



数据库系统概论

An Introduction to Database System

第六章 关系数据理论



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



6.1 问题的提出

关系数据库逻辑结构设计

- 针对具体问题，如何构造一个适合于它的数据库模式，即应该构造几个关系，每个关系由哪些属性组成。
- 什么是一个好的数据库逻辑设计？
- 数据库逻辑设计的工具——关系数据库的规范化理论



第六章 关系数据理论

- 重点、难点
 - 函数依赖的概念
 - 范式的定义及判定
 - 关系的规范化
- 学习目标
 - 能够理解进行关系规范化的重要意义
 - 理解并掌握函数依赖的概念
 - 掌握范式的概念及判定方法，并能以此进行关系的分解和规范化
 - 掌握求解极小函数依赖的算法



概念回顾

- **关系**：描述实体、属性、实体间的联系。
 - 从形式上看，它是一张二维表，是所涉及属性的笛卡尔积的一个子集。
- **关系模式**：用来定义关系。
- **关系数据库**：基于关系模型的数据库，利用关系来描述现实世界。
 - 从形式上看，它由一组关系组成。
- **关系数据库的模式**：定义这组关系的关系模式的全体。



关系模式的形式化定义

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



关系模式的简化表示

- 由于D、DOM与模式设计关系不大，所以关系模式 $R(U, D, DOM, F)$ 可简化为一个三元组：

$$R(U, F)$$

- 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

1. 完整性约束的一种表现形式

- 限定属性的取值范围
- 定义属性值间的相互关连（主要体现与值的相等与否）即通过属性间值的相等与否来描述
- 它是数据库模式设计的关键



什么是数据依赖

2. 数据依赖

- 是通过一个关系内部属性间值的相等与否体现出来的数据间的相互关系
- 现实世界属性间相互联系的抽象
- 是数据内在的性质
- 语义的体现，普通存在于现实生活中



什么是数据依赖（续）

3. 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）



什么是数据依赖（续）

4. 数据依赖对关系模式的影响

- 不合适的数据依赖，造成插入异常、删除异常、更新异常和数据冗余问题



数据依赖对关系模式的影响

[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）、课程号（Cno）

成绩（Grade）

假设用单一的关系模式来存储这些信息：

Student <U、F>

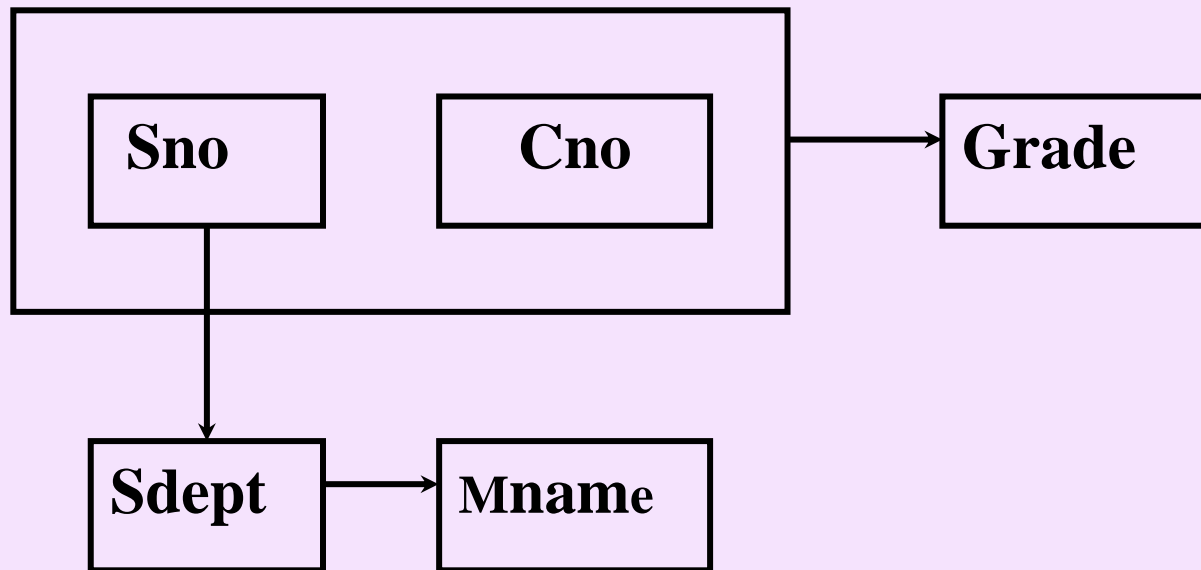
$$U = \{ Sno, Sdept, Mname, Cno, Grade \}$$



数据依赖对关系模式的影响（续）

根据现实语义，属性组U上的一组函数依赖F：

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





关系模式Student<U, F>中存在的问题

1. 数据冗余太大

- ◆ 数据在数据库中重复存放
- ◆ 每一个系主任的姓名重复存储，重复次数与该系所有学生的所有课程成绩出现次数相同
- ◆ 浪费大量的存储空间



关系模式Student<U, F>中存在的问题

2. 更新异常 (Update Anomalies)

- ◆ 数据冗余，更新数据时，维护数据完整性代价大
- ◆ 某系更换系主任后，必须修改与该系学生有关的每一个元组



关系模式Student<U, F>中存在的问题

3. 插入异常 (Insertion Anomalies)

- ◆ 无法插入某部分信息
- ◆ 如果一个系刚成立，尚无学生，则无法把这个系及其系主任的信息存入数据库



关系模式Student<U, F>中存在的问题

4. 删除异常 (Deletion Anomalies)

- ◆ 删除掉不应删除的信息
- ◆ 如果某个系的学生全部毕业了，则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。



数据依赖对关系模式的影响（续）

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



分解关系模式

- 把这个单一模式分解成3个关系模式：

$S(Sno, Sdept, Sno \rightarrow Sdept) ;$

$SC(Sno, Cno, Grade, (Sno, Cno) \rightarrow Grade) ;$

$DEPT(Sdept, Mname, Sdept \rightarrow Mname)$



五、数据依赖对关系模式的影响

- 例2：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下表所示。

职工姓名	职称	基本工资
A	讲师	4000
B	助教	3500
C	助教	3500
D	副教授	4500
E	讲师	4000
F	教授	5000



五、数据依赖对关系模式的影响

- 分析表存在以下问题：
 - 1、数据冗余大（数据在数据库中的重复存放称为数据冗余）
 - 2、插入与删除异常（无法插入某部分信息或删除掉不应删除的信息称为插入或删除异常）



五、数据依赖对关系模式的影响

■ 解决办法：

将表进行分解成表1和表2：

职工姓名	职称
A	讲师
B	助教
C	助教
D	副教授
E	讲师
F	教授

职称	基本工资
助教	3500
副教授	4500
讲师	4000
教授	5000

改进后的好处：

- 1、数据量减少
- 2、表达能力强
- 3、修改方便



6.2 规范化

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。



6.2.1 函数依赖

- 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



一、函数依赖

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 $X \rightarrow Y$ 。



一、函数依赖

[例]student(sno,sname,ssex,birthdate,sdept), 假设不允许重名, 则有:

Sno \rightarrow Ssex,

Sno \rightarrow birthdate

Sno \rightarrow Sdept,

Sno \longleftrightarrow Sname

Sname \rightarrow Ssex, Sname \rightarrow birthdate

Sname \rightarrow Sdept

但**Ssex \nrightarrow birthdate, Ssex \nrightarrow Sdept**

若**X \rightarrow Y**, 并且**Y \rightarrow X**, 则记为**X \longleftrightarrow Y**。

若**Y**不函数依赖于**X**, 则记为**X \nrightarrow Y**。



一、函数依赖

◆说明

1. 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。
2. 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。
 - 例如“姓名→年龄”这个函数依赖只有在不允许有同名人的条件下成立



说明

◆ 说明

3. 数据库设计者可以对现实世界作强制的规定。例如规定不允许同名的人出现，函数依赖“姓名→年龄”成立。所插入的元组必须满足规定的函数依赖，若发现有同名的人存在，则拒绝插入该元组。



二、平凡函数依赖与非平凡函数依赖

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： $(Sno, Cno) \rightarrow Sno$

$(Sno, Cno) \rightarrow Cno$



平凡函数依赖与非平凡函数依赖（续）

- 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。
- 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**）。



三、完全函数依赖与部分函数依赖

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 $X \xrightarrow{P} Y$ 。



完全函数依赖与部分函数依赖（续）

[例1] 中 $(Sno, Cno) \xrightarrow{F} Grade$ 是完全函数依赖,

$(Sno, Cno) \xrightarrow{P} Sdept$ 是部分函数依赖

因为 $Sno \rightarrow Sdept$ 成立, 且 Sno 是 (Sno, Cno) 的真子集



四、传递函数依赖

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 $X \leftrightarrow Y$, 则 Z 直接依赖于 X 。

例: 在关系 $Std(Sno, Sdept, Mname)$ 中, 有:

$Sno \rightarrow Sdept, Sdept \rightarrow Mname$

$Mname$ 传递函数依赖于 Sno



6.2.2 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**候选码**（Candidate Key）。

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

注意：若 $K \xrightarrow{P} U$ ，则 K 称为超码（Surpkey）。候选码是最小的超码，即 K 的任意一个真子集都不是候选码。



码（续）

- 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**）



码（续）

[例2]

关系模式 $S(\underline{Sno}, Sdept, Sage)$ ，单个属性 Sno 是码，
 $SC(\underline{Sno}, \underline{Cno}, Grade)$ 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



外部码

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



属性间的联系与函数依赖的关系

- 如果属性集X、Y间的联系是**1:1**联系，则存在函数依赖： $X \leftrightarrow Y$
- 如果属性集X、Y间的联系是**m:1**联系，则存在函数依赖： $X \rightarrow Y$
- 如果属性集X、Y间的联系是**m:n**联系，则不存在函数依赖



6.2.3 范式

- 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：
 - 第一范式(1NF)
 - 第二范式(2NF)
 - 第三范式(3NF)
 - BC范式(BCNF)
 - 第四范式(4NF)
 - 第五范式(5NF)



6.2.3 范式

- 各种范式之间存在联系：

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

- 某一关系模式R为第n范式，可简记为 **$R \in nNF$** 。

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



6.2.4 2NF

- 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模式

6.2.4 2NF

职工号	姓名	工资		
		基本工资	职务工资	工龄工资

具有组合数据项



职工号	姓名	职称	系名	系办公地址	学历	毕业年份
001	张三	教授	计算机	1--305	大学 研究生	1963 1982
002	李四	讲师	信电	2--204	大学	1989

具有多值数据项



2NF (续)

[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楼

- 函数依赖包括：

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

$Sno \rightarrow Sdept$

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

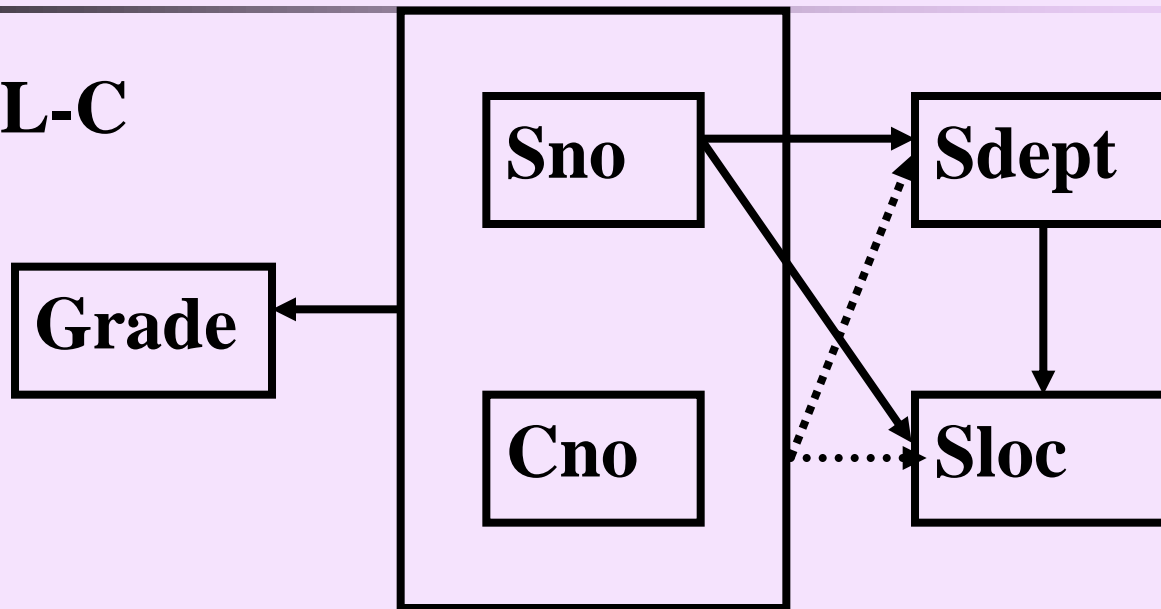
$Sno \rightarrow Sloc$

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

$Sdept \rightarrow Sloc$

2NF (续)

S-L-C



- S-L-C满足第一范式
- S-L-C的码为(Sno, Cno), 主属性: sno,cno; 非主属性: grade,sdept,sloc
- 非主属性Sdept、Sloc部分函数依赖于码 (sno,cno)



S-L-C不是一个好的关系模式（续）

(1) 插入异常

- ◆ 假设 $Sno=95102$, $Sdept=IS$, $Sloc=N$ 的学生还未选课, 即这个学生无 cno , 因课程号是主属性, 因此该学生的信息无法插入S-L-C。



S-L-C不是一个好的关系模式（续）

(2) 删除异常

- ◆ 假定某个学生只选修了3号课程这一门课。现在因身体不适，他连3号课程也不选修了，那么3号课程这个数据项就要删除。因课程号是主属性，此操作将导致该学生信息的整个元组都要删除，把不应删除的信息给删掉。



S-L-C不是一个好的关系模式（续）

(3) 数据冗余度大

- ◆ 如果一个学生选修了**10**门课程，那么他的**Sdept**和**Sloc**值就要重复存储了**10**次。

(4) 修改复杂

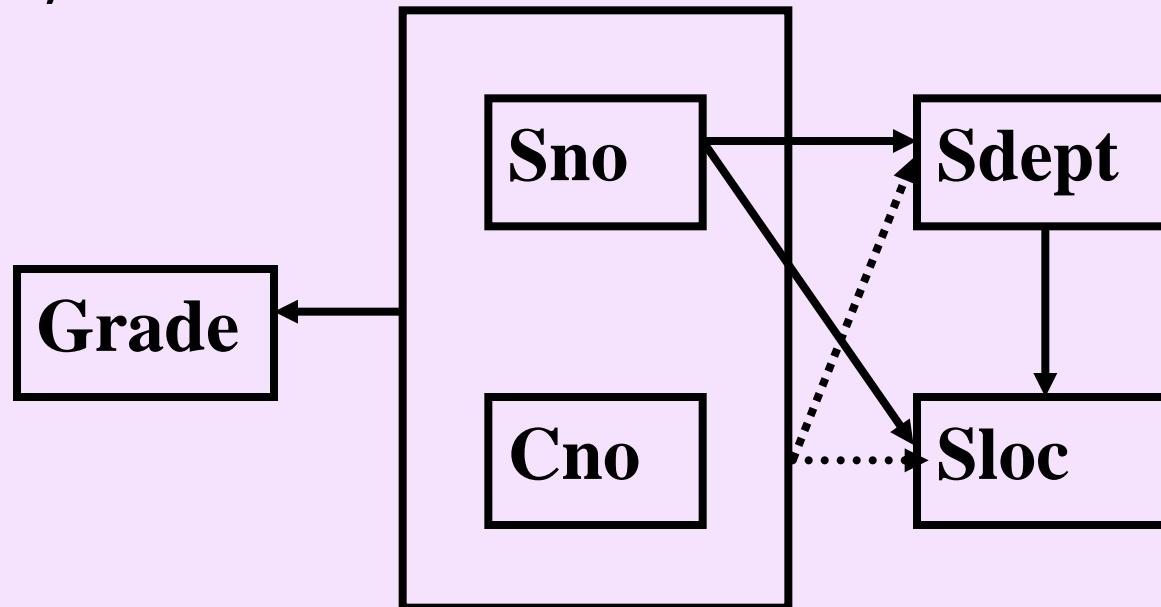
- 例如学生转系，在修改此学生元组的**Sdept**值的同时，还可能需要修改住处（**Sloc**）。如果这个学生选修了**K**门课，则必须无遗漏地修改**K**个元组中全部**Sdept**、**Sloc**信息，造成修改的复杂化。

满足第一范式的关系模式并不一定是一个**好的**关系模式

S-L-C不是一个好的关系模式（续）

- 原因

非主属性Sdept、Sloc部分函数依赖于码(sno,cno)





S-L-C不是一个好的关系模式（续）

- 解决方法

采用投影的方式把S-L-C分解为两个关系模式，
以消除这些部分函数依赖

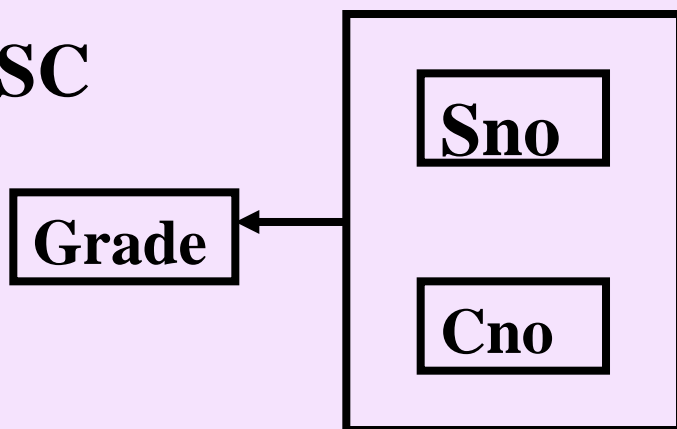
SC (Sno, Cno, Grade)

S-L (Sno, Sdept, Sloc)

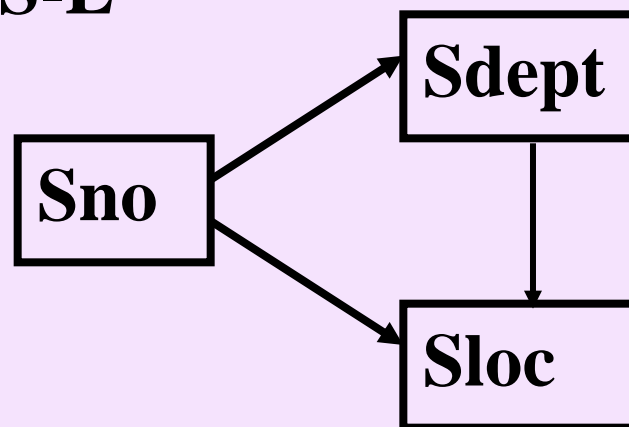
2NF (续)

函数依赖图:

SC



S-L



- ❖ 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



2NF (续)

- 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任何一个候选码，则 $R \in 2NF$ 。

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

S-L-C(Sno, Sdept, Sloc, Cno, Grade) $\notin 2NF$

SC (Sno, Cno, Grade) $\in 2NF$

S-L (Sno, Sdept, Sloc) $\in 2NF$



2NF (续)

对于SC (Sno, Cno, Grade) 和S-L (Sno, Sdept, Sloc)

- ◆ 由于学生选修课程的情况与学生的基本情况是分开存储在两个关系中的，在S-L关系中可以插入尚未选课的学生
- ◆ 删除一个学生的所有选课记录，只是SC关系中没有了关于该学生的记录，S-L关系中关于该学生的基本情况记录不受影响
- ◆ 不论一个学生选修了多少门课程，他的SDEPT和SLOC值只存储1次，大大降低了数据冗余
- ◆ 学生转系只需修改S-L中该学生元组的SDEPT值和SLOC值，由于SDEPT和SLOC并未重复存储，因此减化了修改操作



2NF（续）

- 采用投影分解法把S-L-C分解为两个关系模式：SC和S-L，消除了S-L-C中非主属性对码的部分函数依赖
- 一般地，如果把1NF关系模式通过投影分解方法，消除非主属性对码的部分函数依赖，分解为多个2NF的关系模式。



2NF (续)

- 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。
- 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。



2NF (续)

关系模式S-L(Sno, Sdept, Sloc)

■ 函数依赖:

$Sno \rightarrow Sdept$

$Sdept \rightarrow Sno$

$Sdept \rightarrow Sloc$

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



2NF (续)

分析关系模式S-L(Sno, Sdept, Sloc)中是否仍然存在下列问题？

- ◆数据冗余
- ◆插入异常
- ◆删除异常
- ◆更新复杂

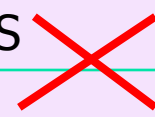


2NF (续)

◆ 插入异常

- 如果某个系（如**MA**）因种种原因（如刚刚成立），目前暂时没有在校学生，我们就无法把这个系的信息存入数据库

sno	sdept	sloc
2016010101	CS	N
2016010102	CS	N
2016010201	IS	M
NULL	MA	S





2NF (续)

◆ 删除异常

- 如果某个系（如**CS**）的学生全部毕业了，我们在删除该系学生信息的同时，会把这个系**CS**的信息也删除掉

sno	sdept	sloc
2016010101	CS	N
2016010102	CS	N
2016010201	IS	M
NULL	MA	S



2NF (续)

- ◆ 数据冗余度大

- ◆ 每个系的学生都住在同一个地方，关于系的住处的信息重复出现，**重复次数与该系学生人数**相同。

sno	sdept	sloc
2016010101	CS	N
2016010102	CS	N
2016010201	IS	M
NULL	MA	S



2NF (续)

◆ 更新复杂

- ◆ 学校调整学生住处时，由于关于每个系的住处信息是重复存储的，**修改时必须同时更新该系所有学生的SLOC属性值。**

sno	sdept	sloc
2016010101	CS	N
2016010102	CS	N
2016010201	IS	
NULL		

S-L不是一个好的关系模式



2NF (续)

- ◆ 原因:

- S-L中Sloc传递函数依赖于sno, 即非主属性传递函数依赖码

- 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:

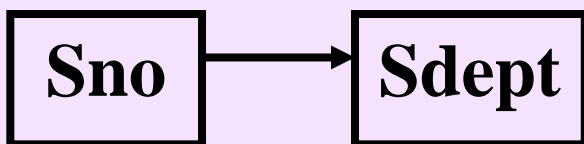
S-D (Sno, Sdept)

D-L (Sdept, Sloc)

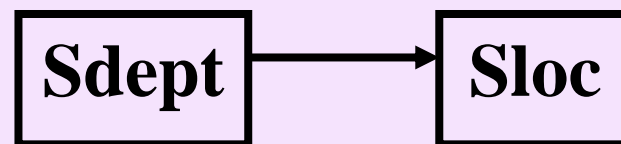


3NF (续)

S-D的码为Sno, D-L的码为Sdept



S-D



D-L

在分解后的关系模式中既没有非主属性对码的部分函数依赖，也没有非主属性对码的传递函数依赖，进一步解决了上述四个问题。



3NF (续)

◆ 异常的情况得到改善

- (1) D-L关系中可以插入系的信息，即使还没有在校学生
- (2) 某个系的学生全部毕业，只是删除S-D关系中相应的元组，D-L关系中关于该系的信息仍存在
- (3) 关于系的住处的信息只在D-L关系中存储一次
- (4) 当学习调整某个系的学生住处时，只需修改D-L关系中一个元组的Sloc属性值



6.2.5 3NF

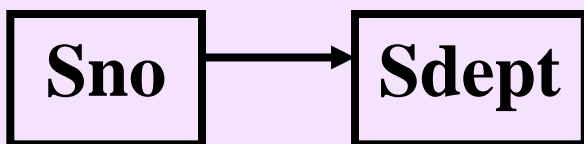
- 3NF的定义

定义**6.7** 设关系模式 $R\langle U, F \rangle \in 1NF$ ，若 R 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 $R\langle U, F \rangle \in 3NF$ 。

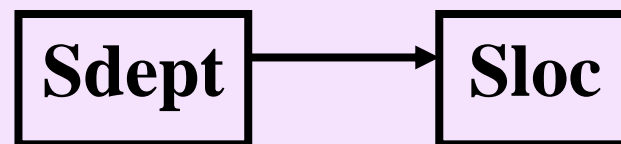
- 若 $R \in 3NF$ ，则每一个非主属性既不部分依赖于码也不传递依赖于码。

3NF (续)

S-D的码为Sno, D-L的码为Sdept

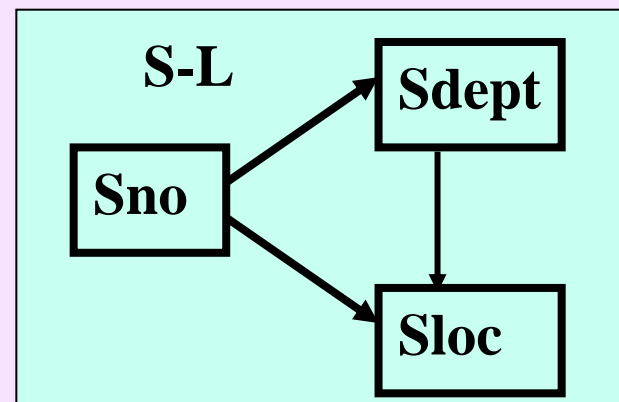


S-D



D-L

- ❖ $S-L(Sno, Sdept, Sloc) \in 2NF$
 $S-L(Sno, Sdept, Sloc) \notin 3NF$
 $S-D(Sno, Sdept) \in 3NF$
 $D-L(Sdept, Sloc) \in 3NF$





3NF（续）

- 3NF的一些性质：
 - 若 $R \in 3NF$ ，则R的**每一个非主属性**既不部分函数依赖于候选码也不传递函数依赖于候选码。
 - 如果 $R \in 3NF$ ，则R也是2NF。
 - 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。



3NF (续)

[例]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。

- ◆ 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
 - 函数依赖:
$$(S, J) \rightarrow T, (S, T) \rightarrow J, T \rightarrow J$$
 - (S, J)和(S, T)都是候选码



3NF (续)

- $STJ \in 3NF$
 - S, T, J都是主属性
 - 不存在非主属性对码的传递依赖或部分依赖
 - 分析 STJ 存在的问题?



3NF（续）

- **插入异常**

- 如果某个教师开设了某门课程，但尚未有学生选修，则有关信息无法存入数据库中

- **删除异常**

- 如果选修过某门课程的学生全部毕业了，在删除这些学生信息的同时，相应教师开设该门课程的信息也同时会被删除



3NF (续)

- **数据冗余度大**

- 虽然一个教师只教一门课，但每个选修该教师该门课程的学生元组都要记录这一信息

- **修改复杂**

- 思考题：列出一些数据，来说明这些异常情况。
- 某个教师讲授的课程，突然有学生选修了该教师该门课程的学生元组，需要进行相应修改



3NF (续)

- 原因

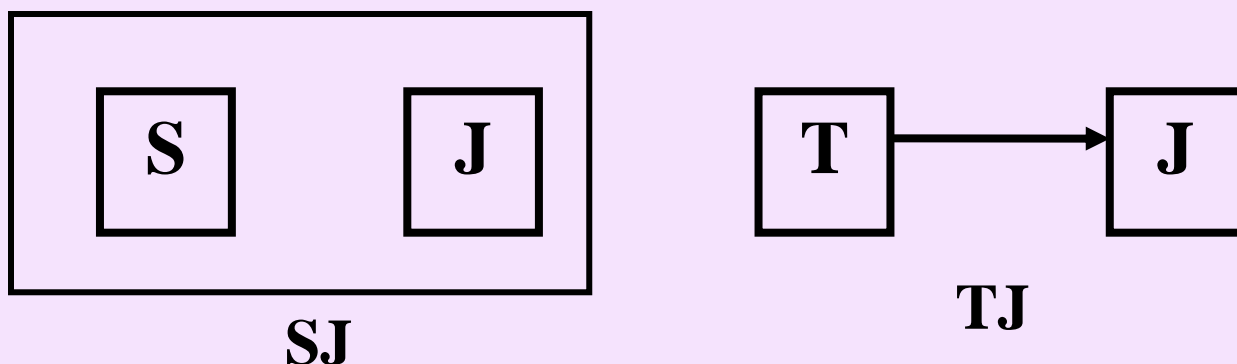
- 主属性J部分函数依赖于码 (S, T), $T \rightarrow J$

- 解决方法

- 采用投影分解法, 将STJ分解为两个关系模式
ST(S, J), TJ(T, J)



BCNF (续)



在分解后的关系模式中没有任何属性对码的**部分函数依赖和传递函数依赖**。它解决了上述四个问题：

- (1) **TJ**关系中可以存储所开设尚未有学生选修的教师信息
- (2) 选修过某门课程的学生全部毕业时，只是删除**SJ**关系中的相应元组，不会影响**TJ**关系中的元组
- (3) 关于，每个教师开设课程的信息只在**TJ**关系中存储一次
- (4) 某个教师开设的课程改名后，也只需修改**TJ**关系中的一个元组

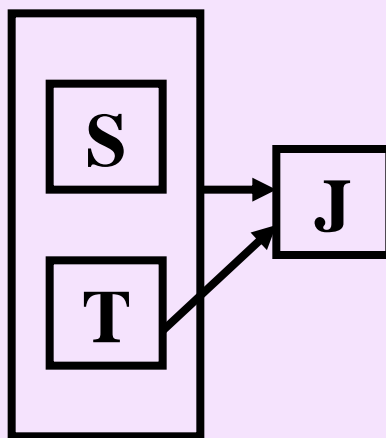
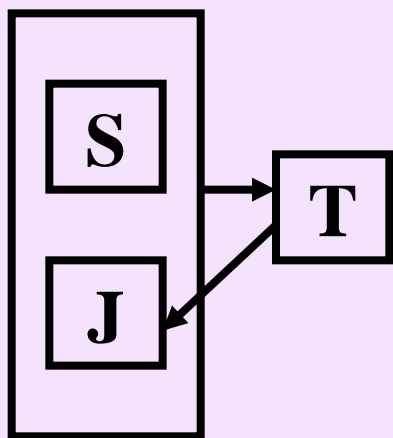


6.2.6 BC范式 (BCNF)

- **定义6.8** 关系模式 $R\langle U, F \rangle \in 1NF$ ，若 $X \rightarrow Y$ 且 $Y \not\subseteq X$ 时 X 必含有码，则 $R\langle U, F \rangle \in BCNF$ 。
- 等价于：在关系模式 $R\langle U, F \rangle$ 中，如果每一个决定属性因素都是码或包含码，则 $R\langle U, F \rangle \in BCNF$ 。
- BCNF也称为修正的（或扩充的）第三范式。

BCNF (续)

[例]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。



$STJ \notin BCNF$

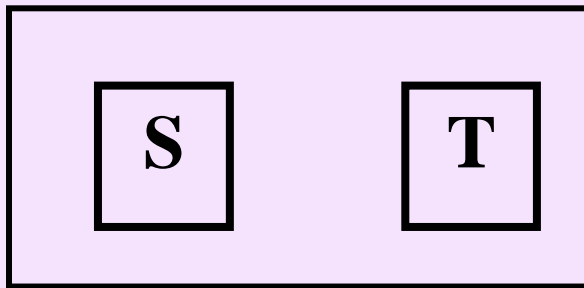
T是决定因素, T不包含码

STJ中的函数依赖



BCNF (续)

- $ST(S, T) \in \text{BCNF}, TJ(T, J) \in \text{BCNF}$



ST



TJ



BCNF (续)

- BCNF 的关系模式所具有的性质
 - 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



3NF与BCNF的关系

- 如果关系模式 $R \in \text{BCNF}$ ，必定有 $R \in 3\text{NF}$
- 如果关系模式 $R \in 3\text{NF}$ ，不一定有 $R \in \text{BCNF}$
- 如果 $R \in 3\text{NF}$ ，且 R 只有一个候选码，则 $R \in \text{BCNF}$
- 如果一个关系数据库中的所有关系模式都属于BCNF，那么在函数依赖范畴内，它已实现了模式的彻底分解，达到了最高的规范化程度，消除了操作异常诸多问题。



BCNF (续)

[例5] 关系模式C (Cno, Cname, Pcnno)

- $C \in 3NF$
- $C \in BCNF$

[例6] 关系模式S (Sno, Sname, Sdept, Sage)

- 假定S有两个码Sno, Sname
- $S \in 3NF$ 。
- $S \in BCNF$



BCNF (续)

[例7] 关系模式SJP (S, J, P)，其中S是学生，J是课程，P是学生课程成绩名次。

- 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
- $SJP \in 3NF$,
- $SJP \in BCNF$



6.2.7 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	{ 李 勇 王 军 }	{ 普通物理学 光学原理 物理习题集 }
数学	{ 李 勇 张 平 }	{ 数学分析 微分方程 高等代数 }
计算数学	{ 张 平 周 峰 }	{ 数学分析 }
⋮	⋮	⋮



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
...



多值依赖（续）

- Teaching \in BCNF
- Teaching具有唯一候选码(C, T, B), 即全码



多值依赖（续）

Teaching模式中存在的问题

- (1) 数据冗余度大
- (2) 插入操作复杂
- (3) 删除操作复杂
- (4) 修改操作复杂



存在
多值依赖



多值依赖（续）

■ 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)

对于一个（物理，光学原理）有一组 T 值 {李勇，王军}，这组值仅由课程 C 上的值（物理）决定，对于另一个（物理，物理习题集）对应的 T 值仍是 {李勇，王军}，因此 T 多值依赖于 C



多值依赖（续）

- 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， $Z=U-X-Y$ 。

交换元组中参考书对应的值，光学原理和物理习题集，但教员信息仍是一致的。



多值依赖（续）

- 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



多值依赖（续）

[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

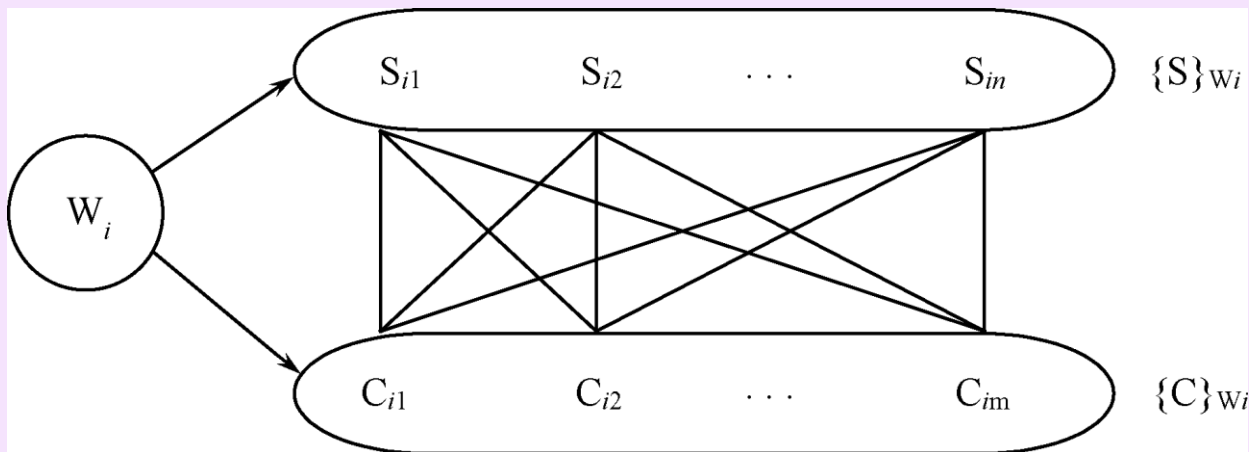


多值依赖（续）

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

多值依赖（续）

用下图表示这种对应



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



多值依赖的性质

(1) 多值依赖具有对称性

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

(2) 多值依赖具有传递性

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

(3) 函数依赖是多值依赖的特殊情况。

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

(4) 若 $X \twoheadrightarrow Y$, $X \twoheadrightarrow Z$, 则 $X \twoheadrightarrow Y \cup Z$ 。

(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的区别

(1) 多值依赖的有效性与属性集的范围有关

(2)

- 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



6.2.8 4NF

- **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 $X \twoheadrightarrow Y$ ($Y \subsetneq X$)， 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



4NF (续)

例: $\text{Teaching}(C, T, B) \notin 4NF$

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



6.2.9 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



规范化小结（续）

■ 关系模式规范化的基本步骤

消除决定属性
集非码的非平
凡函数依赖

1NF

↓ 消除非主属性对码的部分函数依赖

2NF

↓ 消除非主属性对码的传递函数依赖

3NF

↓ 消除主属性对码的部分和传递函数依赖

BCNF

↓ 消除非平凡且非函数依赖的多值依赖

4NF



规范化小结（续）

- 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



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

关系模式 $R \langle U, F \rangle$ 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。



定理 6.1 Armstrong推理规则是正确的

(I) 自反律: 若 $Y \subseteq X \subseteq U$, 则 $X \rightarrow Y$ 为 F 所蕴含

证: 设 $Y \subseteq X \subseteq U$

对 $R \langle U, F \rangle$ 的任一关系 r 中的任意两个元组 t ,

s :

若 $t[X] = s[X]$, 由于 $Y \subseteq X$, 有 $t[Y] = s[Y]$,

所以 $X \rightarrow Y$ 成立, 自反律得证



定理 6.1 Armstrong推理规则是正确的（续）

(2)增广律: 若 $X \rightarrow Y$ 为 F 所蕴含, 且 $Z \subseteq U$, 则 $XZ \rightarrow YZ$ 为 F 所蕴含。

证: 设 $X \rightarrow Y$ 为 F 所蕴含, 且 $Z \subseteq U$ 。

设 $R \langle U, F \rangle$ 的任一关系 r 中任意的两个元组 t, s :

若 $t[XZ] = s[XZ]$, 则有 $t[X] = s[X]$ 和 $t[Z] = s[Z]$;

由 $X \rightarrow Y$, 于是有 $t[Y] = s[Y]$, 所以 $t[YZ] = s[YZ]$,

所以

$XZ \rightarrow YZ$ 为 F 所蕴含, 增广律得证。



定理 6.1 Armstrong推理规则是正确的（续）

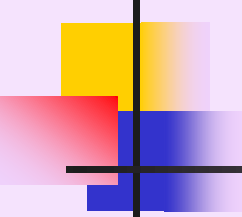
(3) 传递律：若 $X \rightarrow Y$ 及 $Y \rightarrow Z$ 为 F 所蕴含，则 $X \rightarrow Z$ 为 F 所蕴含。

证：设 $X \rightarrow Y$ 及 $Y \rightarrow Z$ 为 F 所蕴含。

对 $R \langle U, F \rangle$ 的任一关系 r 中的任意两个元组 t, s ：

若 $t[X] = s[X]$ ，由于 $X \rightarrow Y$ ，有 $t[Y] = s[Y]$ ；

再由 $Y \rightarrow Z$ ，有 $t[Z] = s[Z]$ ，所以 $X \rightarrow Z$ 为 F 所蕴含，传递律得证。



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)

- 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)



导出规则

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 ($i=1, 2, \dots, k$)



3. 函数依赖闭包

定义6.12 在关系模式 $R\langle U, F \rangle$ 中为 F 所逻辑蕴含的函数依赖的全体叫作 F 的闭包，记为 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



Armstrong公理系统

- Armstrong公理系统是有效的、完备的
 - 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



F的闭包

$$F = \{X \rightarrow Y, Y \rightarrow Z\}$$

$$F^+ = \{$$

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



关于闭包的引理

■ 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

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



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 $B = \{ A \mid (\exists V)(\exists W)(V \rightarrow W \in F \wedge V \subseteq X^{(i)} \wedge A \in W) \}$;
- (3) $X^{(i+1)} = B \cup X^{(i)}$



算法6.1

(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。

对于算法5.1, 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。



Example

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

- $A^+ = AB.$
- $C^+ = C.$
- $(AC)^+ = ABCD.$



Example

$U = \{A, B, C, D\}; A \rightarrow B, BC \rightarrow D.$
 $(AC)^+ = ABCD.$



函数依赖闭包

[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)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 $X^{(1)}=AB\cup CD=ABCD$ 。



函数依赖闭包

(2) 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 $AB \rightarrow C$, $B \rightarrow D$, $C \rightarrow E$, $AC \rightarrow B$, 于是 $X^{(2)} = X^{(1)} \cup BCDE = ABCDE$ 。

(3) 因为 $X^{(2)} = U$ ，算法终止

所以 $(AB)_F^+ = ABCDE$ 。



5. 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。



函数依赖集等价的充要条件

引理6.3 $F^+ = G^+$ 的充分必要条件是

$$F \subseteq G^+, \text{ 和 } G \subseteq F^+$$

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 $F^+ = G^+$ 。



函数依赖集等价

- 要判定 $F \subseteq G^+$ ，只须逐一一对 F 中的函数依赖 $X \rightarrow Y$ ，考察 Y 是否属于 $X_{G^+}^+$ 就行了。因此引理5.3 给出了判断两个函数依赖集等价的可行算法。



6. 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。
- (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。
- (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。



最小依赖集

[例2] 对于6.1节中的关系模式 $S\langle U, F\rangle$ ，其中：

$U = \{ \text{SNO}, \text{SDEPT}, \text{MN}, \text{CNAME}, \text{G} \}$,

$F = \{ \text{SNO} \rightarrow \text{SDEPT}, \text{SDEPT} \rightarrow \text{MN},$
 $(\text{SNO}, \text{CNAME}) \rightarrow \text{G} \}$

设 $F' = \{ \text{SNO} \rightarrow \text{SDEPT}, \text{SNO} \rightarrow \text{MN},$
 $\text{SDEPT} \rightarrow \text{MN}, (\text{SNO}, \text{CNAME}) \rightarrow \text{G},$
 $(\text{SNO}, \text{SDEPT}) \rightarrow \text{SDEPT} \}$

F 是最小覆盖，而 F' 不是。

因为： $F' - \{ \text{SNO} \rightarrow \text{MN} \}$ 与 F' 等价

$F' - \{ (\text{SNO}, \text{SDEPT}) \rightarrow \text{SDEPT} \}$ 也与 F' 等价

$F' - \{ (\text{SNO}, \text{SDEPT}) \rightarrow \text{SDEPT} \}$

$\cup \{ \text{SNO} \rightarrow \text{SDEPT} \}$ 也与 F' 等价



7. 极小化过程

定理6.3 每一个函数依赖集 F 均等价于一个极小函数依赖集 F_m 。此 F_m 称为 F 的最小依赖集

证:构造性证明, 依据定义分三步对 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 $Y=A_1A_2 \dots A_k, k > 2$,
则用 $\{ X \rightarrow A_j | j=1, 2, \dots, k \}$ 来取代 $X \rightarrow Y$ 。

引理6.1保证了 F 变换前后的等价性。



极小化过程

(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。

由于 F 与 $G = F - \{X \rightarrow A\}$ 等价的充要条件是 $A \in X_G^+$

因此 F 变换前后是等价的。



极小化过程

(3) 逐一取出 F 中各函数依赖 $FD_i: X \rightarrow A$,

设 $X = B_1 B_2 \dots B_m$,

逐一考查 B_i ($i=1, 2, \dots, m$),

若 $A \in (X - B_i)_F^+$,

则以 $X - B_i$ 取代 X 。

由于 F 与 $F - \{X \rightarrow A\} \cup \{Z \rightarrow A\}$ 等价的充要条件是 $A \in Z_F^+$,

其中 $Z = X - B_i$

因此 F 变换前后是等价的。



极小化过程

由定义，最后剩下的 F 就一定是极小依赖集。

因为对 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，因此剩下的 F 与原来的 F 等价。 证毕

- 定理6.3的证明过程 也是求 F 极小依赖集的过程



极小化过程

[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 不一定是唯一的它与对各函数依赖 FD_i 及 $X \rightarrow A$ 中 X 各属性的处置顺序有关



极小化过程

- 极小化过程(定理6.3的证明)也是检验 F 是否为极小依赖集的一个算法
 - 若改造后的 F 与原来的 F 相同, 说明 F 本身就是一个最小依赖集



极小化过程

- 在 $R\langle U, F \rangle$ 中可以用与 F 等价的依赖集 G 来取代 F
 - 原因：两个关系模式 $R_1\langle U, F \rangle$, $R_2\langle U, G \rangle$ ，如果 F 与 G 等价，那么 R_1 的关系一定是 R_2 的关系。反过来， R_2 的关系也一定是 R_1 的关系。



*6.4 模式的分解

- 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



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



模式的分解（续）

定义6.16 关系模式 $R\langle U, F \rangle$ 的一个分解：

$$\rho = \{ R_1\langle U_1, F_1 \rangle, R_2\langle U_2, F_2 \rangle, \dots, R_n\langle U_n, F_n \rangle \}$$

$U = \bigcup_{i=1}^n U_i$, 且不存在 $U_i \subseteq U_j$, F_i 为 F 在 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



模式的分解（续）

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. S-L分解为三个关系模式：
SN(Sno)
SD(Sdept)
SO(Sloc)
2. SL分解为下面二个关系模式：
NL(Sno, Sloc)
DL(Sdept, Sloc)
3. 将SL分解为下面二个关系模式：
ND(Sno, Sdept)
NL(Sno, Sloc)



具有无损连接性的模式分解

- 关系模式 $R\langle U, F \rangle$ 的一个分解 $\rho = \{ R_1\langle U_1, F_1 \rangle, R_2\langle U_2, F_2 \rangle, \dots, R_n\langle U_n, F_n \rangle \}$

若 R 与 R_1 、 R_2 、...、 R_n 自然连接的结果相等，则称关系模式 R 的这个分解 ρ 具有无损连接性（Lossless join）

- 具有无损连接性的分解保证不丢失信息
- 无损连接性不一定能解决插入异常、删除异常、修改复杂、数据冗余等问题



模式的分解（续）

第3种分解方法具有无损连接性

问题:这种分解方法没有保持原关系中的函数依赖

- SL中的函数依赖 $Sdept \rightarrow Sloc$ 没有投影到关系模式 ND、NL上



保持函数依赖的模式分解

设关系模式 $R\langle U, F \rangle$ 被分解为若干个关系模式

$$R_1\langle U_1, F_1 \rangle, R_2\langle U_2, F_2 \rangle, \dots, R_n\langle U_n, F_n \rangle$$

（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）



模式的分解（续）

4. 将SL分解为下面二个关系模式：

ND(Sno, Sdept)

DL(Sdept, Sloc)

这种分解方法就保持了函数依赖



模式的分解（续）

- 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。



模式的分解（续）

第1种分解方法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

第2种分解方法保持了函数依赖，但不具有无损连接性

第3种分解方法具有无损连接性，但未持函数依赖

第4种分解方法既具有无损连接性，又保持了函数依赖



模式的分解（续）

SL (Sno, Sdept, Sloc)

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

分解	无损连接性	函数依赖性
SN(Sno),SD(Sdept),SO(Sloc)		
NL(Sno,Sloc),DL(Sdept,Sloc)		
ND(Sno,Sdept),NL(Sno,Sloc)		
ND(Sno,Sdept),DL(Sdept,Sloc)		



判断一个分解的无损连接性

■ 算法5.2 判别一个分解的无损连接性

步骤：1、构造一个k行n列的表

2、检查每一个函数依赖，寻找相同的行

3、处理一遍后，如果有 $a_1a_2...a_n$ 则分解具有无损连接性。

终止算法，否则转(2)

例：已知 $R\langle U, F \rangle, U = \{A, B, C, D, E\},$

$F = \{AB \rightarrow C, C \rightarrow D, D \rightarrow E\}, R$ 的一个分解

$\rho = \{R_1(A, B, C), R_2(C, D), R_3(D, E)\}$ 判断是否具有无损连接性？



分解算法

- 算法5.3 （合成法）转换为3NF的保持函数依赖的分解。
- 例： 设 $R\langle U, F \rangle$, $U = \{A, B, C, D, E, G, H, I, J\}$
 $F = \{A \rightarrow B, AE \rightarrow G, CD \rightarrow A, CE \rightarrow D, BC \rightarrow D, A \rightarrow H\}$, 将 $R\langle U, F \rangle$ 分解成3NF且保持函数依赖。



分解算法

- 算法5.4 转换为3NF既有无损连接性又保持函数依赖的分解
- 例：设 $R\langle U, F \rangle$, $U = \{A, B, C, D, E, G, H, I, J\}$
 $F = \{A \rightarrow B, AE \rightarrow G, CD \rightarrow A, CE \rightarrow D, BC \rightarrow D, A \rightarrow H\}$, 将 $R\langle U, F \rangle$ 转换为3NF既有无损连接又保持函数依赖的分解。



分解算法

- 算法5.5 转换为BCNF的无损连接分解（分解法）
- 例：设 $R\langle U, F \rangle$, $U = \{A, B, C, D, E, G\}$
 $F = \{A \rightarrow B, AE \rightarrow G, CD \rightarrow A, CE \rightarrow D, BC \rightarrow D\}$, 将 $R\langle U, F \rangle$ 分解成BCNF具有无损连接性。



分解算法

- 若要求分解具有无损连接性，那么模式分解一定能够达到4NF。
- 若要求分解保持函数依赖，那么模式分解一定能够达到3NF，但不一定能够达到BCNF。
- 若要求分解既具有无损连接性，又保持函数依赖，则模式分解一定能够达到3NF，但不一定能够达到BCNF。



候选码的求解理论与算法

- 给定的关系模式 $R(A_1, A_2, \dots, A_N)$ 和函数依赖集 F ,
可将其属性分成4类:

L:仅出现在 F 的函数依赖左部的属性

R:仅出现在 F 的函数依赖右部的属性

N:在 F 的函数依赖左右两边均未出现的属性

LR:在 F 的函数依赖左右两边均出现的属性



候选码的求解理论与算法

- 定理1：对 $R\langle U, F \rangle$, 若 X 是L类属性, 则 X 必是 $R\langle U, F \rangle$ 的任一候选码成员。
- 推论1：若 X 是L类属性, 且 X^+ 包含了 R 的全部属性, 则 X 必为 R 的唯一候选码。
- 定理2：若 X 是R类属性, 则 X 不包含在任何候选码中。
- 定理3：若 X 是N类属性, 则 X 必包含在 $R\langle U, F \rangle$ 的任一候选码中。
- 推论2：如果 X 是 R 的L类和N类组成的属性集, 且 X^+ 包含 $R\langle U, F \rangle$ 的全部属性, 则 X 是 $R\langle U, F \rangle$ 的唯一候选码。



候选码的求解理论与算法

- 算法：求解 $R\langle U, F \rangle$ 的候选码



下课了。。。

研
究



休息一会儿。。。