, **QNT**)

分别表示：仓库号，配件号，职工号，配件数量

➡ 有以下条件：

- a. 一个仓库有多个职工。
- b. 一个职工仅在一个仓库工作。
- c. 每个仓库里一种型号的配件由专人负责，但一个人可以管理几种配件。
- d. 同一种型号的配件可以分放在几个仓库中。

➡ 函数依赖

**ENO** ----> **WNO**, **WNO** ~~----~~ **ENO**  
(**WNO**, **PNO**) ----> **ENO**  
(**WNO**, **PNO**) ----> **QNT**  
(**ENO**, **PNO**) ----> **QNT**

## 6.2.5 3NF

例：配件管理关系模式 **WPE** (**WNO**, **PNO**, **ENO**, **QNT**)

分别表示： 仓库号，配件号，职工号，配件数量

➔ **Candidate key**: (**WNO**, **PNO**)、 (**ENO**, **PNO**)

▮ 主属性: **WNO**, **PNO**, **ENO**

▮ 非主属性: **QNT**

➔ **QNT**对两个候选码均是完全函数依赖及直接依赖，

▮ 所以，**WPE** ∈ 3NF

➔ 但是，还是有问题？

➔ 分解成：

▮ 管理: **EP** (**PNO**, **ENO**, **QNT**)

▮ 工作: **EW** (**ENO**, **WNO**)

缺点：

分解后函数依赖的保持性较差。

**ENO** ----> **WNO**,

(**WNO**, **PNO**) ----> **ENO**

(**WNO**, **PNO**) ----> **QNT**

(**ENO**, **PNO**) ----> **QNT**

## 6.2 规范化

6.2.1

函数依赖

6.2.2

码

6.2.3

范式

6.2.4

2NF

6.2.5

3NF

6.2.6

**BCNF**

6.2.7

多值依赖

6.2.8

4NF

6.2.9

规范化小结

## 6.2.6 BC范式 (BCNF)

➡ **定义6.8** 关系模式 $R\langle U, F \rangle \in 1NF$ , 若 $X \rightarrow Y$ 且 $Y \subsetneq X$ 时 $X$ 必含有码, 则 $R\langle U, F \rangle \in BCNF$ 。

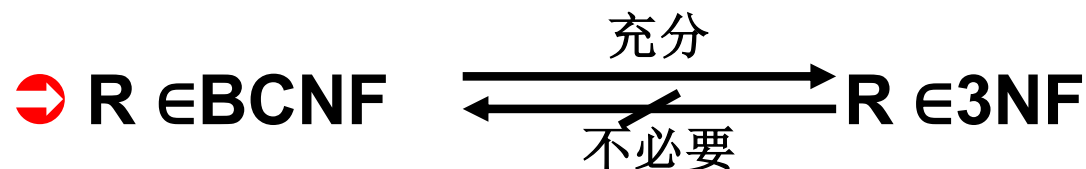
➡ 等价于: 每一个决定属性因素都包含码

➡ 若 $R \in BCNF$

▣ 所有非主属性对每一个码都是完全函数依赖

▣ 所有的主属性对每一个不包含它的码, 也是完全函数依赖

▣ 没有任何属性完全函数依赖于非码的任何一组属性



[例5] 关系模式C (Cno, Cname, Pcno)

■  $C \in 3NF$

■  $C \in BCNF$

[例6] 关系模式S (Sno, Sname, Sdept, Sage)

■ 假定S有两个码Sno, Sname

■  $S \in 3NF$ 。

■  $S \in BCNF$



[例7] 关系模式**SJP** (**S**, **J**, **P**)

**S**:学生, **J**:课程, **P**: 名次

- 函数依赖:  $(S, J) \rightarrow P; (J, P) \rightarrow S$
- $(S, J)$  与  $(J, P)$  都可以作为候选码,属性相交
- $SJP \in 3NF$ ,
- $SJP \in BCNF$

## 6.2.6 BC范式 (BCNF)

[例8]在关系模式**STJ** (**S**, **T**, **J**) 中, **S**表示学生, **T**表示教师, **J**表示课程。

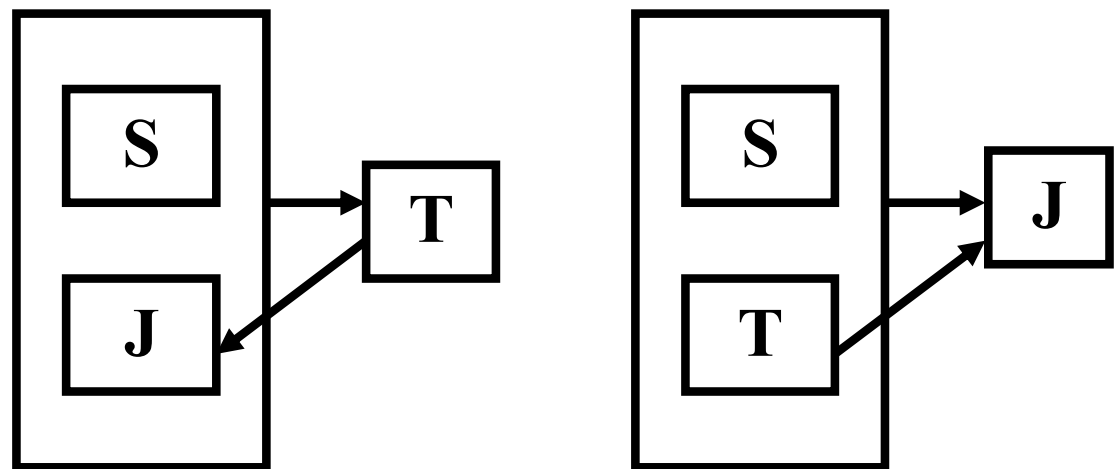
☐ 函数依赖:

$(S, J) \rightarrow T$ ,

$(S, T) \rightarrow J$ ,

$T \rightarrow J$

STJ中的函数依赖



☐  $(S, J)$ 和 $(S, T)$ 都是候选码

➡ **STJ** ∈ 3NF

☐ 没有任何非主属性对码传递依赖或部分依赖

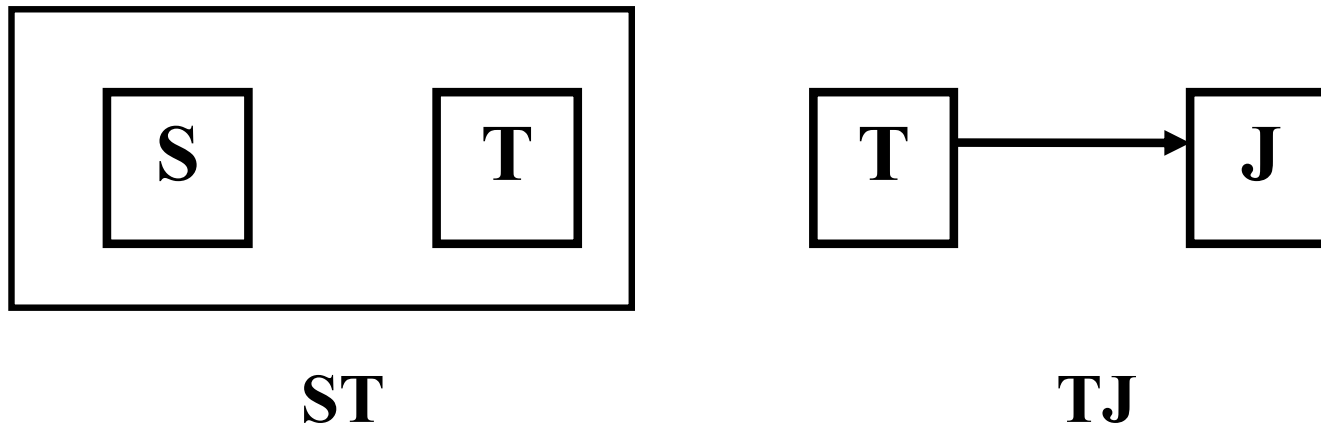
## 6.2.6 BC范式 (BCNF)

➔ ~~STJ~~ ∈ BCNF

☞ T是决定因素，T不包含码

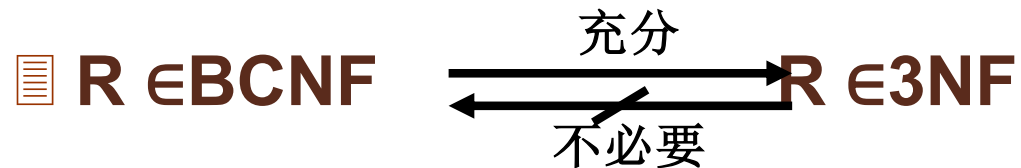
➔ 解决方法：将STJ分解为二个关系模式：

$ST(S, T) \in BCNF, TJ(T, J) \in BCNF$

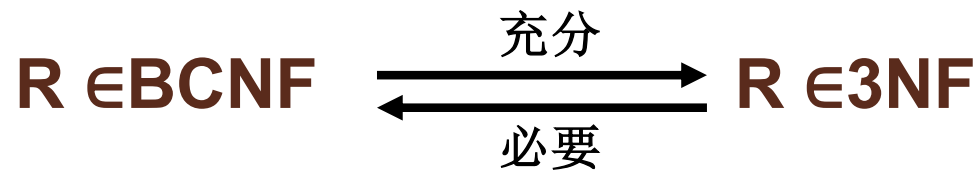


没有任何属性对码的部分函数依赖和传递函数依赖

### ➡ 3NF与BCNF的关系



☞ 如果 $R \in \text{3NF}$ ，且 $R$ 只有一个候选码



➡ 3NF与BCNF是以函数依赖为基础的关系模式规范化程度的测度。



## 6.2 规范化

6.2.1

函数依赖

6.2.2

码

6.2.3

范式

6.2.4

2NF

6.2.5

3NF

6.2.6

BCNF

6.2.7

多值依赖

6.2.8

4NF

6.2.9

规范化小结

## 6.2.7 多值依赖

**[例9]** 学校中某一门课程由多个教师讲授，他们使用相同的一套参考书。

每个教员可以讲授多门课程，每种参考书可以供多门课程使用。

非  
规  
范  
化  
关  
系

课 程 C	教 员 T	参 考 书 B
物理	$\left\{ \begin{array}{l} \text{李 勇} \\ \text{王 军} \end{array} \right\}$	$\left\{ \begin{array}{l} \text{普通物理学} \\ \text{光学原理} \\ \text{物理习题集} \end{array} \right\}$
数学	$\left\{ \begin{array}{l} \text{李 勇} \\ \text{张 平} \end{array} \right\}$	$\left\{ \begin{array}{l} \text{数学分析} \\ \text{微分方程} \\ \text{高等代数} \end{array} \right\}$
计算数学	$\left\{ \begin{array}{l} \text{张 平} \\ \text{周 峰} \end{array} \right\}$	$\left\{ \begin{array}{l} \text{数学分析} \\ \dots \\ \dots \end{array} \right\}$
$\vdots$	$\vdots$	$\vdots$

## 6.2.7 多值依赖

### ❖ 用二维表表示Teaching

课程C	教员T	参考书B
物理	李勇	普通物理学
物理	李勇	光学原理
物理	李勇	物理习题集
物理	王军	普通物理学
物理	王军	光学原理
物理	王军	物理习题集
数学	李勇	数学分析
数学	李勇	微分方程
数学	李勇	高等代数
数学	张平	数学分析
数学	张平	微分方程
数学	张平	高等代数
...	...	...

➔ Teaching ∈ BCNF

➔ Teaching 具有唯一候选码 (C, T, B), 即全码

➔ Teaching 模式中存在的问题

❑ 数据冗余度大

❑ 插入操作复杂

❑ 删除操作复杂

❑ 修改操作复杂

存在多值依赖

### ⇒ 定义6.9

设 $R(U)$ 是一个属性集 $U$ 上的一个关系模式， $X$ 、 $Y$ 和 $Z$ 是 $U$ 的子集，并且 $Z = U - X - Y$ 。关系模式 $R(U)$ 中多值依赖 $X \twoheadrightarrow Y$ 成立，当且仅当对 $R(U)$ 的任一关系 $r$ ，给定的一对 $(x, z)$ 值，有一组 $Y$ 的值，这组值仅仅决定于 $x$ 值而与 $z$ 值无关

### ⇒ 例 Teaching (C, T, B)



## 6.2.7 多值依赖

➡ 多值依赖的另一个等价的形式化的定义：

在 $R(U)$ 的任一关系 $r$ 中，如果存在元组 $t, s$ 使得 $t[X]=s[X]$ ，那么就必然存在元组 $w, v \in r$ ，（ $w, v$ 可以与 $s, t$ 相同），使得 $w[X]=v[X]=t[X]$ ，而 $w[Y]=t[Y]$ ， $w[Z]=s[Z]$ ， $v[Y]=s[Y]$ ， $v[Z]=t[Z]$ （即交换 $s, t$ 元组的 $Y$ 值所得的两个新元组必在 $r$ 中），则 $Y$ 多值依赖于 $X$ ，记为 $X \twoheadrightarrow Y$ 。这里， $X, Y$ 是 $U$ 的子集， $Z=U-X-Y$ 。

➡ 平凡多值依赖和非平凡的多值依赖

☐ 若 $X \twoheadrightarrow Y$ ，而 $Z=\varphi$ ，则称 $X \twoheadrightarrow Y$ 为平凡的多值依赖

☐ 否则称 $X \twoheadrightarrow Y$ 为非平凡的多值依赖

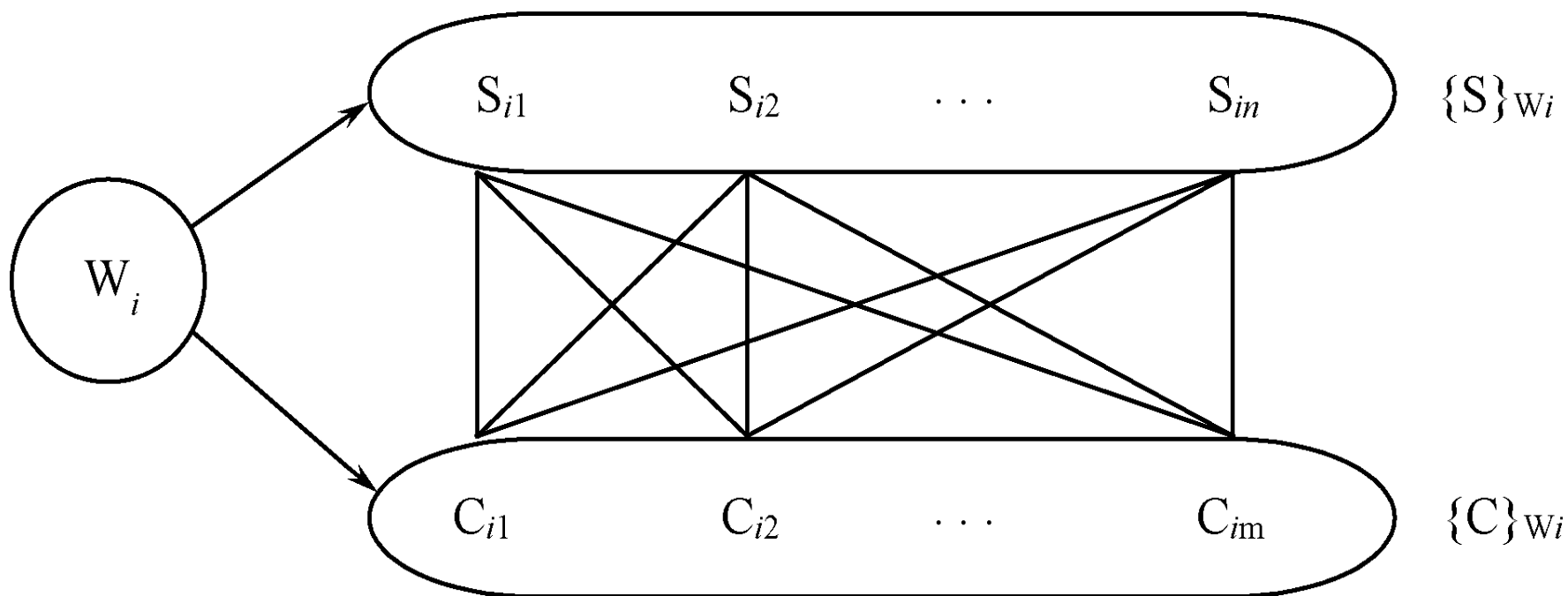
## 6.2.7 多值依赖

### [例10] 关系模式WSC (W, S, C)

- W表示仓库，S表示保管员，C表示商品
- 假设每个仓库有若干个保管员，有若干种商品
- 每个保管员保管所在的仓库的所有商品
- 每种商品被所有保管员保管

W	S	C
W1	S1	C1
W1	S1	C2
W1	S1	C3
W1	S2	C1
W1	S2	C2
W1	S2	C3
W2	S3	C4
W2	S3	C5
W2	S4	C4
W2	S4	C5

用下图表示这种对应



$$W \twoheadrightarrow S \text{ 且 } W \twoheadrightarrow C$$

➡ 多值依赖具有对称性

若 $X \twoheadrightarrow Y$ ，则 $X \twoheadrightarrow Z$ ，其中 $Z = U - X - Y$

➡ 多值依赖具有传递性

若 $X \twoheadrightarrow Y$ ， $Y \twoheadrightarrow Z$ ，则 $X \twoheadrightarrow Z - Y$

➡ 函数依赖是多值依赖的特殊情况：

若 $X \rightarrow Y$ ，则 $X \twoheadrightarrow Y$ 。

➡ 若 $X \twoheadrightarrow Y$ ， $X \twoheadrightarrow Z$ ，则 $X \twoheadrightarrow Y \cup Z$ 。

➡ 若 $X \twoheadrightarrow Y$ ， $X \twoheadrightarrow Z$ ，则 $X \twoheadrightarrow Y \cap Z$ 。

➡ 若 $X \twoheadrightarrow Y$ ， $X \twoheadrightarrow Z$ ，则 $X \twoheadrightarrow Y - Z$ ， $X \twoheadrightarrow Z - Y$ 。

### ➡ 多值依赖与函数依赖的区别

☞ 多值依赖的有效性与属性集的范围有关

- 若 $X \twoheadrightarrow Y$ ，在 $U$ 上成立，则在 $W(XY \subseteq W \subseteq U)$ 上一定成立。但 $X \twoheadrightarrow Y$ ，在 $W(W \subseteq U)$ 上成立，在 $U$ 上并不一定成立。（这是因为多值依赖的定义中不仅涉及到 $X$ 和 $Y$ ，而且涉及到 $U$ 中的其余属性 $Z$ ）

☞ 若函数依赖 $X \rightarrow Y$ 在 $R(U)$ 上成立，则对于任何 $Y' \subset Y$ 均有 $X \rightarrow Y'$ 成立；多值依赖 $X \twoheadrightarrow Y$ 若在 $R(U)$ 上成立，不能断言对于任何 $Y' \subset Y$ 有 $X \twoheadrightarrow Y'$ 成立

## 6.2 规范化

6.2.1

函数依赖

6.2.2

码

6.2.3

范式

6.2.4

2NF

6.2.5

3NF

6.2.6

BCNF

6.2.7

多值依赖

6.2.8

4NF

6.2.9

规范化小结

⇒ **定义6.10** 关系模式 $R\langle U, F \rangle \in 1NF$ ，如果对于 $R$ 的每个非平凡多值依赖 $X \twoheadrightarrow Y$  ( $Y \subseteq X$ )， $X$ 都含有码，则 $R \in 4NF$ 。

⇒ 如果 $R \in 4NF$ ， 则 $R \in BCNF$

- 不允许有非平凡且非函数依赖的多值依赖
- 允许的非平凡多值依赖是函数依赖
- 允许的多值依赖是平凡的多值依赖

例: **Teaching(C,T,B)  $\notin$  4NF**

存在非平凡的多值依赖 **$C \twoheadrightarrow T$** , 且**C**不是码

■ 用投影分解法把**Teaching**分解为如下两个关系模式:

**CT(C, T)  $\in$  4NF**

**CB(C, B)  $\in$  4NF**

**$C \twoheadrightarrow T$ ,  $C \twoheadrightarrow B$** 是平凡多值依赖



## 6.2 规范化

6.2.1

函数依赖

6.2.2

码

6.2.3

范式

6.2.4

2NF

6.2.5

3NF

6.2.6

BCNF

6.2.7

多值依赖

6.2.8

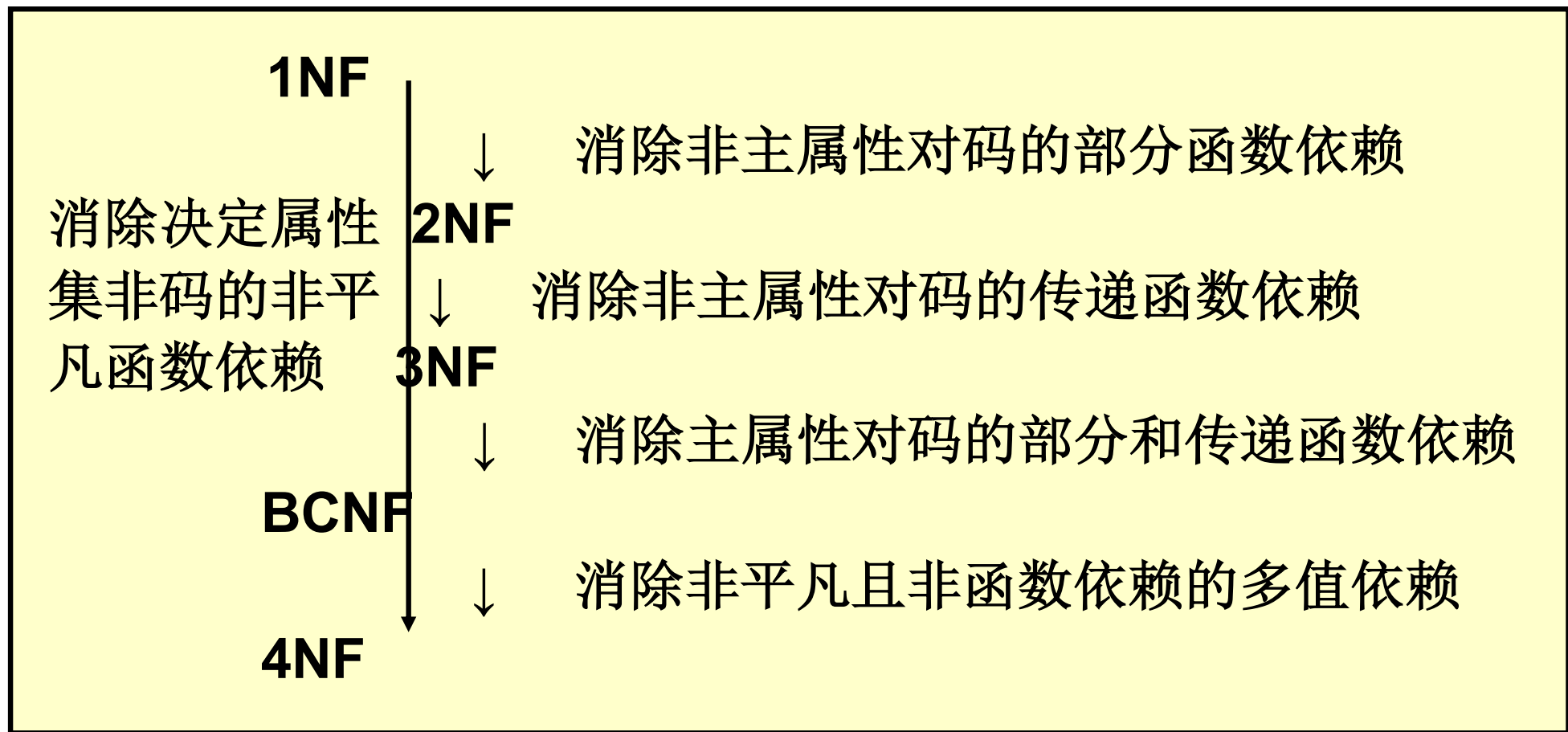
4NF

6.2.9

规范化小结

- ➡ 关系数据库的规范化理论是数据库逻辑设计的工具
- ➡ 目的：尽量消除插入、删除异常，修改复杂，数据冗余
- ➡ 基本思想：逐步消除数据依赖中不合适的部分
  - ▮ 实质：概念的单一化

### ➡ 关系模式规范化的基本步骤

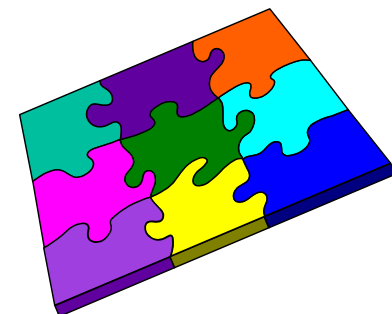


- ➡ 不能说规范化程度越高的关系模式就越好
- ➡ 在设计数据库模式结构时，必须对现实世界的实际情况和用户应用需求作进一步分析，确定一个合适的、能够反映现实世界的模式
- ➡ 上面的规范化步骤可以在其中任何一步终止

# 第六章 关系数据理论

## 本章主要内容

- 问题的提出
- 规范化
- 数据依赖的公理系统
- 模式的分解
- 小结



- ➡ 把低一级的关系模式分解为若干个高一级的关系模式的方法不是唯一的
- ➡ 只有能够保证分解后的关系模式与原关系模式等价，分解方法才有意义

### ➡ 模式分解的标准

☞ 三种模式分解等价的定义：

- 1. 分解具有无损连接性
- 2. 分解要保持函数依赖
- 3. 分解既要保持函数依赖，又要具有无损连接性

**例：**  $SL(Sno, Sdept, Sloc)$ ,

其中 **Sno** 是学号，**Sdept** 是学生所在系，**Sloc** 是学生住处，每个系的学生住在同一个地方。

$F = \{Sno \rightarrow Sdept, Sno \rightarrow Sloc, Sdept \rightarrow Sloc\}$

∴ 存在传递函数依赖，∴  $SL \in 2NF$ 。



## 6.4 模式的分解

➡ 第一种分解: **SN(Sno)**, **SC(Sdept)**, **SO(Sloc)**

➡ 分解后都是规范化程度很高的关系模式, 但分解后丢失了许多信息。

➡ 此种分解方法不可取。要使分解有意义, 起码的要求是后者不能丢失前者的信息。

Sno	Sdept	Sloc
98001	CS	A
98002	IS	B
98003	MA	C
98004	IS	B
98005	PH	B

➡ 若分解后的关系可以通过自然连接恢复为原来的关系, 则这种分解就没有丢失信息。

## 6.4 模式的分解

➡ 第二种分解: **NL(Sno, Sloc), DL(Sdept, Sloc)**

NL		DL					
Sno	Sloc	Sdept	Sloc		Sno	Sloc	Sdept
98001	A	CS	A	NL ⋈ DL ⇒	98001	A	CS
98002	B	IS	B		98002	B	IS
98003	C	MA	C		98002	B	PH
98004	B	IS	B		98003	C	MA
98005	B	PH	B		98004	B	IS
					98004	B	PH
					98005	B	IS
					98005	B	PH

连接后多了三个元组，将无法知道原来关系中究竟有哪些元组，从这个意义上讲，这个分解仍然丢失了信息。

## 6.4 模式的分解

➡ 第三种分解: **ND(Sno, Sdept), NL(Sno, Sloc)**

ND		NL		ND⋈NL		
Sno	Sdept	Sno	Sloc		Sdept	Sloc
98001	CS	98001	A	⇒	98001	A
98002	IS	98002	B		98002	B
98003	MA	98003	C		98003	C
98004	IS	98004	B		98004	B
98005	PH	98005	B		98005	B

没有丢失信息，恢复为原来的关系，这种分解称为：  
“无损连接分解”。

**定义6.1** 设关系模式 $R(U, F)$ 被分解为若干个关系模式 $R_1(U_1, F_1)$ ,  $R_2(U_2, F_2)$ , ...,  $R_n(U_n, F_n)$  (其中 $U = U_1 \cup U_2 \cup \dots \cup U_n$ , 且不存在 $U_i \subseteq U_j$ ,  $F_i$ 为 $F$ 在 $U_i$ 上的投影), 若 $R$ 与 $R_1, \dots, R_n$ 自然连接的结果相等, 则称 $R$ 的这个分解具有无损连接性。

➡ 只有具有无损连接性的分解才能保证不丢失信息

- ➡ 第三种分解虽然具有无损连接性，保证了不丢失原关系中的信息，但它并没有解决各种异常现象。如：**98001**转系，则两个关系中相应的元组必须同时修改。
- ➡ 因为这种分解没有保证原关系中的函数依赖，**SL**中的**Sdept**→**Sloc**即没有投影到**ND**上，也没有投影到**NL**上，而是跨在两个关系上。

**定义6.2** 设关系模式 $R(U, F)$ 被分解为若干个关系模式 $R_1(U_1, F_1)$ ,  $R_2(U_2, F_2)$ , ...,  $R_n(U_n, F_n)$  (其中 $U = U_1 \cup U_2 \cup \dots \cup U_n$ , 且不存在 $U_i \subseteq U_j$ ,  $F_i$ 为 $F$ 在 $U_i$ 上的投影), 若 $F$ 所逻辑蕴涵的函数依赖一定也由分解得到的某个关系模式中的函数依赖 $F_i$ 所逻辑蕴涵, 则称关系模式 $R$ 的这个分解是保持函数依赖的。

➡ **第四种分解: ND(Sno, Sdept), DL(Sdept, Sloc)**  
这种分解保持了函数依赖。

## 6.4 模式的分解

例: **S-L (Sno, Sdept, Sloc)**

**$F = \{ Sno \rightarrow Sdept, Sdept \rightarrow Sloc, Sno \rightarrow Sloc \}$**

**$S-L \in 2NF$**

分解方法可以有多种:

1. 分解为三个关系模式:

**SN(Sno)、SD(Sdept)、SO(Sloc)**

2. 分解为下面二个关系模式:

**NL(Sno, Sloc)、DL(Sdept, Sloc)**

3. 分解为下面二个关系模式:

**ND(Sno, Sdept)、NL(Sno, Sloc)**

4. 分解为下面二个关系模式:

**ND(Sno, Sdept)、DL(Sdept, Sloc)**

- ➡ 1 既不具有无损连接性, 也未保持函数依赖, 它不是原关系模式的一个等价分解
- ➡ 2 保持了函数依赖, 但不具有无损连接性
- ➡ 3 具有无损连接性, 但未持函数依赖
- ➡ 4 既具有无损连接性, 又保持了函数依赖

## 6.4 模式的分解

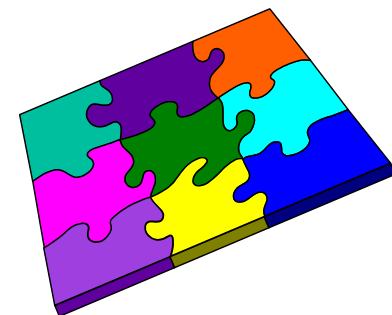
- ➡ 分解具有无损连接性和分解保持函数依赖是两个互相独立的标准。
- ➡ 如果一个分解具有无损连接性，则它能保证不丢失信息。
- ➡ 如果一个分解保持了函数依赖，则它可以减轻或解决各种异常情况。
- ➡ 最后应当注意的是，规范化理论为数据库设计提供了理论的指南和工具，但仅仅是指南和工具。并不是规范化程度越高，模式就越好，实际应用中还必须结合应用环境和具体情况，合理地选择数据库模式。



# 第六章 关系数据理论

## 本章主要内容

- 问题的提出
- 规范化
- 数据依赖的公理系统
- 模式的分解
- 小结



- ➡ 规范化理论为数据库设计提供了理论的指南和工具
  - ☐ 也仅仅是指南和工具
- ➡ 并不是规范化程度越高，模式就越好
  - ☐ 必须结合应用环境和现实世界的具体情况合理地选择数据库模式