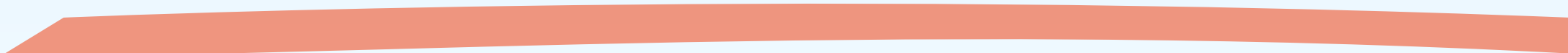


数据库系统

第6章 关系数据理论



胡 敏

jsjxhumin@hfut.edu.cn

第六章 关系数据理论

6.1 问题的提出

6.2 规范化

6.3 数据依赖的公理系统

*6.4 模式的分解

6.5 小结



6.3 数据依赖的公理系统

- 数据依赖的公理系统是模式分解算法的理论基础。
- 函数依赖的一个有效而完备的公理系统——Armstrong公理系统，一套推理规则，是模式分解算法的理论基础。

已知： $R\langle U, F \rangle$, $U = \{X, C, W, Y, Z\}$, $F = \{X \rightarrow YZ, Z \rightarrow CW\}$

问： $X \rightarrow CWYZ$ 是否为 F 逻辑蕴含

■ 用途：

- 从一组函数依赖求得蕴含的函数依赖。例如问 $X \rightarrow Y$ 是否被 F 所蕴含。
- 求给定关系模式的码。
- 模式分解。



6.3 数据依赖的公理系统

■ 逻辑蕴含

定义6.11 对于满足一组函数依赖 F 的关系模式 $R \langle U, F \rangle$ ，其任何一个关系 r ，若函数依赖 $X \rightarrow Y$ 都成立，（即 r 中任意两元组 t, s ，若 $t[X]=s[X]$ ，则 $t[Y]=s[Y]$ ），则称 F 逻辑蕴含 $X \rightarrow Y$



1. Armstrong公理系统

■ 设 U 为属性集总体， F 是 U 上的一组函数依赖，于是有关系模式 $R \langle U, F \rangle$ 。对 $R \langle U, F \rangle$ 来说有以下的推理规则：

- A1.自反律 (Reflexivity)：若 $Y \subseteq X \subseteq U$ ，则 $X \rightarrow Y$ 为 F 所蕴含。
- A2.增广律 (Augmentation)：若 $X \rightarrow Y$ 为 F 所蕴含，且 $Z \subseteq U$ ，则 $XZ \rightarrow YZ$ 为 F 所蕴含。
- A3.传递律 (Transitivity)：若 $X \rightarrow Y$ 及 $Y \rightarrow Z$ 为 F 所蕴含，则 $X \rightarrow Z$ 为 F 所蕴含。

注意：自反律所得到的函数依赖均是平凡的函数依赖，自反律的使用并不依赖于 F

定理6.1 Armstrong推理规则是正确的
(证明参见《数据库系统概论P. 190~P. 191》)



2. 导出规则

1. 根据A1, A2, A3这三条推理规则可以得到下面三条推理规则:

➤ 合并规则: 由 $X \rightarrow Y$, $X \rightarrow Z$, 有 $X \rightarrow YZ$ 。

(A2, A3)

若已知 $sno \rightarrow sname$, $sno \rightarrow sdept$
可得: $sno \rightarrow sname, sdept$

➤ 伪传递规则: 由 $X \rightarrow Y$, $WY \rightarrow Z$, 有 $XW \rightarrow Z$ 。

(A2, A3)

➤ 分解规则: 由 $X \rightarrow Y$ 及 $Z \subseteq Y$, 有 $X \rightarrow Z$ 。

(A1, A3)

若已知: $sno \rightarrow sname, sdept$
可得: $sno \rightarrow sname$, $sno \rightarrow sdept$

2. 根据合并规则和分解规则, 可得引理6.1

引理6.1 $X \rightarrow A_1 A_2 \dots A_k$ 成立的充分必要条件是 $X \rightarrow A_i$ 成立 ($i=1, 2, \dots, k$)



实例:设有关系模式R, A、B、C、D、E、F是它的属性子集, R满足下列函数依赖:

$$F=\{A\rightarrow BC, \quad CD\rightarrow EF\}$$

证明: 函数依赖 $AD\rightarrow F$ 成立。

证明:	$A\rightarrow BC$	条件
	$A\rightarrow C$	分解规则
	$AD\rightarrow CD$	增广律
	$CD\rightarrow EF$	条件
	$AD\rightarrow EF$	传递律
	$AD\rightarrow F$	分解规则



3. 函数依赖闭包

■ 函数依赖闭包定义

➤ 定义6.12: 由被**F**逻辑蕴涵的函数依赖的全体构成的集合, 称为**F**的闭包, 记作**F⁺**。

■ 定义6.13 设**F**为属性集**U**上的一组函数依赖, $X \subseteq U$, $X_F^+ = \{ A/X \rightarrow A \text{ 能由 } F \text{ 根据Armstrong公理导出} \}$, X_F^+ 称为属性集**X**关于函数依赖集**F** 的闭包。



3. 函数依赖闭包

$R\langle U, F \rangle$, $U=(X, Y, Z)$, $F= \{ X \rightarrow Y, Y \rightarrow Z \}$,
 $F^+ = \{$

$X \rightarrow X$, $Y \rightarrow Y$, $Z \rightarrow Z$,
 $X \rightarrow Y$, $Y \rightarrow Z$,
 $X \rightarrow Z$, $Y \rightarrow YZ$,
 $X \rightarrow XY$,
 $X \rightarrow XZ$,
 $X \rightarrow YZ$,

$XY \rightarrow X$, $XZ \rightarrow X$, $YZ \rightarrow Y$,
 $XY \rightarrow Y$, $XZ \rightarrow Y$, $YZ \rightarrow Z$,
 $XY \rightarrow Z$, $XZ \rightarrow Z$, $YZ \rightarrow YZ$,
 $XY \rightarrow XY$, $XZ \rightarrow XY$, Z ,
 $XY \rightarrow YZ$, $XZ \rightarrow XZ$,
 $XY \rightarrow XZ$, $XZ \rightarrow XY$,
 $XY \rightarrow XYZ$, $XZ \rightarrow XYZ$,

$XYZ \rightarrow X$,
 $XYZ \rightarrow Y$,
 $XYZ \rightarrow$
 $XYZ \rightarrow XY$,
 $XYZ \rightarrow YZ$
 $XYZ \rightarrow XZ$,
 $XYZ \rightarrow XYZ$

}



3. 函数依赖闭包

■ 引理6.2

设 F 为属性集 U 上的一组函数依赖, $X, Y \subseteq U$, $X \rightarrow Y$ 能由 F 根据Armstrong公理导出的充分必要条件是 $Y \subseteq X_F^+$

■ 用途

- (1) 将判定 $X \rightarrow Y$ 是否能由 F 根据Armstrong公理导出的问题, 转化为求出 X_F^+ 、判定 Y 是否为 X_F^+ 的子集的问题
- (2) 如果 $X_F^+ = U$, X 是 $R \langle U, F \rangle$ 的候选码。

■ X_F^+ 可以用算法来求得!



4. 求属性集 X 关于 F 的闭包 X_F

算法6.1

对于 $R\langle U, F \rangle$ ，求属性集 X ($X \subseteq U$) 关于 U 上的函

数依赖集 F 的闭包 X_F^+ 。

输入: X, F

输出: X_F^+

步骤:

迭代



4.求闭包的算法

[例1] 已知关系模式 $R\langle U, F\rangle$, 其中

$U=\{A, B, C, D, E\};$

$F=\{AB\rightarrow C, B\rightarrow D, C\rightarrow E, EC\rightarrow B, AC\rightarrow B\}.$

求 $(AB)_F^+$ 。

解 设 $X^{(0)}=AB;$

(1) $X^{(1)}=AB\cup CD=ABCD.$

(2) $X^{(0)}\neq X^{(1)}$

$X^{(2)}=X^{(1)}\cup BE=ABCDE.$

(3) $X^{(2)}=U$, 算法终止
 $\rightarrow (AB)_F^+=ABCDE.$

则再在F中找出左边是AB子集的函数依赖, 其结果是: **$AB\rightarrow C$** , **$B\rightarrow D$** , 则**CD**将加入闭包集中。

则再在F中找出左边是ABCDE子集的函数依赖, 并在F中未扫描过的依赖, 其结果是: **$C\rightarrow E$** , **$AC\rightarrow B$** , 则**BE**将加入闭包集中。



4.求闭包的算法

■ 算法6.1 求属性集 X ($X \subseteq U$) 关于 U 上的函数依赖集 F 的闭包 X_F^+

输入: X, F

输出: X_F^+

步骤:

(1) 令 $X^{(0)} = X, i=0$

(2) 求 B , 这里 $B = \{ A | (\exists V)(\exists W)(V \rightarrow W \in F \wedge V \subseteq X^{(i)} \wedge A \in W) \}$;

(3) $X^{(i+1)} = B \cup X^{(i)}$

(4) 判断 $X^{(i+1)} = X^{(i)}$ 吗?

(5) 若相等或 $X^{(i)} = U$, 则 $X^{(i)}$ 就是 X_F^+ , 算法终止。

(6) 若否, 则 $i=i+1$, 返回第 (2) 步。

对 $X^{(i)}$ 中的每个元素, 依次检查相应的函数依赖, 将依赖它的属性加入 B 。

即在 F 中寻找其左边为 $X^{(i)}$ 的子集的函数依赖, 将其右边在 $X^{(i)}$ 中未出现过的属性组成的新集合 B

对于算法6.1, 令 $a_i = |X^{(i)}|$, $\{a_i\}$ 形成一个步长大于1的严格递增的序列, 序列的上界是 $|U|$, 因此该算法最多 $|U| - |X|$ 次循环就会终止。



4.求闭包的算法

例2、 $U=\{A, B, C, D\}$; $F=\{A\rightarrow B, BC\rightarrow D, B\rightarrow C\}$;

(1) 求 A_F^+

解: $X(0)=A$

$\therefore X(1)=X(0) \cup B=AB$, 显然 $X(1) \neq X(0)$ 。

$\therefore X(2)=X(1) \cup C=ABC$, 显然 $X(1) \neq X(2)$ 。

$\therefore X(3)=X(2) \cup D=ABCD=U$, 循环终止。

$\therefore A_F^+=ABCD$ 。

则再在**F**中找出左边是**ABC**子集的函数依赖, 其结果是:
 $A\rightarrow B, B\rightarrow C, BC\rightarrow D$ 。
则**D**将加入闭包集中。



(2) 求 B_F^+ 。

$X(0)=B$

$X(1)=BCD$

$X(2)=BCD=X(1)$, 循环终止。所以 $B_F^+ = BCD$ 。

(3) 求 C_F^+

$X(0)=C$

$X(1)=C=X(0)$, 循环终止。所以 $C_F^+ = C$ 。



5. Armstrong公理系统的有效性与完备性

■ 有效性与完备性的含义

- 有效性：由 F 出发根据Armstrong公理推导出来的每一个函数依赖一定在 F^+ 中
- 完备性： F^+ 中的每一个函数依赖，必定可以由 F 出发根据Armstrong公理推导出来

■ 定理6.2 Armstrong公理系统是有效的、完备的。

（证明参见《数据库系统概论P.192~P.193》）



6. 函数依赖集等价

定义6.14 如果 $G^+=F^+$ ，就说函数依赖集 F 覆盖 G （ F 是 G 的覆盖，或 G 是 F 的覆盖），或 F 与 G 等价。 注：表示能力上是完全相同的。

引理6.3 $F^+ = G^+$ 的充分必要条件是 $F \subseteq G^+$ ，和 $G \subseteq F^+$

证：必要性显然，只证充分性。

(1) 若 $F \subseteq G^+$ ，则 $X_F^+ \subseteq X_{G^+}^+$ 。

(2) 任取 $X \rightarrow Y \in F^+$ 则有 $Y \subseteq X_F^+ \subseteq X_{G^+}^+$ 。

所以 $X \rightarrow Y \in (G^+)^+ = G^+$ 。即 $F^+ \subseteq G^+$ 。

(3) 同理可证 $G^+ \subseteq F^+$ ，所以 $F^+ = G^+$ 。

如： $F1 = \{sno \rightarrow sname, sno \rightarrow sdept, sdept \rightarrow dname\}$

$F2 = \{sno \rightarrow sname, sdept \rightarrow dname, sno \rightarrow sdept, sno \rightarrow dname\}$ $F1 \equiv F2$

7. 最小依赖集

定义6.15 如果函数依赖集 F 满足下列条件，则称 F 为一个极小函数依赖集。亦称为最小依赖集或最小覆盖。

(1) F 中任一函数依赖的右部仅含有一个属性。

即 F 中的函数依赖均不能由 F 中其他函数依赖导出

(2) F 中不存在这样的函数依赖 $X \rightarrow A$ ，使得 F 与 $F - \{X \rightarrow A\}$ 等价。

F 中各函数依赖左部均为最小属性集(不存在冗余属性)

(3) F 中不存在这样的函数依赖 $X \rightarrow A$ ， X 有真子集 Z 使得 $F - \{X \rightarrow A\} \cup \{Z \rightarrow A\}$ 与 F 等价。



例题

$R\langle U, F \rangle$, $U=ABCD$,

- 函数依赖集 $F = \{A \rightarrow BD, AB \rightarrow C, C \rightarrow D\}$ 。
- 求：F最小函数依赖集
- 解法步骤：
 - 1.将F中的所有函数依赖的右边化为单一属性
 - 2.去掉F中的所有函数依赖左边的冗余属性
 - 3.去掉F中所有冗余的函数依赖



例题图解 (1)

❖ (1) 将F中的所有函数依赖右边化为单一属性

$$F = \{A \rightarrow BD, AB \rightarrow C, C \rightarrow D\}$$

$$\text{❖ } F = \{ A \rightarrow B, A \rightarrow D, AB \rightarrow C, C \rightarrow D \}$$



例题图解 (2)

❖ 2. 去掉F中的所有函数依赖左边的冗余属性

$$F = \{A \rightarrow B, A \rightarrow D, AB \rightarrow C, C \rightarrow D\}$$

$$A^+ = \{A, B, C, D\}$$

$$B^+ = \{B\}$$

$$F = \{A \rightarrow B, A \rightarrow D, A \rightarrow C, C \rightarrow D\}$$



例题图解 (3)

❖ 3. 去掉F中所有冗余的函数依赖关系

$F = \{A \rightarrow B, A \rightarrow D, A \rightarrow C, C \rightarrow D\}$

$F = \{ A \rightarrow B, \text{ (空) }, A \rightarrow C, C \rightarrow D \}$

$A^+ = \{ A, B, C, D \}$

$F = \{ A \rightarrow B, A \rightarrow C, C \rightarrow D \}$



最小依赖集 例子

[例2] 关系模式 $S\langle U, F\rangle$, 其中:

$U = \{ \text{Sno}, \text{Sdept}, \text{Mname}, \text{Cno}, \text{Grade} \},$

$F = \{ \text{Sno} \rightarrow \text{Sdept}, \text{Sdept} \rightarrow \text{Mname}, (\text{Sno}, \text{Cno}) \rightarrow \text{Grade} \}$

设 $F' = \{ \text{Sno} \rightarrow \text{Sdept}, \text{Sno} \rightarrow \text{Mname}, \text{Sdept} \rightarrow \text{Mname},$

$(\text{Sno}, \text{Cno}) \rightarrow \text{Grade}, (\text{Sno}, \text{Sdept}) \rightarrow \text{Sdept} \}$

F 是最小覆盖, 而 F' 不是。

因为: $F' - \{ \text{Sno} \rightarrow \text{Mname} \}$ 与 F' 等价

$F' - \{ (\text{Sno}, \text{Sdept}) \rightarrow \text{Sdept} \}$ 也与 F' 等价



极小化过程（续）

[例3] $F = \{A \rightarrow B, B \rightarrow A, B \rightarrow C, A \rightarrow C, C \rightarrow A\}$

F_{m1} 、 F_{m2} 都是 F 的最小依赖集：

$$F_{m1} = \{A \rightarrow B, B \rightarrow C, C \rightarrow A\}$$

$$F_{m2} = \{A \rightarrow B, B \rightarrow A, A \rightarrow C, C \rightarrow A\}$$

- F 的最小依赖集 F_m 不唯一
- 极小化过程也是检验 F 是否为极小依赖集的一个算法



6.3 数据依赖的公理系统

1. Armstrong公理系统
2. 导出规则
3. 函数依赖闭包 F^+
4. 求属性集 X ($X \subseteq U$) 关于 F 的闭包 X_F^+
5. Armstrong公理系统的有效性与完备性
6. 函数依赖集等价的概念
7. 最小依赖集或最小覆盖



第六章 关系数据理论

6.1 问题的提出

6.2 规范化

6.3 数据依赖的公理系统

***6.4 模式的分解**

6.5 小结



6.4 模式的分解(选学)

- 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。

目的: 规范化的目的是使结构合理，使数据冗余尽量小，清除插入，删除和更新异常。

方法: 将关系模式投影分解成两个或两个以上的关系模式，但关系模式的分解不是唯一的。

要求: 分解后的关系模式集合应当与原关系模式“等价”，既“具有无损连接性（保证不丢失信息），又保持函数依赖特性（减轻或解决各种异常）”

一个关系模式达到BCNF，说明在函数依赖的范畴内，已实现了彻底分离，可消除“异常”，但在实际应用中，并不一定要求全部模式都达到BCNF。



6.4.1 模式分解的3个定义（续）

■ 关系模式的规范化过程是通过对关系模式的分解来实现的

➤ 什么是模式分解

定义6.16 $R\langle U, F \rangle$ 的一个分解是指： $\rho = \{R_1\langle U_1, F_1 \rangle, \dots, R_n\langle U_n, F_n \rangle\}$ ，其中 $U = U_1 \cup U_2 \cup \dots \cup U_n$ ，并且没有 $U_i \subseteq U_j$ ， $1 \leq i, j \leq n$ ， F_i 是 F 在 U_i 上的投影。

相应地将 R 存储在二维表 r 中的数据分散到二维表 r_1, r_2, \dots, r_n 中去，其中 r_i 是 r 在属性集 U_i 上的投影。

定义6.17 函数依赖集合 $\{X \rightarrow Y \mid X \rightarrow Y \in F \wedge XY \subseteq U_i\}$ 的一个覆盖 F_i 叫作 F 在属性 U_i 上的投影。



关系模式分解的标准

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



分解准则

- 将一个关系模式 $R\langle U, F \rangle$ 分解为若干个关系模式 $R_1\langle U_1, F_1 \rangle, R_2\langle U_2, F_2 \rangle, \dots, R_n\langle U_n, F_n \rangle$, 意味着将存储在一张二维表 r 中的数据分散到了若干二维表 r_1, r_2, \dots, r_n 中。
- 分解不应该丢失信息, 即通过对关系 r_1, r_2, \dots, r_n 的自然连接能重新得到关系 r 中的所有信息。
- 将关系 r 投影为 r_1, r_2, \dots, r_n 时不会丢失信息, 但是对 r_1, r_2, \dots, r_n 做自然连接时可能产生一些 r 中原来没有的元组, 从而无法区别哪些元组是 r 中原来有的, 哪些是不应该有的。--丢失了信息



示例

S-D-L (Sno, Dept, Loc)

■ 三种分解方案:

方案1: S-L (Sno, Loc) , D-L (Dept, Loc)

方案2: S-D (Sno, Dept) , S-L (Sno, Loc)

方案3: S-D (Sno, Dept) , D-L (Dept, Loc)



关系模式分解准确（续）

这三种分解方案是否都满足分解要求呢？

假设此关系模式的数据如表所示，此关系用 r 表示。

<i>Sno</i>	<i>Dept</i>	<i>Loc</i>
<i>S01</i>	<i>D1</i>	<i>L1</i>
<i>S02</i>	<i>D2</i>	<i>L2</i>
<i>S03</i>	<i>D2</i>	<i>L2</i>
<i>S04</i>	<i>D3</i>	<i>L1</i>



分析方案1

若按方案1将S-D-L投影到S-L和D-L的属性上，然后做自然连接。

Sno	Loc
S01	L1
S02	L2
S03	L2
S04	L1

Dept	Loc
D1	L1
D2	L2
D3	L1

自然连接

Sno	Dept	Loc
S01	D1	L1
S01	D3	L1
S02	D2	L2
S03	D2	L2
S04	D1	L1
S04	D3	L1



无损连接性

- 定义6.18 将关系模式 $R\langle U, F \rangle$ 分解为个关系模式 $R_1\langle U_1, F_1 \rangle, R_2\langle U_2, F_2 \rangle, \dots, R_n\langle U_n, F_n \rangle$, 若对于 R 中的任何一个可能的 r , 都有 $r = r_1 * r_2 * \dots * r_n$, 即 r 在 R_1, R_2, \dots, R_n 上的投影的自然连接等于 r , 则称关系模式 R 的这个分解具有无损连接性。
- 方案1不具有无损连接性。

Sno	Dept	Loc
S01	D1	L1
S02	D2	L2
S03	D2	L2
S04	D3	L1

Sno	Dept	Loc
S01	D1	L1
S01	D3	L1
S02	D2	L2
S03	D2	L2
S04	D1	L1
S04	D3	L1



分析方案2

分析方案2。将S-D-L投影到S-D，S-L的属性上，然后做自然连接。

Sno	Dept
S01	D1
S02	D2
S03	D2
S04	D3

Sno	Loc
S01	L1
S02	L2
S03	L2
S04	L1

自然连接

Sno	Dept	Loc
S01	D1	L1
S02	D2	L2
S03	D2	L2
S04	D3	L1

方案2具有无损连接性。



分析方案3

分析方案3。将S-D-L投影到S-D，S-L的属性上，然后做自然连接。

Sno	Dept
S01	D1
S02	D2
S03	D2
S04	D3

Dept	Loc
D1	L1
D2	L2
D3	L1

自然连接

Sno	Dept	Loc
S01	D1	L1
S02	D2	L2
S03	D2	L2
S04	D3	L1

方案3具有无损连接性。



模式的分解（续）

定义 6.19 设关系模式 $R\langle U, F \rangle$ 被分解为若干个关系模式 $R_1\langle U_1, F_1 \rangle$, $R_2\langle U_2, F_2 \rangle$, ...,

$R_n\langle U_n, F_n \rangle$

（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 所逻辑蕴含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 所逻辑蕴含，则称关系模式 R 的这个分解是保持函数依赖的（Preserve dependency）



方案2

■ 方案2具有无损连接性，但没保持函数依赖

如果 (S03, D2)
改为 (S03, D3)

则 (S03, L2) 需
改为 (S03, L1)

Sno	Dept
S01	D1
S03	D2
S03	D2
S04	D3

Sno	Loc
S01	L1
S02	L2
S03	L2
S03	L2
S04	L1

Sno	Dept	Loc
S01	D1	L1
S02	D2	L2
S03	D2	L2
S04	D3	L1

如果这两个修改没有同时进行，则会出现不一致信息。S-D-L中的函数依赖
 $Sdept \rightarrow Sloc$ 既没有投影到 关系模式SD上，也没有投影到关系模式SL上
因此方案2没有保持原有的函数依赖关系： $Dept \rightarrow Loc$



分析方案3

- 分解方案3既满足无损连接性，又保持了原有的函数依赖关系 $Sno \rightarrow Sdept$ 、 $Sdept \rightarrow Sloc$ ，因此是好的分解方法。

如果 (S03,D2) 改为 (S03,D3)，这时D-L 不需要修改。

Sno	Dept
S01	D1
S02	D2
S03	D2
S03	D3
S04	D3

Dept	Loc
D1	L1
D2	L2
D3	L1

Sno	Dept	Loc
S01	D1	L1
S02	D2	L2
S03	D2	L2
S04	D3	L1



关系模式设计的一般原则

- 一般情况下，在进行模式分解时，应将有直接依赖关系的属性放置在一个关系模式中，
- 这样得到的分解结果一般具有无损连接性，并能保持函数依赖关系不变。



模式的分解总结

- 如果一个分解具有无损连接性，则它能够保证不丢失信息
- 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
- 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。具有无损连接性的分解不一定能够保持函数依赖；同样，保持函数依赖的分解也不一定具有无损连接性。



第六章 关系数据理论

6.1 问题的提出

6.2 规范化

6.3 数据依赖的公理系统

*6.4 模式的分解

6.5 小结



第六章 关系数据理论

6.1 问题的提出

6.2 规范化

6.3 数据依赖的公理系统

*6.4 模式的分解

6.5 小结



6.5 小结

■ 函数依赖

- 平凡函数依赖与非平凡函数依赖
- 完全函数依赖与部分函数依赖
- 传递函数依赖
- 码

■ 范式的概念

■ 关系模式规范化的基本步骤

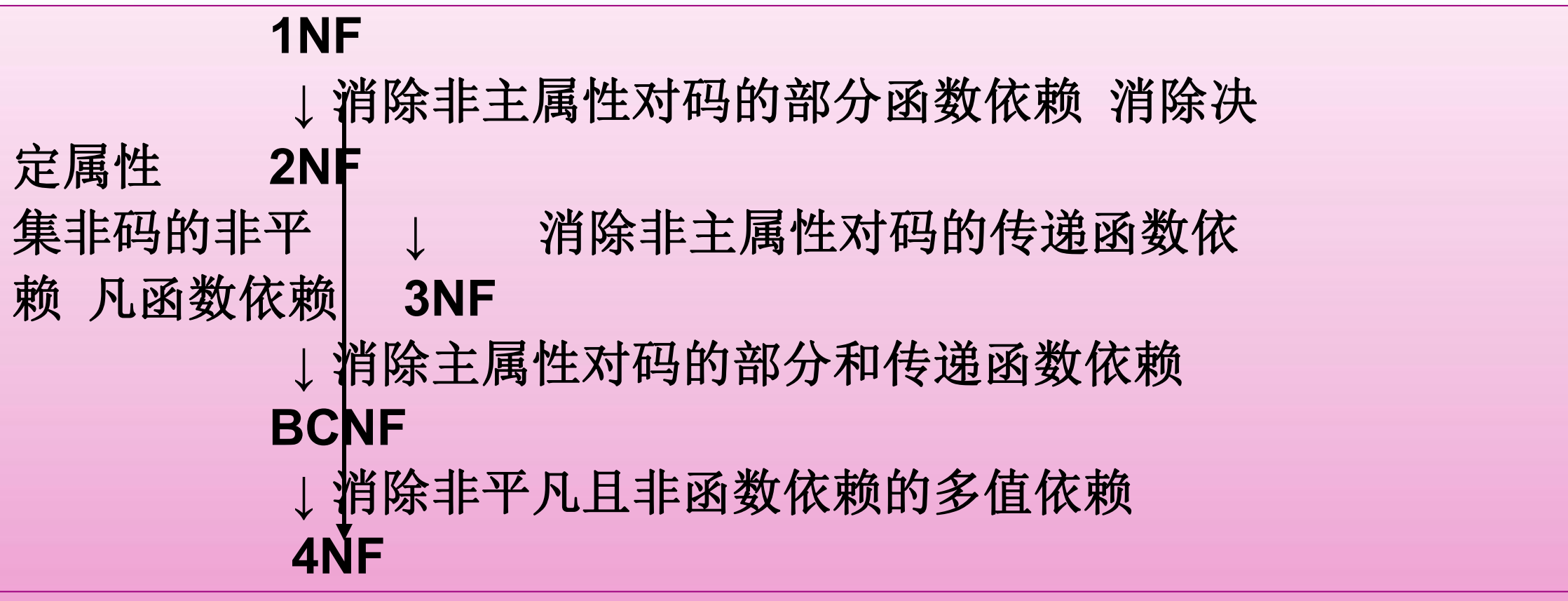
■ Armstrong公理系统

■ 模式的分解



6.5 小结

范式



6.5 Armstrong公理系统

- 自反律；增广律；传递律
- 合并规则；伪传递规则；分解规则
- 函数依赖闭包 F^+ 求属性集 X 关于 F 的闭包 X_F^+
- 最小依赖集



小结

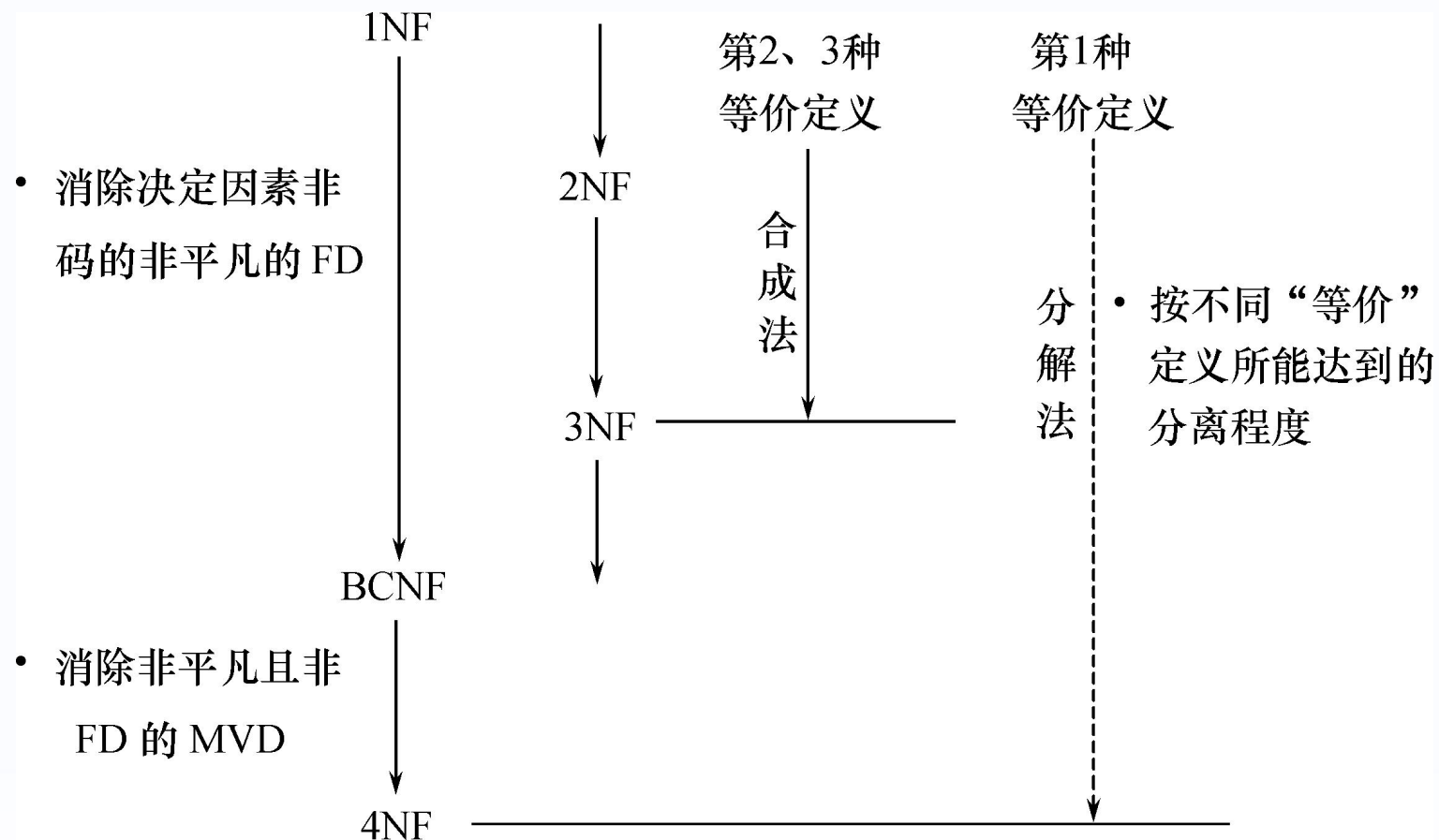
关系模式的规范化基本思想

1. 逐步消除不合适的数据依赖中使模式中的各关系模式达到某种程度的“分离”——模式分解
2. 一事一地的模式设计原则
3. 规范化理论为数据库设计提供了理论指南和算法工具。
4. 结合应用环境的具体情况，合理地设计数据库模式。



6.5 小结

关系模式的规范化，其基本思想：



作业

P203: 2、6、8



下课了。。。。

研究



休息一会儿。。。。

