

第4章 关系数据理论

4.1 关系模式设计中的问题

4.2 数据依赖

4.3 关系模式分解

4.4 关系模式规范化

4.1 关系模式设计中的问题

- 针对一个具体的应用问题，应该如何构造一个适合于它的关系数据库模式？
- 应该构造几个关系模式？
- 每个关系由哪些属性组成？
- 本章的任务：研究设计一个“好”的（即没有“毛病”的）关系模式的方法。

关系数据库设计中的主要问题

- ◆ 什么样的数据库模式才合理？
- ◆ 怎么分解才能满足要求？
- ◆ 衡量的标准是什么？
- ◆ 理论基础是什么？
- ◆ 如何实现？

[例4-1]：学校数据库, 单一的关系模式 : Student $\langle U, F \rangle$
 $U = \{Sno, Sdept, Mname, Cname, Grade\}$

■ 属性定义:

- Sno: 学生学号、 Sdept: 所在系、
- Mname: 系主任姓名、 Cname: 课程名、
- Grade: 成绩

■ 问题: 这是一个“好”的关系模式吗?

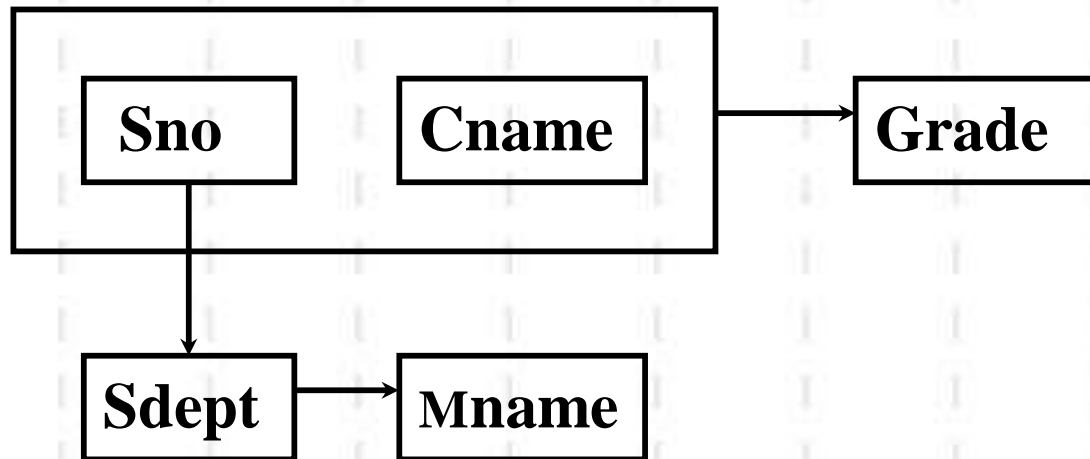
学校数据库的语义：

1. 一个系有若干学生， 一个学生只属于一个系；
2. 一个系只有一名主任；
3. 一个学生可以选修多门课程， 每门课程有若干学生选修；
4. 每个学生所学的每门课程都有一个成绩。

实例分析

- 属性组U上的一组函数依赖F:

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



实例分析

关系模式Student<U, F>中存在的问题

Student(Sno, Sdept, Mname, Cname, Grade)

1. 数据冗余——浪费大量的存储空间
2. 更新异常 ~~Update Anomalies~~ 每个系主任的姓名重复出现

数据冗余，更新数据时，维护数据完整性代价大。

3. 插入异常 ~~Insert Anomalies~~ 例：某系更换系主任后，系统必须修改与该系学生有关的每一个元组。

4. 删除异常 ~~Delete Anomalies~~ 例：如果某个系刚成立，尚无学生，我们就无法把这个系及其系主任的信息存入数据库。。

例，如果某个系的学生全部毕业了，我们在删除该系学生信息的同时，把这个系及其系主任的信息也丢掉了。

结论

- Student关系模式不是一个好的模式。
- “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。

分解成三个关系模式：

S (SNO, SDEPT, $SNO \rightarrow DEPT$);

SG (SNO, CNAME, Grade, $(SNO, CNAME) \rightarrow Grade$);

DEPT (SDEPT, Mname, $SDEPT \rightarrow MN$);

关系规范化

- 关系模式设计得好与坏，直接影响到数据冗余度、数据一致性等问题。
- 要设计好的数据库模式，必须有一定的理论为基础。这就是关系规范化理论。
- 关系规范化的过程是以达到高级别范式的关系去取代原有关系的过程。
- 随着规范程度的提高，冗余与操作异常可能性减少。

4.2 数据依赖

- 数据依赖
 - 通过关系中属性间值的相等与否体现出的数据间的相互依存和制约关系；
 - 是对现实世界中事物间相互联系的抽象, 是数据的内在性质, 是语义的体现。
- 数据依赖的类型
 - 函数依赖 (Functional Dependency, 简记为FD)
 - 多值依赖 (Multivalued Dependency, 简记为MVD)
 - 连接依赖 (Join Dependency, 简记为JD)

函数依赖的定义

- 定义4.1: 设 R 为关系模式, r 为 R 上的任意一个关系实例, X, Y 是 R 的两个属性子集, 若对于 r 上的任意两个元组 t_1, t_2 , 都有: 如果 $t_1[X] = t_2[X]$, 则必有 $t_1[Y] = t_2[Y]$, 则称在 R 上 X 函数决定 Y 或者 Y 函数依赖于 X , 记为 $X \rightarrow Y$, 称 X 为决定子或左部, 称 Y 为被决定子或右部。

函数依赖的定义：例

<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>
<i>a1</i>	<i>b1</i>	<i>c1</i>	<i>d1</i>
<i>a1</i>	<i>b1</i>	<i>c2</i>	<i>d2</i>
<i>a2</i>	<i>b2</i>	<i>c3</i>	<i>d3</i>
<i>a3</i>	<i>b1</i>	<i>c4</i>	<i>d4</i>

<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>
<i>a1</i>	<i>b1</i>	<i>c1</i>	<i>d1</i>
<i>a1</i>	<i>b2</i>	<i>c2</i>	<i>d2</i>
<i>a2</i>	<i>b2</i>	<i>c3</i>	<i>d3</i>
<i>a3</i>	<i>b2</i>	<i>c4</i>	<i>d4</i>

- 分别检验： $A \rightarrow C$? $C \rightarrow A$? $AB \rightarrow D$?
- 辨识：
 - 满足依赖的关系：依赖在模式的某个关系实例上成立
 - 模式上成立的依赖：依赖在模式的所有关系实例上都成立

函数依赖的定义

- 函数依赖不是指关系模式R的某个或某些关系实例满足的约束条件，而是指R的所有关系实例均要满足的约束条件。
- 函数依赖是语义范畴的概念，只能根据数据的语义来确定函数依赖是否成立。
- 例：对于学生关系
- 学生(学号, 姓名, 性别, 出生日期, 班级, 宿舍, 专业)
- 班级 \rightarrow 性别 是否成立？
- 如果规定“男女生分班”，则成立
- 否则，不成立。

平凡函数依赖与非平凡函数依赖

定义4.2：在关系模式 $R(U)$ 中，对于 U 的子集 X 和 Y ：

如果 $X \rightarrow Y$ ，且 $Y \subseteq X$ ，则称 $X \rightarrow Y$ 是平凡的函数依赖

若 $X \rightarrow Y$ ，且 $Y \not\subseteq X$ ，则称 $X \rightarrow Y$ 是非平凡的函数依赖

对于任一关系模式，平凡函数依赖都是必然成立的，它不反映新的语义，因此若不特别声明，我们总是讨论非平凡函数依赖。

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： $(Sno, Cno) \rightarrow Sno$

$(Sno, Cno) \rightarrow Cno$

完全函数依赖与部分函数依赖

定义4.3: 在关系模式 $R(U)$ 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 完全函数依赖于 X , 记作 $X \xrightarrow{F} Y$ 。

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

完全函数依赖中的左部不包含冗余属性, 即只要将决定子中的任何一个属性去掉, 这个函数依赖就不再成立了

例: 在关系 $SC(Sno, Cno, Grade)$ 中,

由于: $Sno \rightarrow Grade, Cno \rightarrow Grade$,

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

传递函数依赖与直接函数依赖

定义4.4: 在关系模式 $R(U)$ 中, 如果 $X \rightarrow Y$, $Y \rightarrow Z$, 且 $Y \subsetneq X$, $Y \not\rightarrow X$, 则称 Z 传递函数依赖于 X 。否则称 Z 直接函数依赖于 X

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

例: 在关系 $Std(Sno, Sdept, Mname)$ 中, 有:

$Sno \rightarrow Sdept, Sdept \rightarrow Mname$

因此, $Mname$ 传递函数依赖于 Sno

函数依赖的逻辑蕴涵

- 定义4.5: 设 F 是在关系模式 R 上的函数依赖的集合, $X \rightarrow Y$ 是一个函数依赖, 如果对于 R 的每个满足 F 的关系 r 也满足 $X \rightarrow Y$, 那么称 F 逻辑蕴涵 $X \rightarrow Y$, 记为 $F \models X \rightarrow Y$
- 例如: $\{X \rightarrow Y, Y \rightarrow Z\} \models X \rightarrow Z$

函数依赖集合的闭包

➤ 定义4.6：由函数依赖集合F所逻辑蕴涵的全部函数依赖所构成的集合称为F的闭包，记作 F^+ ，即：

$$F^+ = \{ X \rightarrow Y \mid F \models X \rightarrow Y \}$$

➤ 据定义4.6易知，函数依赖集的闭包 F^+ 具有以下特点：

- (1) $F \subseteq F^+$ ，这是因为根据闭包的定义F中的每个函数依赖必定也在 F^+ 中；
- (2) $(F^+)^+ = F^+$ ，该性质说明闭包运算是幂等的，即F经过任意多次的闭包运算后其结果仍然等于 F^+ ；
- (3) 如果 $F = F^+$ ，则称F是完备的

函数依赖集合的闭包

•例4-2: 已知关系模式R(ABC),

$F = \{A \rightarrow B, B \rightarrow C\}$, 求 F^+

解: 据已知条件和推理规则, 可知 F^+ 有43个FD:

$A \rightarrow \phi$	$AB \rightarrow \phi$	$AC \rightarrow \phi$	$ABC \rightarrow \phi$	$B \rightarrow \phi$	$C \rightarrow \phi$
$A \rightarrow A$	$AB \rightarrow A$	$AC \rightarrow A$	$ABC \rightarrow A$	$B \rightarrow B$	$C \rightarrow C$
$A \rightarrow B$	$AB \rightarrow B$	$AC \rightarrow B$	$ABC \rightarrow B$	$B \rightarrow C$	$\phi \rightarrow \phi$
$A \rightarrow C$	$AB \rightarrow C$	$AC \rightarrow C$	$ABC \rightarrow C$	$B \rightarrow BC$	
$A \rightarrow AB$	$AB \rightarrow AB$	$AC \rightarrow AB$	$ABC \rightarrow AB$	$BC \rightarrow \phi$	
$A \rightarrow AC$	$AB \rightarrow AC$	$AC \rightarrow AC$	$ABC \rightarrow AC$	$BC \rightarrow B$	
$A \rightarrow BC$	$AB \rightarrow BC$	$AC \rightarrow BC$	$ABC \rightarrow BC$	$BC \rightarrow C$	
$A \rightarrow ABC$	$AB \rightarrow ABC$	$AC \rightarrow ABC$	$ABC \rightarrow ABC$	$BC \rightarrow BC$	

• F^+ 的计算是一个NP完全问题

函数依赖的推理规则

■ 设 U 是关系模式 R 的属性集， F 是 R 上的函数依赖集，则有：

➤ A1（自反律）：如果 $Y \subseteq X \subseteq U$ ，则 $X \rightarrow Y$ 成立；

➤ A2（增广律）：如果 $X \rightarrow Y$ 成立，且 $Z \subseteq U$ ，则 $XZ \rightarrow YZ$ 成立

➤ A3（传递律）：如果 $X \rightarrow Y$ ， $Y \rightarrow Z$ 成立，则 $X \rightarrow Z$ 成立

· 人们又把自反律(A1)，传递律(A2)和增广律(A3)称为Armstrong公理

函数依赖的推理规则

2022-3-20

⊆

➤ 引理4-1: Armstrong公理是正确的

证: A1中的 $X \rightarrow Y$ 其实就是平凡的函数依赖, 我们仅对A2、A3进行证明:

A2: 设 t_1, t_2 是关系模式R的任一关系实例r上的两个元组, 如果 $t_1[XZ] = t_2[XZ]$ 成立, 则一定有 $t_1[X] = t_2[X]$ 及 $t_1[Z] = t_2[Z]$ 也成立。根据已知 $X \rightarrow Y$ 及 $t_1[X] = t_2[X]$ 可得 $t_1[Y] = t_2[Y]$, 于是可知 $t_1[Y] = t_2[Y]$ 与 $t_1[Z] = t_2[Z]$ 同时成立, 即 $t_1[YZ] = t_2[YZ]$, 由定义4-1易知 $XZ \rightarrow YZ$ 成立。

A3: 设 t_1, t_2 是关系模式R的任一关系实例r上的两个元组, 如果 $t_1[X] = t_2[X]$ 成立, 则根据 $X \rightarrow Y$ 可得 $t_1[Y] = t_2[Y]$, 又因为 $Y \rightarrow Z$ 也成立, 所以有 $t_1[Z] = t_2[Z]$, 结合 $t_1[X] = t_2[X]$, $t_1[Z] = t_2[Z]$ 及定义4-1可知 $X \rightarrow Z$ 成立。

函数依赖的推理规则

➤ 由A1~A3易推出下面的三条推理规则也是正确的：

- (1) 合并规则：若 $X \rightarrow Y$, $X \rightarrow Z$ 成立, 则 $X \rightarrow YZ$ 成立；
- (2) 伪传递规则：若 $X \rightarrow Y$, $WY \subseteq Z$ 成立, 则 $WX \rightarrow Z$ 成立；
- (3) 分解规则：若 $X \rightarrow Y$, 且 $Z \subseteq Y$, 则有 $X \rightarrow Z$ ；

➤ 证：略

FD的逻辑导出

- 定义4.7：给定关系模式 $R\langle U, F \rangle$ ，如果能由 F 根据Armstrong公理推导出 $X \rightarrow Y$ ，则称 $X \rightarrow Y$ 是 F 的逻辑导出，记为： $F \Rightarrow X \rightarrow Y$
- 定义4.8：属性集 X 关于 $R(U, F)$ 上的函数依赖集合 F 的闭包 X_F^+ 定义为 $X_F^+ = \{A \mid A \in U, F \Rightarrow X \rightarrow A\}$
- 根据A1（自反律）可知对于任一属性集 X 一定有 $X \subseteq X_F^+$ ，此外，求 X_F^+ 的计算量应该远远小于求 F^+ 的计算量，因为 X_F^+ 中的属性是可数的，最多就是属性全集 U 而已

属性集合的闭包

- 引理4-2： $X \rightarrow Y$ 可由Armstrong公理推出的充分必要条件是 $Y \subseteq X_F^+$
- 证：充分性：已知 $Y \subseteq X_F^+$ 成立，要证 $X \rightarrow Y$ 可由Armstrong公理推出。设 $Y = B_1 B_2 \cdots B_n$ ，则有 $B_1 B_2 \cdots B_n \subseteq X_F^+$ ，于是有 $B_i \in X_F^+$ ($1 \leq i \leq n$)，按照定义4.6可知 $X \rightarrow B_i$ 可由Armstrong公理推出，再根据合并规则及 i 的任意性可得 $X \rightarrow B_1 B_2 \cdots B_n$ 可由Armstrong公理推出，即 $X \rightarrow Y$ 可由Armstrong公理推出；
- 必要性：已知 $X \rightarrow Y$ 可由Armstrong公理推出，要证 $Y \subseteq X_F^+$ 。类似于充分性的证明，利用分解规则可证。□

属性集的闭包

- 算法4.1: 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 $B=\{A \mid (\text{存在 } V \rightarrow W) (V \rightarrow W \in F \wedge V \subseteq X(i) \wedge A \in W)\}$;
 - 3) $X(i+1)=X(i) \cup B$
 - 4) 判断 $X(i+1)=X(i)$ 吗? X_F^+
 - 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。

属性集的闭包

- 例4-3: 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: 由算法4.1, $X(0) = AB$; 计算 $X(1)$; 逐一的扫描 F 集合中各个函数依赖, 找左部为 A, B , 或 AB 的函数依赖。得到两个: $AB \rightarrow C, B \rightarrow D$ 。于是 $X(1) = AB \cup CD = ABCD$ 。
因为 $X(0) \neq X(1)$, 所以再找出左部为 $ABCD$ 子集的那些函数依赖, 又得到 $C \rightarrow E, AC \rightarrow B$, 于是 $X(2) = X(1) \cup BE = ABCDE$ 。
因为 $X(2)$ 已等于全部属性集合, 所以 $(AB)^{+}_F = ABCDE$
- 对于算法4.1, 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

属性集的闭包

- 例4-4：属性集U为ABCD，
FD集为 $\{ A \rightarrow B, B \rightarrow C, D \rightarrow B \}$
- 则用上述算法，可求出：
- $A^+ = ABC$ ，
 $AD^+ = ABCD$
 $BD^+ = BCD$
等等

Armstrong公理的正确性和完备性

- **Armstrong公理的正确性**是指“使用推理规则从FD集F推出的FD必定在 F^+ 中”
即，如果 $F \Rightarrow X \rightarrow Y$ 则 $F \models X \rightarrow Y$
- **Armstrong公理的完备性**是指“ F^+ 中的FD都能从F集使用推理规则集导出”
即，如果 $F \models X \rightarrow Y$ 则 $F \Rightarrow X \rightarrow Y$
- **Armstrong公理的正确性和完备性**保证了FD推导过程的有效性和可靠性

Armstrong公理的正确性和完备性

- 定理4.1: Armstrong公理是正确的
- 证明: 略

Armstrong公理的正确性和完备性

- **定理4.2: Armstrong公理是完备的**

- 证明: 我们证明完备性的逆否命题, 即若函数依赖 $X \rightarrow Y$ 不能由F从Armstrong公理导出, 那么它必然不为F所蕴含, 其证明分三步:

- 1) 若 $V \rightarrow W$ 成立, 且 $V \subseteq X_F^+$, 则 $W \subseteq X_F^+$
因为 $V \subseteq X_F^+$, 所以 $X \rightarrow V$ 成立; 由A3规则有 $X \rightarrow W$ 成立, 所以 $W \subseteq X_F^+$

Armstrong公理的正确性和完备性

2) 由下列两个元组构成的二维表，必是 $R\langle U, F \rangle$ 的一个关系 r ，即满足 F 中的全部函数依赖。

X_F^+						$U - X_F^+$					
1	1	.	.	.	1	0	0	.	.	.	0
1	1	.	.	.	1	1	1	.	.	.	1

若 r 不是 $R\langle U, F \rangle$ 的关系，则必由于 F 中有函数依赖 $V \rightarrow W$ 在 r 上不成立所致，由 r 的构成可知， V 必定是 X_F^+ 的子集，而 W 不是 X_F^+ 的子集，但是由1) $W \subseteq X_F^+$ ，矛盾！所以 r 必是 $R\langle U, F \rangle$ 的一个关系

Armstrong公理的正确性和完备性

3) 若 $X \rightarrow Y$ 不能由F从Armstrong公理导出, 则Y不是 X_F^+ 的子集, 因此必有Y的子集 Y' 满足 $Y' \subseteq U - X_F^+$, 即 $X \rightarrow Y$ 在r上不成立, 故 $X \rightarrow Y$ 必不为F蕴含。

证毕

Armstrong公理的正确性和完备性

- Armstrong公理的正确性和完备性说明“逻辑导出”与“逻辑蕴含”是两个等阶的概念。
- 因此, F^+ 也可以说成是由 F 出发借助Armstrong公理导出的函数依赖的集合。
- 从蕴含(或导出)的概念出发, 又引出了两个函数依赖集等价和最小依赖集的概念。

函数依赖集的等价覆盖

- 定义4.9：设 F, G 是两个函数依赖集合，如果 $F^+ = G^+$ ，则称 F 等价于 G ，或者 F 与 G 互相覆盖。
- 引理4.3： F 与 G 等价的充分必要条件是 $F \subseteq G^+$ 并且 $G \subseteq F^+$
- 证：必要性显然，只需证充分性：
 - 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 $F^+ = G^+$

函数依赖集的等价覆盖

- 要判定 $F \subseteq G^+$, 只须逐一一对 F 中的函数依赖 $X \rightarrow Y$, 考查 Y 是否属于 $X_{G^+}^+$
- 引理4.3给出了判断两个函数依赖集等价的可行算法。

最小函数依赖集

➤ 定义4.10：若F满足下列条件：

1) F中所有函数依赖的右部均为单属性；

2) F中不存在这样的函数依赖 $X \rightarrow A$ 及 $Z \subset X$ ，使得

$$F^+ = (F - \{X \rightarrow A\} \cup \{Z \rightarrow A\})^+$$

即F中所有函数依赖的左部没有多余属性；

3) F中不存在这样的函数依赖 $X \rightarrow A$ ：使得

$$F^+ = (F - \{X \rightarrow A\})^+$$

即F中没有多余的函数依赖

则称F为最小函数依赖集或最小覆盖

最小函数依赖集

- 定理4.3：每一个函数依赖集F均等价于一个最小函数依赖集 F_{min}
 - 证：这是一个构造性的证明，我们分三步对F进行“极小化处理”，找出F的一个最小依赖集：
 1. 逐一检查F中各函数依赖 $FD_i: X \rightarrow Y$ ，若： $Y = B_1 \dots B_k$ ， $k \geq 2$ ，则以 $\{X \rightarrow B_j \mid j=1, 2, \dots, k\}$ 取代 $X \rightarrow Y$ ；
 2. 逐一检查F中各函数依赖 $FD_i: X \rightarrow A$ ，令 $G = F - \{X \rightarrow A\}$ ，若 $A \in X_G^+$ ，则从F中去掉此函数依赖
- 注：因为F与G等价的充要条件是 $A \in X_G^+$

最小函数依赖集

3. 逐一取出 F 中各函数依赖 $FD_i: X \rightarrow A$, 设: $X = B_1 \dots B_m$, $m > 1$
逐一考查 B_i ($i = 1, 2, \dots, m$), 若 $A \in (X - B_i)_{F^+}$, 则以
 $(X - B_i) \rightarrow A$ 取代 $X \rightarrow A$

因为 F 与 $F - \{X \rightarrow A\} \cup \{Z \rightarrow A\}$ 等价的充要条件是 $A \in Z^+$ 其中
 $Z = X - B_i$

- 最后剩下的 F 就一定是最小依赖集, 并且与原来的 F 等价, 因为, 我们对 F 的每一次“改造”都保证了改造前后的两个函数依赖集等价。

最小函数依赖集

例4-5: $R(A, B, C, D, E, H, I)$,

$F = \{A \rightarrow BE, AH \rightarrow D, B \rightarrow C, BD \rightarrow E, C \rightarrow D, H \rightarrow I, I \rightarrow H, H \rightarrow BE\}$

试求F的最小依赖集 F_{min}

解: (1)右部拆成单属性

$F = \{A \rightarrow B, A \rightarrow E, AH \rightarrow D, B \rightarrow C, BD \rightarrow E, C \rightarrow D, H \rightarrow I, I \rightarrow H, H \rightarrow B, H \rightarrow E\}$

(2)考察左部不是单属性的函数依赖, 消除多余属性

$AH \rightarrow D \quad \because ((AH) - H)_F^+ = ABECD, D \in ((AH) - H)_F^+$

\therefore 以 $A \rightarrow D$ 取代 $AH \rightarrow D$

$BD \rightarrow E \quad \because ((BD) - D)_F^+ = BCDE, E \in ((BD) - D)_F^+$

\therefore 以 $B \rightarrow E$ 取代 $BD \rightarrow E$

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

最小函数依赖集

(3)消除多余的函数依赖

$A \rightarrow B$ $\because A_G^+ = AED, B \notin A_G^+ (G=F - \{A \rightarrow B\}) \therefore$ 保留该函数依赖

$A \rightarrow E$ $\because A_G^+ = ABCDE, E \in A_G^+ (G=F - \{A \rightarrow E\}) \therefore$ 不保留该函数依赖

$A \rightarrow D$ $\because (A)_G^+ = ABCDE, D \in (A)_G^+ (G=F - \{A \rightarrow D\}) \therefore$ 不保留该函数依赖

$B \rightarrow C$ $\because B_G^+ = B, C \notin B_G^+ (G=F - \{B \rightarrow C\}) \therefore$ 保留该函数依赖

$B \rightarrow E$ $\because (B)_G^+ = BCD, E \notin (B)_G^+ (G=F - \{B \rightarrow E\}) \therefore$ 保留该函数依赖

$C \rightarrow D$ $\because C_G^+ = AE, D \notin C_G^+ (G=F - \{C \rightarrow D\}) \therefore$ 保留该函数依赖

$H \rightarrow I$ $\because H_G^+ = HBECD, I \notin H_G^+ (G=F - \{H \rightarrow I\}) \therefore$ 保留该函数依赖

$I \rightarrow H$ $\because I_G^+ = I, H \notin I_G^+ (G=F - \{I \rightarrow H\}) \therefore$ 保留该函数依赖

$H \rightarrow B$ $\because H_G^+ = HIE, B \notin H_G^+ (G=F - \{H \rightarrow B\}) \therefore$ 保留该函数依赖

$H \rightarrow E$ $\because H_G^+ = HBCDE, E \in H_G^+ (G=F - \{H \rightarrow E\}) \therefore$ 不保留该函数依赖

最后剩下的F是最小函数依赖集：

$F_{\min} = \{A \rightarrow B, B \rightarrow C, B \rightarrow E, C \rightarrow D, H \rightarrow I, I \rightarrow H, H \rightarrow B\}$

最小函数依赖集

- F 的最小依赖集 F_{min} 不一定唯一，它和我们对各函数依赖 FD_i 及 $X \rightarrow A$ 中 X 各属性的处置顺序有关。

多值依赖

- 除了函数依赖以外，关系的属性间还存在其他一些数据依赖关系，多值依赖 (multivalued dependency, MVD) 就是其中之一。

多值依赖

- 例4-6：学校中某一门课程由多个教员讲授，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。我们可以用一个非规范化的关系来表示教员T, 课程C和参考书B之间的关系：

• 课程C 教员T 参考书B

物理	李 勇	普通物理学
	王 军	光学原理
		物理习题集

数学	李 勇	数学分析
	张 平	微分方程
		高等代数

多值依赖

- 把这张表变成一张规范化的二维表, 就成为:
- Teaching

课程C	教员T	参考书B
物理	李	普通物理学
物理	李	光学原理
物理	李	物理习题集
物理	王	普通物理学
物理	王	光学原理
物理	王	物理习题集
数学	李	数学分析
数学	李	微分方程
数学	李	高等代数
数学	张	数学分析
数学	张	微分方程
数学	张	高等代数

多值依赖

- 仔细考察这类关系模式，发现它具有有一种称为多值依赖(MVD)的数据依赖。
- 定义4.11：设 $R(U)$ 是属性集 U 上的一个关系模式。 X , Y , Z 是 U 的子集, 并且 $Z=U-XY$ 。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。
- 例如，在关系模式TEACHING中，对于(物理, 光学原理)有一组 T 值{李勇, 王军}，这组值仅仅决定于课程 C 上的值(物理)，也就是说对于另一个(物理, 普通物理学)它对应的一组 T 值仍是{李勇, 王军}，尽管这时参考书 B 的值已经改变了。
- 因此， T 多值依赖于 C ，即 $C \twoheadrightarrow T$ 。

多值依赖

- 以上对多值依赖进行了一些直观的讨论，下面对MVD进行形式化描述
- 定义4.12：设 $R(U)$ 是属性集 U 上的一个关系模式。 X, Y, Z 是 U 的子集, 并且 $Z=U-X-Y$, 如果对 $R(U)$ 的任一关系 r , 都有如下性质：如果 r 中存在2个元组 s, t , 使得：

$$s[X]=t[X]$$

则 r 中必存在元组 u , 使得：

$$(1) \quad s[X]=t[X]=u[X]$$

$$(2) \quad u[Y]=t[Y] \text{ 且 } u[Z]=s[Z]$$

(即交换 s, t 在 Y 上的值得到的2个元组必在 r 中)

则称关系模式 R 满足多值依赖 $X \twoheadrightarrow Y$

多值依赖

- 一般, 当关系至少有三个属性, 其中的两个是多值时, 且它们的值只依赖于第三个属性时, 才会有多值依赖。
- 如果关系中存在非平凡非FD的多值依赖则会导致数据冗余和更新异常等问题, 所以要消除异常, 必须消除多值依赖。
- 方法是, 通过建立两个关系, 让每个关系只存储一个多值属性的数据。

MVD举例

对于W的每一个值，S有一个完整的集与它对应不论C取什么值，
所以 $W \twoheadrightarrow S$

- 例4-7：关系模式WSC(W, S, C)中，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

MVD举例

- 例4-8: 设关系模式 $R(A, B, C)$ 上有一个MVD $A \twoheadrightarrow B$, 如果 R 中已有 (a, b_1, c_1) , (a, b_2, c_2) , (a, b_3, c_3) , 那么这些关系中至少还应该存在哪些元组?

至少还存在6个元组:

(a, b_1, c_2) (a, b_2, c_1) (a, b_1, c_3)
)

(a, b_3, c_1) (a, b_2, c_3) (a, b_3, c_2)
)

有关多值依赖的推导规则

- 与函数依赖一样，多值依赖也有一组推导规则：
- A4：互补律(MVD)
 - 如果 $X \twoheadrightarrow Y$ ，则 $X \twoheadrightarrow (U - XY)$
 - 以后如果需要，可用 $X \twoheadrightarrow Y \mid (U - XY)$ 表示多值依赖，以强调其互补关系
- A5：扩展律(MVD)
 - 如果 $X \twoheadrightarrow Y$ 且 $V \subseteq W$ ，则 $WX \twoheadrightarrow VY$
- A6：传递律(MVD)
 - 如果 $X \twoheadrightarrow Y$ 且 $Y \twoheadrightarrow Z$ ，则 $X \twoheadrightarrow (Z - Y)$
- 下面两条为(FD+MVD)公理：
- A7：如果 $X \rightarrow Y$ ，则 $X \twoheadrightarrow Y$ ，即FD是MVD的特例
- A8：如果 $X \twoheadrightarrow Y$ 、 $Z \subseteq Y$ 且对某个与Y不相交的W有： $W \rightarrow Z$ ，则 $X \rightarrow Z$

有关多值依赖的推导规则

- 由上述公理，还可以得出下列四个有关MVD的推导规则：

1) MVD合并规则

如果 $X \twoheadrightarrow Y$ 、 $X \twoheadrightarrow Z$ ，则 $X \twoheadrightarrow YZ$

2) MVD伪传递规则

如果 $X \twoheadrightarrow Y$ 、 $WY \twoheadrightarrow Z$ ，则 $WX \twoheadrightarrow (Z - WY)$

3) 混合伪传递规则

如果 $X \twoheadrightarrow Y$ 、 $XY \twoheadrightarrow Z$ ，则 $X \twoheadrightarrow (Z - Y)$

4) MVD分解规则

如果 $X \twoheadrightarrow Y$ 、 $X \twoheadrightarrow Z$ ，则 $X \twoheadrightarrow (Y \cap Z)$ 、 $X \twoheadrightarrow (Y - Z)$
 $X \twoheadrightarrow (Z - Y)$ 均成立。

- 以上规则的证明从略

多值依赖的性质

- 定义4.13: 若 $X \twoheadrightarrow Y$, 而 $Z = U - XY$ 为空, 则称 $X \twoheadrightarrow Y$ 为平凡的多值依赖。
- 多值依赖具有以下性质:
 - 1) 对称性: 即若 $X \twoheadrightarrow Y$, 则 $X \twoheadrightarrow Z$, 其中 $Z = U - XY$
 - 2) 传递性: 即若 $X \twoheadrightarrow Y$, $Y \twoheadrightarrow Z$, 则 $X \twoheadrightarrow Z - Y$
 - 3) 函数依赖可以看作是多值依赖的特殊情况: 即若 $X \rightarrow Y$, 则 $X \twoheadrightarrow Y$ 。这是因为当 $X \rightarrow Y$ 时, 对 X 的每一个值 x , Y 有一个确定的值 y 与之对应, 所以 $X \twoheadrightarrow Y$
 - 4) 若 $X \twoheadrightarrow Y$, $X \twoheadrightarrow Z$, 则 $X \twoheadrightarrow YZ$
 - 5) 若 $X \twoheadrightarrow Y$, $X \twoheadrightarrow Z$, 则 $X \twoheadrightarrow Y \cap Z$
 - 6) 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

- 区别

- 函数依赖规定某些元组不能出现在关系中，也称为相等产生依赖
- 多值依赖要求某种形式的其它元组必须在关系中，称为元组产生依赖

- 有效性范围

- $X \rightarrow Y$ 的有效性仅决定于 X 、 Y 属性集上的值，它在任何属性集 W ($XY \subseteq W \subseteq U$) 上都成立。

若 $X \rightarrow Y$ 在 $R(U)$ 上成立，则对于任何 $Y' \subseteq Y$ ，均有 $X \rightarrow Y'$ 成立。

多值依赖与函数依赖

- 有效性范围
- $X \twoheadrightarrow Y$ 的有效性与其属性集范围有关
- $X \twoheadrightarrow Y$ 在属性集 W ($XY \subseteq W \subseteq U$) 上成立, 但在 U 上不一定成立
- $X \twoheadrightarrow Y$ 在 U 上成立 $\Rightarrow X \twoheadrightarrow Y$ 在属性集 W ($XY \subseteq W \subseteq U$) 上成立
- 若在 $R(U)$ 上, $X \twoheadrightarrow Y$ 在属性集 W ($XY \subseteq W \subseteq U$) 上成立, 则称 $X \twoheadrightarrow Y$ 为 $R(U)$ 的嵌入式多值依赖
- 若 $X \twoheadrightarrow Y$ 在 $R(U)$ 上成立, 则不能断言对于 $Y' \subseteq Y$, 是否有 $X \twoheadrightarrow Y'$ 成立

嵌入型多值依赖

- 定义4.14: 设关系模式 $R(U)$, X 和 Y 是属性集 U 的子集, W 是 U 的真子集, 并且 $XY \subseteq W$ 。如果MVD $X \twoheadrightarrow Y$ 在模式 R 上不成立, 但在模式 W 上成立, 则称 $X \twoheadrightarrow Y$ 为 R 上的嵌入型多值依赖 (Embedded MVD, 简记为eMVD), 用符号 $(X \twoheadrightarrow Y)_W$ 表示。

4.3 关系模式分解

- 我们可以通过把一个“大的”关系模式分解成多个“小的”关系模式，以解决插入异常、删除异常和更新异常等问题。
- 实际上，关系模式的分解，不仅仅是属性集合的分解，也是对关系模式上的函数依赖集，以及关系模式的当前值的分解的具体表现。
- 关系模式的分解并不是随意的，必须保证原来的关系模式的信息和语义性质不被丢失，即要求分解应当既“无损分解”又“保持函数依赖”。

关系模式分解的定义

- 定义4. 15: 设 $R(U)$ 为关系模式, 则称:
 $\rho = \{R_1(U_1), R_2(U_2), \dots, R_k(U_k)\}$ (其中 $U = \bigcup_{i=1}^k U_i$) 为 R 的一个分解
- 定义4. 16: 函数依赖集 F 在属性集 $U_i (\subseteq U)$ 上的投影定义为:

$$\Pi_{U_i}(F) = \{X \rightarrow Y \mid X \rightarrow Y \in F^+ \wedge XY \subseteq U_i\}$$

- U 称为泛关系模式, U 对应的当前值称为泛关系
- 数据库对应的当前值 σ 称为数据库实例, 它是由数据库模式中的每一个关系模式的当前值组成, 即:

$$\sigma = \{r_1, \dots, r_k\}$$

泛关系模式

泛关系模式

U



数据库模式

$\rho = \{U_1, \dots, U_k\}$

u



$\sigma = \langle u_1, \dots, u_k \rangle$

泛关系

数据库实例(数据库)

函数依赖集 $F \dots\dots\dots \{F_1, \dots, F_k\}$

这里有两个问题:

- ① σ 和 u 是否等价? 用“无损分解”表示
- ② $\{F_1, \dots, F_k\}$ 与 F 是否等价? 用“保持依赖”表示

泛关系假设

- 假设已知一个数据库模式U，它仅由单个关系模式组成，问题是要设计一个模式SD，它与U“等价”，但在某些方面更好一些。
- 泛关系假设是运用规范化理论的障碍，因为承认了泛关系假设，就等于承认了现实世界各实体间只能有一种联系，而这常常是办不到的，比如工人与机器之间就可以存在“使用”、“维护”等多种联系。
- 对此，人们提出了一些办法，希望解决这个矛盾。
- 例如，建立一个不受泛关系假设限制的理论，或者采用某些手段使现实世界向信息世界转换时适合于泛关系的要求。

无损联接分解

- 定义4.17: 设R是一个关系模式, F是R上的一个FD集, $\rho = \{ R_1, \dots, R_k \}$ 为R的一个分解, 如果对R中满足F的每一个关系r, 都有

$$r = \pi_{R_1}(r) \bowtie \dots \bowtie \pi_{R_k}(r)$$

- 则称分解 ρ 相对于F是“无损联接分解”(lossless join decomposition), 简称“无损分解”, 否则, 称 ρ 为“有损联接分解”(loss join decomposition), 简称“有损分解”

无损联接分解：例

• 例4-9:

r	A	B	C
	1	1	1
	1	2	1

(a)

r ₁	A	B
	1	1
	1	2

(b)

r ₂	A	C
	1	1

(c)

图 未丢失信息的分解

r	A	B	C
	1	1	4
	1	2	3

(a)

r ₁	A	B
	1	1
	1	2

(b)

r ₂	A	C
	1	4
	1	3

(c)

r ₁ ⋈ r ₂	A	B	C
	1	1	4
	1	1	3
	1	2	4
	1	2	3

(d)

图 丢失信息的分解

投影连接运算

- 定义4.18: 设 $\rho = \{ R_1, \dots, R_k \}$ 为 R 的一个分解, r 是 R 上的一个具体关系, 则 r 对于 ρ 的投影连接运算定义为:

$$m_{\rho}(r) = \bigtriangleright \bigtriangleleft_{i=1}^k \prod_{ui} (r)$$

- 引理4.4: 设 $\rho = \{ R_1, \dots, R_k \}$ 是 R 的一个分解, r 是 R 的任一关系, $r_i = \pi_{R_i}(r)$ ($1 \leq i \leq k$), 那么有下列性质:

- ① $r \subseteq m_{\rho}(r)$;
- ② 若 $s = m_{\rho}(r)$, 则 $\pi_{R_i}(s) = r_i$;
- ③ $m_{\rho}(m_{\rho}(r)) = m_{\rho}(r)$, 这个性质称为幂等性 (Idempotent)

投影连接运算

•证明:

(1) 任取 $t \in r$, 则 $t[R_i] \in r_i$ ($i=1, 2, \dots, k$)。由于 $t[R_1], \dots, t[R_k]$ 本来取自 t 在 R_i 上的投影, 因此在进行连接时, 必然相互匹配, 拼接成元组 t , 故 $t \in m_\rho(r)$, 即 $r \subseteq m_\rho(r)$

(2) 由 (1) 可知, $r \subseteq s$, 因此 $\Pi_{R_i}(r) \subseteq \Pi_{R_i}(s)$, 或 $r_i \subseteq \Pi_{R_i}(s)$ 。现只须证 $\Pi_{R_i}(s) \subseteq r_i$ 。为此任娶 $u \in \Pi_{R_i}(s)$, 则必有 $t_s \in s$, 使 $t_s[R_i] = u$, 由 s 的构造方式可知 $t_s[R_i] \in r_i$ ($i=1, \dots, k$), 即 $u \in r_i$, 因此有 $\Pi_{R_i}(s) \subseteq r_i$

(3) 由 $\Pi_{R_i}(s) = r_i$, 可知:

$$m_\rho(m_\rho(r)) = m_\rho(s) = \bigtriangleright \bigtriangleleft_{i=1}^k \prod_{R_i}(s) = \bigtriangleright \bigtriangleleft_{i=1}^k (r_i) = m_\rho(r)$$

(证毕)

无损联接分解

- 由引理4.4可得到下面的结论：
- 如果 $r \neq m\rho(r)$ ，则 ρ 不是无损分解。
- 由引理4.4(1)可知， r 经过分解后再连接，如果 ρ 不是无损分解，则所得到的结果的元组数总比原来的 r 多，即所谓“元组增加，信息丢失！”（有损）。
- 由引理4.4(2)可知，对 r 进行 $m\rho(r)$ 变换后得到的 s 可满足条件：
$$\Pi_{R_i}(s) = r_i$$
- 但必须注意：如果 r_i 不是 r 的投影，而是独立的关系，则一般而言，连接后的关系 s 不再满足条件 $\Pi_{R_i}(s) = r_i$ ，而是 $\Pi_{R_i}(s) \subseteq r_i$

无损联接分解

- 例4-10: 设关系模式 $R(ABC)$ 分解成 $\rho = \{ AB, BC \}$,
(a) 和 (b) 分别是模式AB和BC上的值 r_1 和 r_2 , (c) 是 $r_1 \bowtie r_2$ 的值。显然 $\pi_{BC}(r_1 \bowtie r_2) \neq r_2$, 这里 r_2 中元组 (b_2c_2) 是一个悬挂元组, 由于它的存在, 使得 r_1 和 r_2 不存在泛关系 r

r_1	A	B
	a_1	b_1

(a) 关系 r_1

r_2	B	C
	b_1	c_1
	b_2	c_2

(b) 关系 r_2

$r_1 \bowtie r_2$	A	B	C
	a_1	b_1	c_1

(c) $r_1 \bowtie r_2$

无损分解的判别算法

- 算法4.2: 检验一个分解是否无损连接分解
- 输入: 关系模式 $R(A_1, \dots, A_n)$; R 上的函数依赖集 F ;
 R 的一个分解 $\rho = \{R_1, R_2, \dots, R_k\}$;
- 输出: ρ 是否无损连接分解
- 方法: (1) 建立一个 n 列、 k 行的符号表 M :

	A_1	A_2	A_j	A_n
R_1
R_2

R_i			$M[i, j]$...

R_k

无损分解的判别算法

- 符号表M各元素的值由下面的规则确定：

$$M[i, j] = \begin{cases} a_j & \text{若 } A_j \in U_i \\ b_{ij} & \text{若 } A_j \notin U_i \end{cases}$$

- 用F中的每一函数依赖 $X \rightarrow Y$ 对M反复进行下列检查和处理：检查X中的属性所对应的列，找出在X上取值相等的行，如果找到两个(或多个)行在X上取值相等，就将对应行中Y中属性所对应的符号改为一致，即如果其中之一为 a_j ，则将其符号也改为 a_j ；如果全部符号都是“b”符号，则将它们改为同样的“b”符号。
- 如此进行下去，直到发现M中某一行变为： a_1, a_2, \dots, a_n ，则说明 ρ 是无损连接分解。
- 否则，一直进行到M不再改变为止，则说明 ρ 不是无损连接分解。

无损分解的判别算法

- 定理4.4: 算法4.2可以正确判断一个分解是否无损连接分解。
- 证明: (略)

无损分解的判别算法

- 例4-11: $R(A, B, C, D, E)$

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

- 判断:

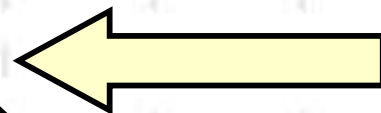
$$p = \{R_1(A, D), R_2(A, B), R_3(B, E), R_4(C, D, E), R_5(A, E)\}$$

- 是否是无损连接的分解?

无损分解的判别算法

1、构造初始表格

	A	B	C	D	E
AD	a1	b12	b13	a4	b15
AB	a1	a2	b13	a4	b25
BE	a1	a2	a3	a4	a5
CDE	a1	b42	a3	a4	a5
AE	a1	b52	a3	a4	a5



2、处理表格

A→C: 将b23, b53改为b13

B→C: 将b33改为b13

C→D: 将b24, b34, b54改为a4

DE→C: 将b13改为a3

CE→A: 将b31,b41改为a1

因此是无损
连接的分解

无损分解的判别算法

- 对于只有2个关系模式的分解可以通过下面的定理判断其是否为无损联接分解
- 定理4.5: 设 $\rho = \{ R_1, R_2 \}$ 是关系模式R的一个分解, F是R上成立的FD集, 那么分解 ρ 相对于F是无损分解的充分必要条件是
$$(R_1 \cap R_2) \rightarrow (R_1 - R_2) \text{ 或 } (R_1 \cap R_2) \rightarrow (R_2 - R_1)$$
- 证明: (略)
- 定理4.6: 如果FD: $X \rightarrow Y$ 在模式R上成立, 且 $X \cap Y = \phi$, 则 $\rho = \{ R - Y, XY \}$ 是R的无损分解
- 证明: (略)

保持函数依赖的分解

- 保持关系模式分解等价的另一个重要条件是关系模式的函数依赖集在分解后仍在数据库模式中保持不变。
- 即关系模式R 到 $\rho = \{R_1, R_2 \cdots R_k\}$ 的分解, 应使函数依赖集F, 被F在这些 R_i 上的投影蕴涵。

保持函数依赖的分解

- 定义4.19: 设 $\rho = \{ R_1, \dots, R_k \}$ 是 R 的一个分解, F 是 R 上的FD集, 如果有:

$$\bigcup \pi_{R_i}(F) \models F$$

则称分解 ρ 保持函数依赖集 F 。

保持函数依赖的分解

- 下面介绍检验分解是否保持函数依赖的算法
- 检验分解是否保持函数依赖实质上就是检验

$\bigcup_{i=1}^k \Pi_{U_i}(F)$ 是否覆盖F

- 算法4.3: 检验分解是否保持函数依赖。
- 输入: $\rho = \{R_1, R_2, \dots, R_k\}$, 函数依赖集F
- 输出: ρ 是否保持F

- 方法: 令 $G = \bigcup_{i=1}^k \Pi_{U_i}(F)$

- 为检验G是否覆盖F, 可对F中的每一函数依赖 $X \rightarrow Y$ 进行下列检验:
- 首先计算 X_G^+ , 然后检查Y是否被包含在 X_G^+ 中

保持函数依赖的分解

- 下面是不必求出 G 而计算 X_G^+ 的算法:

$Z := X;$

repeat

 for $i=1$ to k do

$Z := Z \cup ((Z \cap U_i)_{F^+} \cap U_i)$

until Z 不再变化;

- 如果 Y 被包含在 X_G^+ 中, 则 $X \rightarrow Y \in G^+$
- 如果 F 中的所有函数依赖经检验都属于 G^+ 则 ρ 保持依赖, 否则不保持依赖

保持函数依赖的分解：举例

- 例4. 12: $R(ABCD)$ $F = \{A \rightarrow B, B \rightarrow C, C \rightarrow D, D \rightarrow A\}$
试判别 $\rho = \{R_1(AB), R_2(BC), R_3(CD)\}$ 是否保持依赖 F
- 解：显然， ρ 保持 $A \rightarrow B, B \rightarrow C, C \rightarrow D$
- 下面检验 ρ 是否保持 $D \rightarrow A$?
- 令 $G = \bigcup_{i=1}^k \Pi_{U_i}(F)$
- $\therefore D_G^+ = DCBA$ 其中包含 A
- $\therefore \rho$ 保持依赖 F

保持函数依赖的分解： 举例

- 设关系模式 $R = \{CITY, ST, ZIP\}$ 满足函数依赖：

$$(CITY, ST) \rightarrow ZIP$$

$$ZIP \rightarrow CITY$$

分解成 $\rho = \{R_1(ST, ZIP), R_2(CITY, ZIP)\}$

- 检查该分解是否具有保持函数依赖性：

$$\because \Pi_{R_1} = \emptyset \text{ (平凡的函数依赖)}$$

$$\Pi_{R_2} = \{ZIP \rightarrow CITY\}$$

$$\Pi_{R_1} \cup \Pi_{R_2} = \{ZIP \rightarrow CITY\}$$

- $\therefore \rho$ 不具有保持函数依赖性

模式分解与模式等价问题

- 本节讨论的关系模式分解的两个特性实际上涉及到两个数据库模式的等价问题，这种等价包括数据等价和依赖等价两个方面。
- 数据等价是指两个数据库实例应表示同样的信息内容，用“无损分解”衡量。
- 如果是无损分解，那么对泛关系反复的投影和联接都不会丢失信息。
- 依赖等价是指两个数据库模式应有相同的依赖集闭包。
- 在依赖集闭包相等情况下，数据的语义是不会出差错的。
- 违反数据等价或依赖等价的分解很难说是一个好的模式设计。

模式分解与模式等价问题

- 例4-13: 关系模式R (ABC) , $\rho = \{ AB, AC \}$ 是R的一个分解
- 试分析分别在 $F_1 = \{ A \rightarrow B \}$, $F_2 = \{ A \rightarrow C, B \rightarrow C \}$, $F_3 = \{ B \rightarrow A \}$, $F_4 = \{ C \rightarrow B, B \rightarrow A \}$ 情况下, 检验 ρ 是否具有无损分解和保持FD的分解特性
- 解:
 - ① 相对于 $F_1 = \{ A \rightarrow B \}$, 分解 ρ 是无损分解且保持FD的分解。
 - ② 相对于 $F_2 = \{ A \rightarrow C, B \rightarrow C \}$, 分解 ρ 是无损分解, 但不保持FD集, 因为丢失了 $B \rightarrow C$ 。
 - ③ 相对于 $F_3 = \{ B \rightarrow A \}$, 分解 ρ 是损失分解但保持FD集的分解。
 - ④ 相对于 $F_4 = \{ C \rightarrow B, B \rightarrow A \}$, 分解 ρ 是损失分解且不保持FD集的分解, 因为丢失了 $C \rightarrow B$ 。

4.4 关系模式的规范化

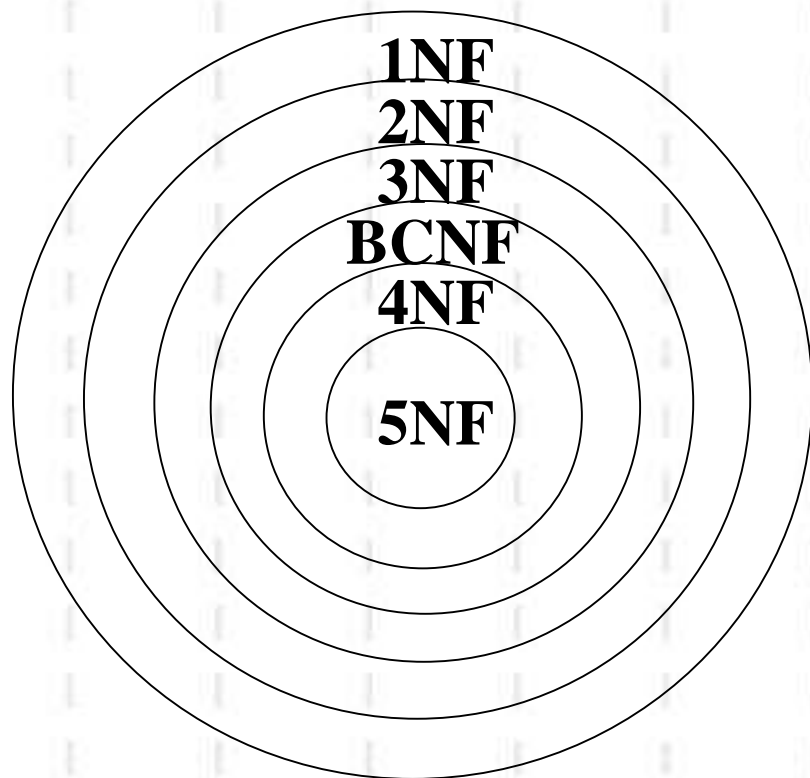
- 为了使数据库设计的理论和方法走向完备，人们研究了关系规范化理论，以指导我们设计规范化的数据库模式。
- 在进行关系模式设计时，首先面临的一个问题是如何评价一个关系模式的“好”与“坏”，即用什么标准衡量？
- 这个标准就是模式的范式(Normal Forms，简记为NF)。
- 而使得关系模式达到一定范式要求，就是关系模式的规范化。

范式

- 关系数据库中的关系模式要满足一定的规范化要求，满足不同程度规范化要求的模式，称为不同的范式 (Normal Forms, 简记为NF)。
- 范式的种类与数据依赖有着直接的联系，基于FD的范式有1NF、2NF、3NF、BCNF等多种。
- 满足最低要求的关系模式称为第一范式，简称1NF。在第一范式中满足进一步要求的为第二范式，其余以此类推。
- R为第几范式就写成： $R \in xNF$

各种范式之间的关系

- 按属性间依赖情况来区分，关系规范化的程度为1NF, 2NF, 3NF, BCNF, 4NF 和 5NF等
- 各种范式之间的关系如图所示：



第一范式(1NF)

- 定义4. 20：如果关系模式R的每个关系r的属性值都是不可分的原子值，则称R是第一范式(first normal form, 简记为1NF)的模式。
- 满足1NF的关系称为“规范化”的关系，否则称为“非规范化”的关系。
- 关系数据库研究的关系都是规范化的关系。
- 例如关系模式R (NAME, ADDRESS, PHONE)，如果一个人有两个电话号码 (PHONE)，那么在关系中至少要出现两个元组，以便存储这两个号码。
- 1NF是关系模式应具备的最起码的条件。

第一范式(1NF)

- 分量是否需要再分，与具体应用有关，如果用到值的一部分，则需要进一步分割，例如：

姓名	生日
王军	68. 7. 10
张立	69. 7. 10
李明	80. 3. 28

姓名	年	月日
王军	68	7. 10
张立	69	7. 10
李明	80	3. 28

- 如果只是查询出生日期，则它满足1NF
- 如果查询两人生日是否相同，则只比较月、日，需要将生日分解，就不满足1NF
- 如果比较两人的生肖呢？

键(Key)

- 定义4.21: 设 K 为 $R\langle U, F \rangle$ 中的属性或属性组合, 若 $K \rightarrow U \in F^+$, 则称 K 为 R 的一个超键
- 定义4.22: 如果一个超键 K 的任何真子集都不再是超键, 则称 K 为 R 的一个候选键(Candidate key), 候选键有时也简称为键。
- 若关系模式中有多个候选键, 则选定其中的一个为主键(Primary key)。
- 定义4.23: 包含在任何一个候选键中的属性, 称为主属性(Prime attribute)或键属性(Key attribute)。不包含在任何候选键中的属性, 则称为非主属性(Nonprime attribute)或非键属性(Non-key attribute)。

键(Key)

- 最简单的情况，单个属性是候选键。最极端的情况下，整个属性组是候选键，并称为全键 (All-key)。
- 例4-14：关系模式 $R(P, W, A)$ ，属性 P 表示演奏者， W 表示作品， A 表示听众。假设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。听众也可以欣赏不同演奏者的不同作品，这个关系模式的候选键为 (P, W, A) ，即All-key。

求关系模式上全部候选键

- 算法4.4: 求关系模式上的全部候选键
- 输入：关系模式 $R(U, F)$ ，其中 F 是最小覆盖
- 输出： R 上的所有候选键
- 步骤
 - 1) 将 R 的全部属性分为四类，分别是 $C1$ ：不在任何函数依赖中出现的属性； $C2$ ：仅在函数依赖决定子中出现的属性； $C3$ ：仅在函数依赖右部出现的属性； $C4$ ：在函数依赖左部右部均有出现的属性。注意，任何候选键中必然会包括 $C1$ 、 $C2$ 中的属性

求关系模式上全部候选键

2) : 若 $(C1 \cup C2)^+ = U$ 或者 $C4 = \Phi$ 则 $C1 \cup C2$ 即为惟一的候选键；否则，逐一将 $C4$ 中的属性加入 $C1 \cup C2$ 并计算其闭包，若其闭包为 U ，则 $C1 \cup C2$ 与该属性构成关系上的一个候选键，重复该过程直至找出所有的候选键。

求关系模式上全部候选键

■例4-14：求 $R(A, B, C, D)$ ， $F = \{A \rightarrow B, B \rightarrow C, BD \rightarrow A\}$ 的所有候选键：

■解： $C1 = \Phi$ ， $C2 = \{D\}$ ， $C3 = \{C\}$ ， $C4 = \{A, B\}$ ， $(C1 \cup C2)^+ = \{D\}$ ，所以 $C1 \cup C2$ 不是候选键，依次将 $C4$ 中的 A 、 B 加入 $C1 \cup C2$ 中计算其闭包可得： $(AD)^+ = ADBC$ ， $(BD)^+ = BDAC$ ，所以可知 R 的候选键为 AD 和 BD

第二范式(2NF)

- 定义4. 24：如果关系模式R中的所有非主属性都完全函数依赖于所有候选键，则称R满足2NF，表示为 $R \in 2NF$ 。
- 2NF是在1NF的基础上消除非主属性对于所有候选键的部分函数依赖。
- 如果数据库模式中每个关系模式都是2NF，则称数据库模式为2NF的数据库模式。

第二范式(2NF)

- 例4-15 : 设关系模式R (S#, C#, GRADE, TNAME, TADDR) 的属性分别表示学生学号、选修课程的编号、成绩、任课教师姓名和教师地址等意义。(S#, C#) 是R的候选键。
- R上有两个FD: (S#, C#) \rightarrow (TNAME, TADDR) 和 C# \rightarrow (TNAME, TADDR), 由于前一个FD是部分依赖, 因此R不是2NF模式。
- 此时R的关系会出现冗余和异常现象, 譬如某一门课程有100个学生选修, 那么在关系中就会存在100个元组, 因而教师的姓名和地址就会重复100次。
- 如果把R分解成R1 (C#, TNAME, TADDR) 和R2 (S#, C#, GRADE) 后, 部分依赖 (S#, C#) \rightarrow (TNAME, TADDR) 就消失了, R1和R2都是2NF模式。

第三范式(3NF)

- 定义4. 25：如果关系模式R中的非主属性既不部分函数依赖也不传递函数依赖于R上的所有候选键，则称R满足3NF，表示为 $R \in 3NF$ 。
- 3NF是在2NF的基础上消除非主属性对于键的传递依赖。
- 如果数据库模式中每个关系模式都是3NF，则称其为3NF的数据库模式。

第三范式(3NF)

- 在例4-15中，R2是2NF模式，而且也是3NF；
R1 (C#, TNAME, TADDR) 是2NF，但不是3NF；
因为R1中存在函数依赖 $C\# \rightarrow TNAME$ 和
 $TNAME \rightarrow TADDR$ ， $C\# \rightarrow TADDR$ 是一个传递依,此
时R1的关系中也会出现冗余和异常操作;譬如一个
教师开设五门课程，那么关系中就会出现五个元组，
教师的地址就会重复五次。
- 如果把R2分解成R21 (TNAME, TADDR) 和R22
(C#, TNAME) 后， $C\# \rightarrow TADDR$ 就不会出现在
R21和R22中,R21和R22都是3NF模式。

第三范式(3NF)

- 推论1: 如果R是3NF模式, 那么R也是2NF模式
- 证明: 略
- 推论2: 设关系模式R, 当R上每一个FD $X \rightarrow A$ 满足下列三个条件之一: $A \in X$ (即 $X \rightarrow A$ 是一个平凡的FD); X 是R的超键; A 是主属性。则关系模式R就是3NF模式
- 证明: 略

第三范式(3NF)

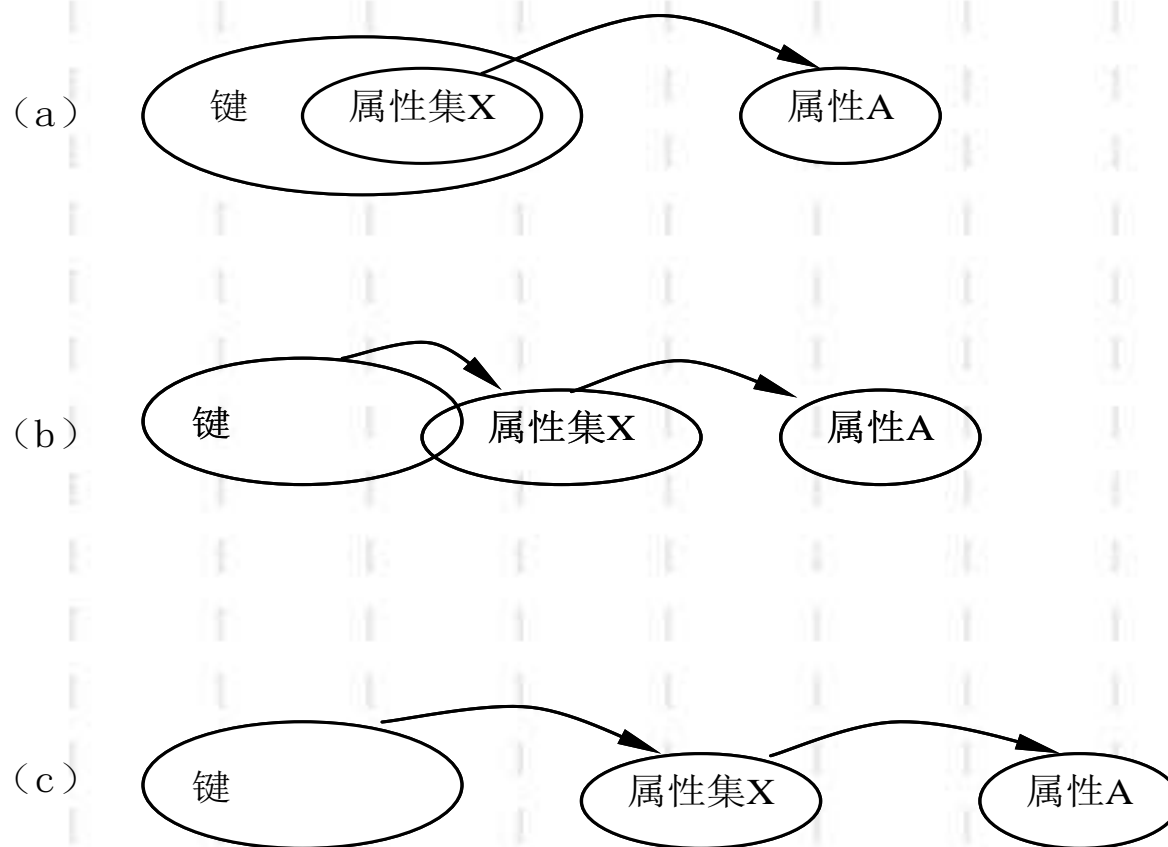


图6.15 违反3NF的传递依赖的三种情况

BCNF

- 定义4. 26：若对于R上的任何非平凡函数依赖 $X \rightarrow Y$ 都有X必包含R的某个候选键，则称R为BCNF，表示为 $R \in \text{BCNF}$ 。
- 定义4. 26'：如果关系模式R中的每个属性都不传递依赖于R的候选键，则称R是BCNF模式。

BCNF

- 推论：如果R是BCNF模式，那么R也是3NF模式。
- BCNF是在3NF的基础上消除主属性对于键的部分依赖和传递依赖。
- 如果数据库模式中每个关系模式都是BCNF，则称为BCNF的数据库模式。

BCNF

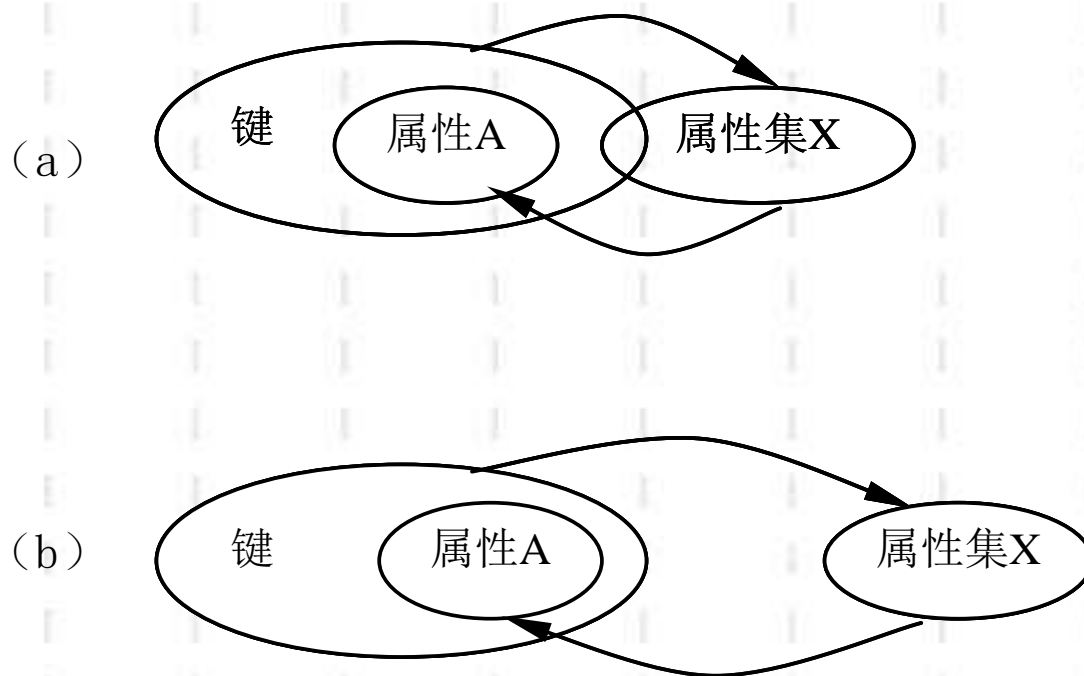


图6.16 主属性对候选键的传递依赖

BCNF

- 上述定义说明，若关系模式 $R\langle U, F \rangle$ 中的每一个决定因素都包含键，则 $R \in \text{BCNF}$ 。
- 这就是说，在BCNF中，除了键决定其它所有属性这样的函数依赖及其所蕴涵的函数依赖外，不再有其它非平凡的函数依赖，特别是不存在以不含键的属性组作为决定子(即左部)的非平凡的函数依赖。
- BCNF在概念上已足够“单纯”，就函数依赖而言，它已进行了所有必须的分解，并消除了增、删、改异常和数据冗余。

BCNF举例

- 例1： 关系模式SJP (S, J, P) 中， S表示学生， J表示课程， P表示名次； 每一个学生选修每门课程的成绩有一定的名次， 每门课程中每一名次只有一个学生（即没有并列名次）， 由语义可得到下面的函数依赖：

$$(S, J) \rightarrow P, (J, P) \rightarrow S$$

- (S, J) 与 (J, P) 都可以作为候选键。
- 这两个键各由两个属性组成, 而且它们是相交的
- 这个关系模式中显然没有非主属性对键的传递依赖或部分依赖， 所以SJP是3NF
- 而且除 (S, J) 与 (J, P) 以外没有其它决定因素， 所以SJP也是BCNF。

BCNF举例

- 例2 : 关系模式STJ(S, T, J)中, S表示学生, T表示教师, J表示课程。每一教师只教一门课。每门课有若干教师, 某一学生选定某门课, 就对应一个固定的教师。由语义可得到如下的函数依赖:

$$(S, J) \rightarrow T; (S, T) \rightarrow J; T \rightarrow J$$

- 这里 (S, J), (S, T) 都是候选键。
- 由于STJ中没有非主属性, 因此也不会任何非主属性对键传递依赖或部分依赖, 所以STJ是3NF。
- 但STJ不是BCNF, 因为T是决定因素, 而T不包含键。

BCNF举例

- 3NF的“不彻底”性表现在可能存在主属性对键的部分依赖和传递依赖。
- 任何非BCNF的关系模式都可以通过分解规范化为BCNF。
- 例如：STJ可分解为ST(S, T)与TJ(T, J)，并且它们都是BCNF，但注意这一分解已不保持函数依赖。

BCNF举例

- 例3：设有关系模式 $R(S, C, Z)$ ，其中 S 表示街道(street)， C 表示城市(city)， Z 表示邮编(zip)；由语义可以得到以下函数依赖集：

$$F = \{CS \rightarrow Z, Z \rightarrow C\}$$

- CS 是 R 的键
- 在 F 中， $Z \rightarrow C$ 的左部虽不是超键，但 C 是主属性，在3NF中允许这样函数依赖，而BCNF中则不允许
- 因此， R 属于3NF，但不属于BCNF。
- R 在被更新时仍可能发生异常，例如要插入一邮政编码与某个城市的对应关系，则必须知道该邮政编码所对应的一个街道。
- 我们可以将 R 分解为 $R_1(Z, C)$ 和 $R_2(S, Z)$ ，其中 R_1 和 R_2 都是BCNF，并且这一分解是无损的，但却不保持函数依赖。

BCNF举例

- 一般而言，属于3NF但不属于BCNF的关系并不多见
- 而且，即使出现这样的关系模式，其引起的更新异常也不是很严重。
- 例如，上面例子中的 $R(S, C, Z)$ ，虽然不属于BCNF，但基本上是“合理”的关系模式。
- 任何关系模式都可以分解为一组3NF，且既无损连接，又保持函数依赖。

第四范式(4NF)

- 定义4. 27: 设D是关系模式R上的FD和MVD集合。如果D中每个非平凡的MVD $X \twoheadrightarrow Y$ 的左部X都是R的超键, 则称R是4NF的模式。
- 4NF就是限制关系模式的属性之间不允许有非平凡且非函数依赖的多值依赖。
- 因为根据定义, 对于每一个非平凡的多值依赖 $X \twoheadrightarrow Y$, X都含有候选码, 于是就有 $X \rightarrow Y$, 所以4NF所允许的非平凡的多值依赖实际上是函数依赖。

第四范式(4NF)

- 显然，如果一个关系模式是4NF，则必为BCNF。但反之不然。
- 多值依赖的“毛病”在于数据冗余太大。
- 我们可以用投影分解的方法消去非平凡且非函数依赖的多值依赖。
- 函数依赖和多值依赖是两种最重要的数据依赖
- 如果只考虑函数依赖(FD)，则属于BCNF的关系模式规范化程度为最高。
- 如果考虑多值依赖(MVD)，则属于4NF的关系模式规范化程度为最高。

判断范式级别

- 给定关系模式和函数依赖集合，要求判断达到的最高范式。
- 方法：
 1. 求出给定关系的所有候选键(可能不止一个)；
 2. 根据键，确定主属性和非主属性；
 3. 判断是否满足第一范式(看属性的值域是否可以分解)；
 4. 判断是否满足第二范式(非主属性对键的部分函数依赖)；
 5. 判断是否满足第三范式(非主属性对键的传递函数依赖)；
 6. 判断是否满足BCNF范式(主属性对键的传递函数依赖)

判断范式级别：举例

例4-17: 已知关系R(A, B, C, D, E) 和 $F = \{AB \rightarrow CE, E \rightarrow AB, C \rightarrow D\}$, 试判断该关系最高为第几范式?

解:

1、关系R的候选键为: AB和E。

因为: $(AB)^+ = \{ABCED\}$

$(E)^+ = \{EABCD\}$

AB可以决定关系的所有属性, E可以决定关系的所有属性, 所以AB 和 E是关系的候选键

2、主属性为: A、B、E, 非主属性为: C、D

3、判断是否满足各个范式的要求:

- 因为R的所有的属性值域都是不可再分, 所以R已经在1NF中了
- 该关系为2NF, 因为C、D不存在对任何键的部分依赖
- 该关系不是3NF, 因为存在非主属性对键的传递依赖 $C \rightarrow D$
- R 2NF

判断范式级别： 举例

例4-18: 已知关系R (A, B, C, D, E, F) 和F={ $A \rightarrow B$, $C \rightarrow DF$, $AC \rightarrow E$, $D \rightarrow F$ }, 该关系最高为第几范式?

解:

- 该关系的键为AC, 因为 $(AC)^+ = \{ACBDFE\}$
- 主属性为: A、C
- 非主属性为: B, D, E, F
- 该关系明显满足1NF, 但是由于 $A \rightarrow B$, $C \rightarrow DF$ 存在非主属性对键的部分函数依赖, 所以该关系不是2NF
- 所以R满足的最高范式是1NF

判断范式级别：举例

- 例4-19：已知关系 $r(A, B, C, D)$ 和 $F = \{AB \rightarrow D, AC \rightarrow BD, B \rightarrow C\}$ ，试求出其最高范式？

解：该关系的键是：AB和AC

- 主属性是 $\{ABC\}$ ，非主属性是 $\{D\}$ 。
- D不部分依赖于任何键，所以该关系是2NF。
- 该关系也是3NF，因为只有一个非主属性，从而不可能存在两个不同非主属性间的传递依赖
- 该关系不是BCNF，因为有 $B \rightarrow C$ ，而B不是 r 的键。

规范化算法

- 下面分别介绍将关系模式规范化为BCNF和3NF的算法，首先给出两个引理。
- 引理4.5: 设 $R(U, F)$ 为关系模式， $\rho = \{R_1, R_2, \dots, R_k\}$ 是 R 的一个无损分解， $\tau = \{S_1, S_2, \dots, S_m\}$ 是 R_i 的一个无损分解，则 $\{R_1, \dots, R_{i-1}, S_1, \dots, S_m, R_{i+1}, \dots, R_k\}$ 也是 R 的一个无损分解。
- 证: 略
- 引理4.6: 设 $R(U, F)$ 为关系模式， $\rho = \{R_1, R_2, \dots, R_k\}$ 是 R 的一个无损分解，则：
 $\eta = \{R_1, R_2, \dots, R_k, R_{k+1}, \dots, R_n\}$ ($n > k$) 也是 R 的一个无损分解。
- 证: 略

分解成BCNF模式集的算法

- 算法4.5：无损联接地分解成BCNF模式集
- 输入：关系模式R, R上成立的函数依赖集F
- 输出：R的一个无损联接分解 $\rho = \{R_1, R_2, \dots, R_k\}$ ，满足每个 R_i 相对于 $\Pi_{R_i}(F)$ 是BCNF模式
- 方法：
 1. 置 $\rho = \{R\}$ ；
 2. 检查 ρ 中各关系模式是否为BCNF，若是，则转4；
 3. 设 ρ 中一个关系模式S不是BCNF
则S中存在函数依赖 $X \rightarrow A$ ， $A \in \setminus X$ ，且X不是S的键，
设 $S_1 = XA$ ， $S_2 = S - A$ ，用分解 $\{S_1, S_2\}$ 代替S，返回到；
 4. 分解结束，输出 ρ ；

分解成BCNF模式集的算法

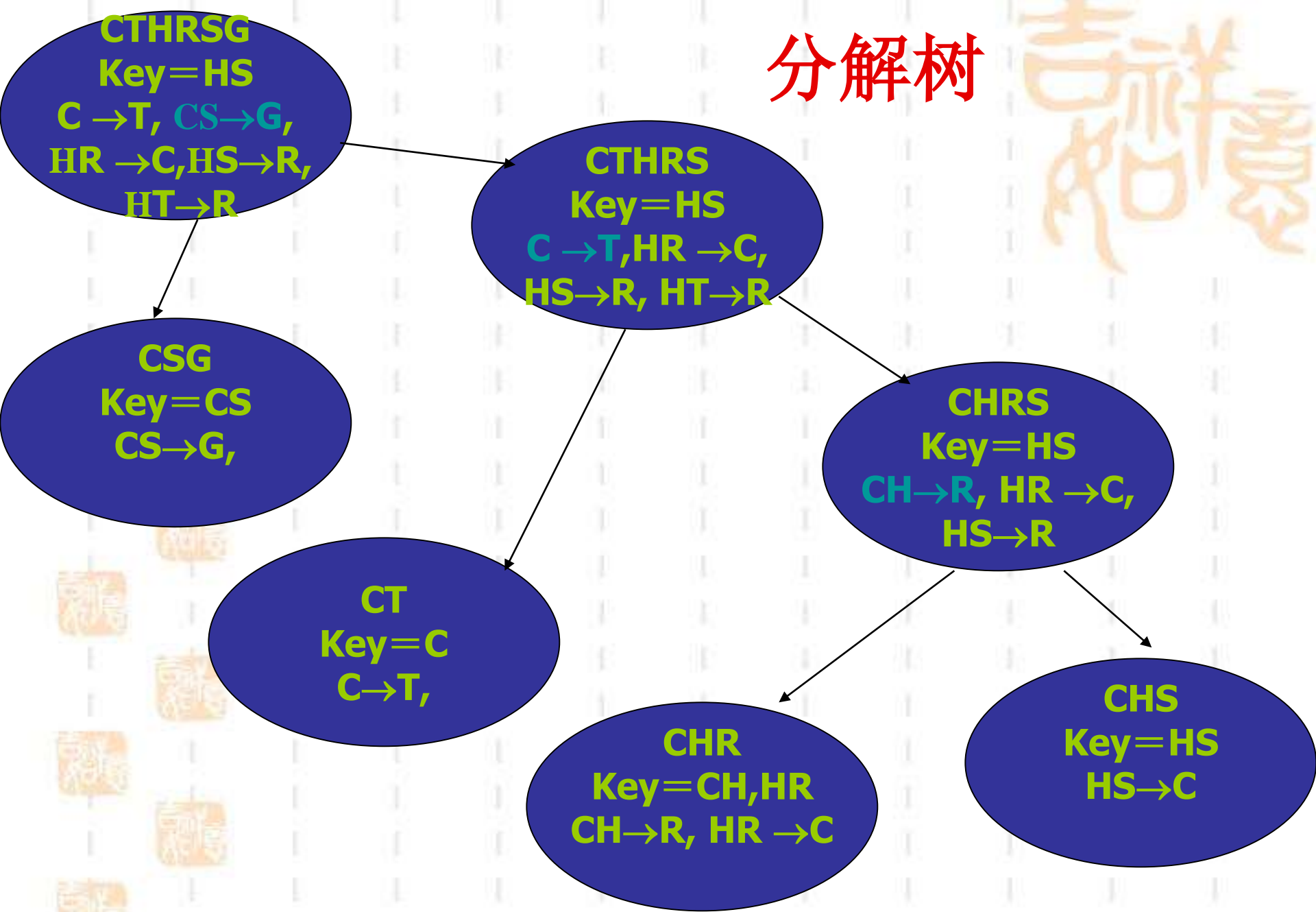
- 在上述分解过程中， $S_1(XA)$ 和 $S_2(XY)$ 中的属性都应是 U_S 的真子集，否则即 $XA = U_S$ ，由 $X \rightarrow A$ ，必有 X 是 S 的超键，与假设矛盾。
- 因此， ρ 每经过一次分解得到新的关系模式 $\{S_1, S_2\}$ 中的属性总必分解前的关系模式 (S) 中属性少，而当一个关系模式只包含两属性时，必为 BCNF。
- 所以，按照上述算法进行分解一定会终止，并将使 ρ 中的所有关系模式都成为 BCNF
- BCNF 分解的结果一般不唯一，这与选择分解的顺序有关。

分解成BCNF模式集的算法:举例

- 已知关系模式 $R(C, T, H, R, S, G)$, 其中:
- C : 表示课程 T : 表示教师 H : 表示时间
- R : 表示教室 S : 表示学生 G : 表示成绩
- $C \rightarrow T$: 每门课程仅有一名教师讲授
- $HR \rightarrow C$: 在任一时间, 每个教室只能上一门课程
- $HT \rightarrow R$: 在一个时间, 一个教师只能在一个教室上课
- $CS \rightarrow G$: 每个学生的每门课程只有一个成绩
- $HS \rightarrow R$: 在一个时间, 每个学生只能在一个教室内听课

键为HS

分解树



分解成BCNF模式集的算法:举例

- $R(C, T, H, R, S, G)$ 经过三次分解, 分解为:
 - $R_1(C, S, G)$: 学生的各门课程成绩
 - $R_2(C, T)$: 各门课的任课教师
 - $R_3(C, H, R)$: 每门课程的上课时间和每个时间的上课教室
 - $R_4(C, H, S)$: 每个学生的上课时间表
- 这些模式都是BCNF, 而且这些分解是无损分解
- 但是保持函数依赖吗?
- 分解成BCNF模式集的算法的缺点是涉及F+的计算, 这一计算的复杂性是指数型的, 有人证明判断一个关系模式是否属性BCNF是一个NP完全问题

分解成3NF模式集

- 算法4.6-1：将一个关系模式转化为一组3NF且保持依赖
- 输入：关系模式 $R(U, F)$
- 输出： R 的一个保持依赖的分解 $\sigma = \{R_1, \dots, R_k\}$ ，其中 R_i 为3NF
- 方法：
 1. 将 F 以其最小覆盖替代，仍记为 F
 2. 找出 R 中不在 F 中出现的属性，从 R 中分离出来组成一个关系模式 R_0
 3. 其余属性组成的关系模式仍记为 R
 4. 将 F 中的所有函数依赖按照左部相同的原则进行分组，并将每一组函数依赖 $\{X \rightarrow A_1, \dots, X \rightarrow A_n\}$ ($n \geq 1$) 所涉及的属性 $XA_1 \dots A_n$ 作为 σ 中的一个关系模式输出

分解成3NF模式集

- 算法4.6-1产生的分解只保持依赖，但不一定无损分解，所以要得到即保持依赖，又无损连接的分解必须进一步使用算法4.6-2
- 算法4.6-2：将一个关系模式转化为3NF且即无损又保持依赖
- 输入：关系模式 $R(U, F)$
- 输出： R 的一个既无损又保持依赖的分解 τ
 $= \{R_1, R_2, \dots, R_k\}$ ，其中 R_i 属于3NF
- 方法：
 1. 用算法4.5-1得到 R 的一个保持依赖的3NF分解 σ ，设 $\sigma = \{R_1, \dots, R_k\}$
 2. 如果经检验 σ 已是无损分解，则将 σ 作为 τ 输出，算法终止；
 3. 否则，求出 R 的一个键 X ，并令 $\tau = \sigma \cup \{R(X)\}$

分解成3NF模式集

- 定理4.7: 算法4.6可将一个关系模式转化为一组3NF且既无损又保持依赖。
- 证:
 1. 显然 τ 保持依赖
 2. R_i 为3NF (注意: $R(X)$ 必属于3NF)
 3. τ 是无损分解

分解成3NF模式集

- 用算法4.6得到的分解不一定是最小的关系模式集合，即如果有 $U_i \subseteq U_j$ ($i \neq j$)，则 R_i 是多余的，可以删除。
- 与算法4.4类似，用算法4.5得到的分解也不唯一。

分解成3NF模式集:举例

•例：现有一与学生有关的关系模式R(S#, Sname, Province, City, Street, Zipcode), 各属性分别表示学号、姓名、来自省份、城市、街道及邮政编码, R上有函数依赖集F = {S#→Sname, S#→Province, S#→City, S#→Street, S#→Zipcode, (Province, City, Street)→Zipcode, Zipcode→Province, Zipcode→City}, 试将其分解为3NF形式且保持无损连接及函数依赖。

分解成3NF模式集:举例

•解:

• (1) 首先计算F的最小覆盖Fmin, 因
S#→Zipcode是冗余的, 所以Fmin=
{S#→Sname, S#→Province, S#→City,
S#→Street, (Province, City,
Street)→Zipcode, Zipcode→Province,
Zipcode→City};

分解成3NF模式集:举例

•解:

(2) 由于R的所有属性均在Fmin中出现, 所以直接将函数依赖分组构成新的关系模式,

$p = \{R1(S\#, Sname, Province, City, Street), R2(Province, City, Street, Zipcode), R3(Zipcode, Province, City)\}$

, 因R3是R2的子集, 故将其略去, 得到

$p = \{R1(S\#, Sname, Province, City, Street), R2(Province, City, Street, Zipcode)\};$

分解成3NF模式集:举例

•解:

(3) 关系R上的候选键只有一个S#, 因其也是R1的子集, 所以不再加入中, 于是满足要求的最终关系模式集合仍为

$p = \{R1(S\#, Sname, Province, City, Street),$

$R2(Province, City, Street, Zipcode)\}$

分解成3NF模式集:举例

- 例：设 $R(A, B, C)$ $F = \{A \rightarrow C, B \rightarrow C\}$, 试将 R 转化为一组3NF且既无损又保持依赖
- 解：按保持函数依赖分解进行分组
$$\rho = \{ \{AC; A \rightarrow C\}, \{BC; B \rightarrow C\} \}$$
- 候选键为 AB
- $\tau = \rho \cup \{AB\}$
- 分解为：
$$\{ \{AC; A \rightarrow C\}, \{BC; B \rightarrow C\}, \{AB\} \}$$

基于MVD的关系规范化

- 以上讨论的关系规范化只考虑了函数依赖 (FD)，下面讨论基于多值依赖 (MVD) 的关系规范化。
- 例：考虑关系模式 $R(T, S, C)$ ，其中不存在任何非平凡的函数依赖，故 (T, S, C) 全键， R 已是 BCNF。
- 由于存在非平凡非FD的MVD： $T \twoheadrightarrow S | C$ ，这个关系仍存在数据冗余和更新异常等问题。
- 为此，应当将 R 进一步分解为 $R_1(T, S)$ ， $R_2(T, C)$ ，那么这样分解是否可行 (即无损)？

基于MVD的关系规范化

- 定理4.8: 设 $R(U, D)$ 为关系模式, D 为 R 的函数依赖和多值依赖集, $\rho = \{R_1, R_2\}$ 是 R 的一个分解, 则 ρ 为无损分解的充分必要条件是:

$$U_1 \cap U_2 \twoheadrightarrow (U_1 - U_2) \mid (U_2 - U_1) \in D^+$$

- 证:略

基于MVD的关系规范化

- 定义4. 28：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 $X \twoheadrightarrow Y$ ， X 都含有码，则称 R 为4NF。
- 4NF就是限制关系模式的属性之间不允许有非平凡且非函数依赖的多值依赖。
- 因为根据定义，对于每一个非平凡的多值依赖 $X \twoheadrightarrow Y$ ， X 都含有候选码，于是就有 $X \rightarrow Y$ ，所以4NF所允许的非平凡的多值依赖实际上是函数依赖。

基于MVD的关系规范化

- 显然，如果一个关系模式是4NF，则必为BCNF，但反之不然。
- 多值依赖的“毛病”在于数据冗余太大。
- 我们可以用投影分解的方法消去非平凡且非函数依赖的多值依赖。
- 函数依赖和多值依赖是两种最重要的数据依赖。如果只考虑函数依赖，则BCNF的关系模式规范化程度为最高。
- 如果考虑多值依赖，则4NF的关系模式规范化程度为最高。

规范化过程总结



规范化的实质

- 所谓规范化实质上是概念的“单一化”。
- 手段：消除不合适的数据依赖，使模式中的各关系模式达到某种程度的“分离”。
- 目的：控制数据冗余，避免插入、删除和修改异常，从而增强数据库结构的稳定性和灵活性。

关系模式设计的原则

- 好的关系模式应符合三条原则：
 - 表达性
 - 分离性
 - 最小冗余性

作业

- P202
- 6. 2
- 6. 6
- 6. 7

吉祥如意

