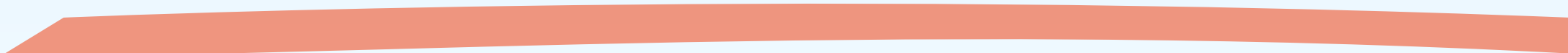


数据库系统

第6章 关系数据理论



胡 敏

jsjxhumin@hfut.edu.cn

第二篇 设计与应用开发篇

第六章 关系理论

关系规范化理论

第七章 数据库设计

步骤与方法

第八章 数据库编程

SQL、ODBC



第 6 章 关系数据理论

- 关系模型、关系数据库，关系数据库管理系统（RDBMS）。
- 关系数据库**逻辑设计**：关系模型——设计关系数据库，针对一个现实问题，如何选择一个比较好的关系模式的集合（构造几个关系），每个关系又应该由哪些属性组成。
- 关系数据库**规范化理论**：数据库逻辑设计的理论依据。
 - 要求了解规范化理论的研究动机及其在数据库设计中的作用，
 - 掌握函数依赖的有关概念，
 - 第一范式、第二范式、第三范式和BC范式的定义，
 - 重点掌握并能够灵活运用关系模式规范化的方法和关系模式分解的方法，这也是本章的难点。



第6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



6.1 问题的提出

关系数据库逻辑设计

问题——什么是一个好的数据库逻辑设计

- 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



问题的提出

[例1] 建立一个描述学校教务的关系模式：

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

■ 语义：

- 1. 一个系有若干学生， 但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。

■ 单一的关系模式： **Student (Sno, Sdept, Mname, Cno, Grade)**



问题的提出

■ 关系模式

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

简化为一个三元组:

$R(U, F)$

R: 关系名

U: 组成该关系的属性名集合

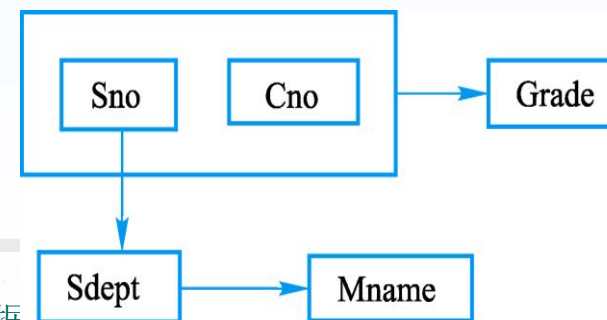
D: 属性组U中属性所来自的域的集合

DOM: 属性向域的映象集合

F: 属性间数据的依赖关系集合

Student(U,F)

| Sno | Sdept | Mname | Cno | Grade |
|-----------|-------|-------|-------|-------|
| 200201001 | 计算机系 | 李明 | C1 | 95 |
| 200201002 | 计算机系 | 李明 | C1 | 87 |
| 200201003 | 计算机系 | 李明 | C1 | 67 |
| 200201004 | 计算机系 | 李明 | C1 | 59 |
| 200201005 | 计算机系 | 李明 | C1 | 90 |
| | | | | |



什么是数据依赖

■ 数据依赖: 属性值间的相互关联（主要体现于值的相等与否）,是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

现实世界属性间相互联系的抽象，属于数据内在的性质



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)



关系模式Student表存在的问题

1. 数据冗余太大

| Sno | Sdept | Mname | Cno | Grade |
|-----------|-------|-------|-------|-------|
| 200201001 | 计算机系 | 李明 | C1 | 95 |
| 200201002 | 计算机系 | 李明 | C1 | 87 |
| 200201003 | 计算机系 | 李明 | C1 | 67 |
| 200201004 | 计算机系 | 李明 | C1 | 59 |
| 200201005 | 计算机系 | 李明 | C1 | 90 |
| | | | | |



关系模式Student表存在的问题

2. 更新异常(Update Anomalies)

数据冗余造成更新数据时维护数据的完整性代价高。

- 如某系更换系主任后，必须修改与该学生有关的每个元组；否则会造成数据不一致性。

| Sno | Sdept | Mname | Cno | Grade |
|-----------|-------|-------|-------|-------|
| 200201001 | 计算机系 | 李明 | C1 | 95 |
| 200201002 | 计算机系 | 李明 | C1 | 87 |
| 200201003 | 计算机系 | 李明 | C1 | 67 |
| 200201004 | 计算机系 | 李明 | C1 | 59 |
| 200201005 | 计算机系 | 李明 | C1 | 90 |
| | | | | |



关系模式Student表存在的问题

3. 插入异常(Insertion Anomalies)

■如某个系刚成立，尚无学生，就无法将系及系主任信息存入数据库的Student表中。

4. 删除异常(Deletion Anomalies)

■如某个系全部学生毕业了，在删除该系学生的同时，把系及其系主任信息也丢失了。

| Sno | Sdept | Mname | Cno | Grade |
|-----------|----------------|---------------|-------|-------|
| 200201001 | 计算机系 | 李明 | C1 | 95 |
| 200201002 | 计算机系 | 李明 | C1 | 87 |
| 200201003 | 计算机系 | 李明 | C1 | 67 |
| 200201004 | 计算机系 | 李明 | C1 | 59 |
| 200201005 | 计算机系 | 李明 | C1 | 90 |
| | | | | |
| NULL | 信息系 | 吴杰 | NULL | NULL |



数据依赖对关系模式的影响（续）

结论：

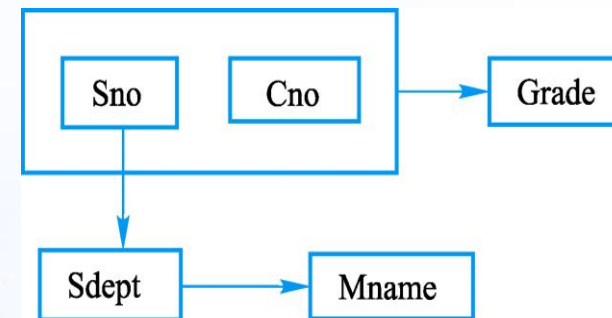
Student关系模式不是一个好的模式。

“好”的模式：

⇒ 无插入异常、删除异常、更新异常，数据冗余少

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



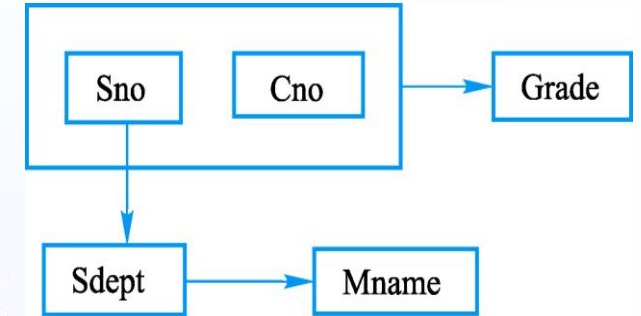
分解关系模式

把这个单一模式分成3个关系模式:

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

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

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



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。



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



6.2 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



6.2.1 函数依赖

- 函数依赖概念
- 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 $X \rightarrow Y$ 。

| Sno | Sdept | Mname | Cno | Grade |
|-----------|-------|-------|-------|-------|
| 200201001 | 计算机系 | 李明 | C1 | 95 |
| 200201002 | 计算机系 | 李明 | C1 | 87 |
| | | | | |

1. 所有关系实例均要满足
2. 语义范畴的概念：只能根据语义确定函数依赖
3. 数据库设计者可以对现实世界作强制的规定



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 $Y \not\subseteq X$, 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**(自然成立)

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）

- 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 $X \leftrightarrow Y$ 。
- 若 Y 不函数依赖于 X ，则记作 $X \nrightarrow Y$ 。



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 $X \xrightarrow{P} Y$ 。

例1: 单一的关系模式: **Student (U, F)**

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

[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 $X \longleftrightarrow Y$, 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



6.2.2 码

定义6.4 设K为R(U,F)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)。

■ 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(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



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。



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



6.2.3 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, , Boyce和Codd共同提出的范式)
 - 第四范式(4NF)
 - 第五范式(5NF)
$$1NF \supset 2NF \supset 3NF \supset BCNF \supset 4NF \supset 5NF$$
- 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**



1NF

■ 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
- 但满足第一范式的关系模式并不一定是一个好的关系模式



1NF (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：

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

$Sno \rightarrow Sdept$

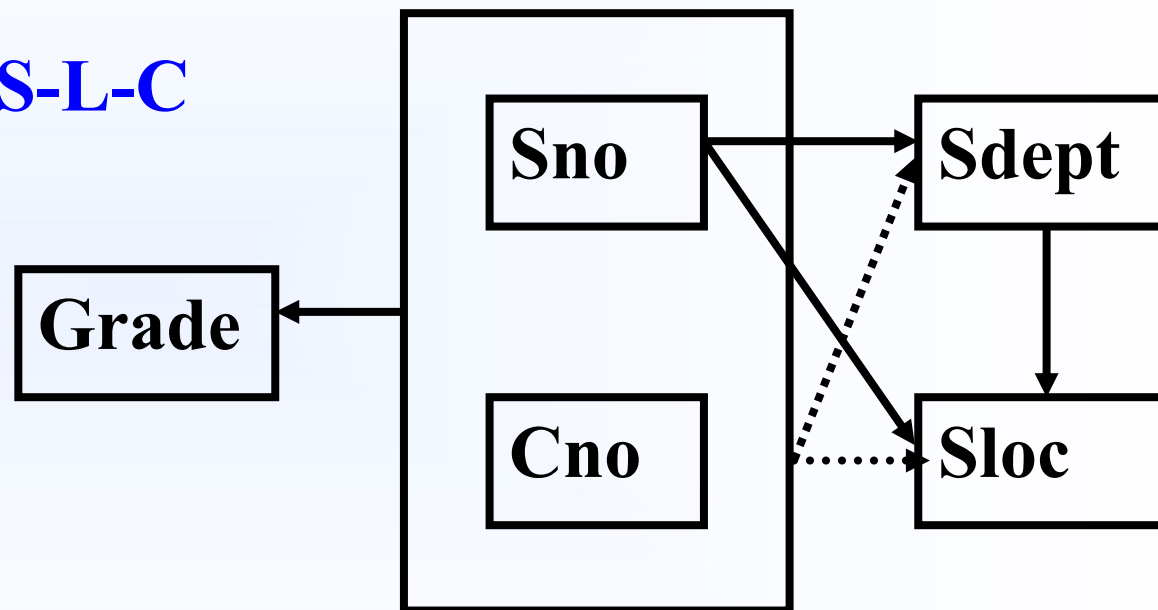
$(Sno, Cno) \xrightarrow{P} Sdept$

$Sno \rightarrow Sloc$

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

$Sdept \rightarrow Sloc$

S-L-C



■ 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不是一个好的关系模式（续）

(1) 插入异常

- 若插入一个学生Sno=2019010102,Sdept=Ma,Sloc=B01,但该生还没有选课, 即该学生无Cno, 这样的元组无法插入S-L-C。 (为什么?)

| Sno | Sdept | Sloc | Cno | Grade |
|------------|-------|------|------|-------|
| 2019010101 | IS | N | C3 | 89 |
| 2019010101 | IS | N | C2 | 97 |
| 2019010101 | IS | N | C5 | 88 |
| 2019010103 | IS | N | C1 | 86 |
| 2019010103 | IS | N | C3 | 92 |
| 2019010104 | IS | N | C3 | 79 |
| 2019010102 | Ma | B01 | null | null |



S-L-C不是一个好的关系模式（续）

(2) 删除异常

- 若假定某个学生只选择一门课程，如2019010104选择一门C3。现学生不选C3则将删除C3数据项，会造成学号信息也被删除。（为什么？）

| Sno | Sdept | Sloc | Cno | Grade |
|------------|-------|------|-----|-------|
| 2019010101 | IS | N | C3 | 89 |
| 2019010101 | IS | N | C2 | 97 |
| 2019010101 | IS | N | C5 | 88 |
| 2019010103 | IS | N | C1 | 86 |
| 2019010103 | IS | N | C3 | 92 |
| 2019010104 | IS | N | C3 | 79 |
| | | | | |



S-L-C不是一个好的关系模式（续）

(3) 修改复杂

若某个学生要求转系(IS→CS), 本来只需要修改此学生元组中的Sdept分量, 但还必须修改学生住处分量Sloc。 (为什么?)

(4) 数据冗余大

如果该学生选修了K门课程, 则Sdept、Sloc重复存储了K次, 数据不仅冗余而且必须毫无遗漏地修改K个元组中的全部Sdept和Sloc属性值。

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)

| Sno | Sdept | Sloc | Cno | Grade |
|------------|-------|------|-----|-------|
| 2019010101 | IS | N | C3 | 89 |
| 2019010101 | IS | N | C2 | 97 |
| 2019010101 | IS | N | C5 | 88 |
| 2019010103 | IS | N | C1 | 86 |
| 2019010103 | IS | N | C3 | 92 |
| 2019010104 | IS | N | C3 | 79 |
| | | | | |

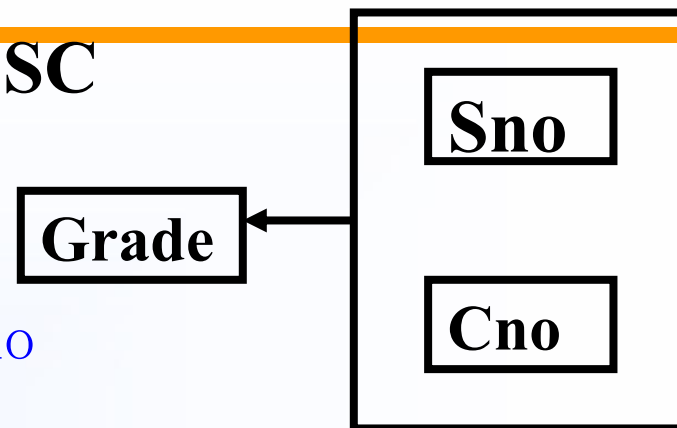


1NF (续)

| Sno | Cno | Grade |
|-----|-----|-------|
| | | |
| | | |

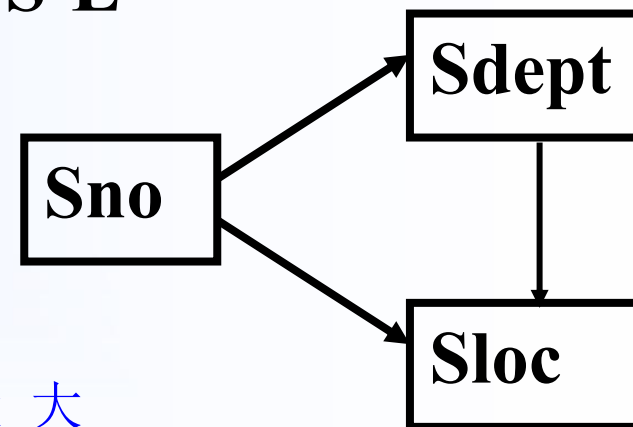
| Sno | Sdept | Sloc |
|-----|-------|------|
| | | |
| | | |

SC



- 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

S-L



- (1) 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。
- (2) 删除一个学生的所有选课记录，只是SC关系中没有关于该学生的记录了，S-L关系中关于该学生的记录不受影响。
- (3) 不论一个学生选多少门课程，他的Sdept和Sloc值都只存储1次。这就大大降低了数据冗余。
- (4) 学生转系只需修改S-L关系中该学生元组的Sdept值和Sloc值，由于Sdept、Sloc并未重复存储，因此减化了修改操作。



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 $R \in 2NF$ 。

分解实例

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

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

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

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



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)

■ 2NF的缺点:

➤ 数据冗余

每个系名和住处的名字存储次数等于该系学生人数。

Student (Sno, Sdept, Sloc, Cno, Grade) \notin 2NF



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

SD (Sno, Sdept, Sloc) \in 2NF



2NF (续)

■ 2NF的缺点: $SC(Sno, Cno, Grade) \in 2NF$

$SD(Sno, Sdept, Sloc) \in 2NF$

| Sno | Sdept | Sloc |
|-------------|-------|------|
| 2014101 | IS | N |
| 2014102 | IS | N |
| 2014103 | IS | N |
| 2014104 | IS | N |
| <u>null</u> | MA | S |
| | | |

➤ 插入异常

当一个新系没有招生时, 该系信息无法插入。

➤ 删除异常

如某系全部毕业而无招生时, 删除全部学生信息也随之删除了该系信息。

➤ 更新异常

如更换系住处时, 仍需改动较多学生的记录。



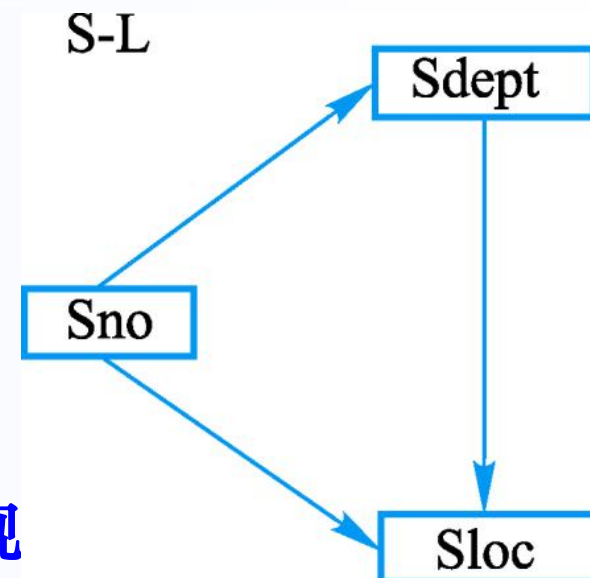
2NF (续)

■ 2NF的缺点产生的原因:

- SD (Sno, Sdept, Sloc) 中存在非主属性对主键的传递依赖。

SD (Sno, Sdept, Sloc) \in 2NF

Sno \rightarrow Sdept, Sdept \rightarrow Sloc 需要进一步对SD进行规范化



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



6.2.5 3NF

■ 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。

■ 3NF具有的性质(自行证明)

性质1: 如果 $R(U, F) \in 3NF$, 则 $R(U, F) \in 2NF$ 。

性质2: 如果 $R(U, F) \in 2NF$, 则 $R(U, F)$ 不一定是 3NF。



6.2.5 3NF

■ 3NF定义（由性质导出）

如果关系模式 $R \in 2NF$ ，且每个非主属性都不传递依赖于 R 的每个码，则 $R \in 3NF$ 。

- 第三范式（3NF）要求一个数据库表中不包含已在其它表中已包含的非主关键字信息。
- 属性不依赖于其它非主属性
- 不存在：关键字段 \rightarrow 非关键字段 $x \rightarrow$ 非关键字段 y



3NF (续)

例：2NF关系模式S-L(Sno, Sdept, Sloc)中

➤ 函数依赖：

$Sno \rightarrow Sdept$

$Sdept \nrightarrow Sno$

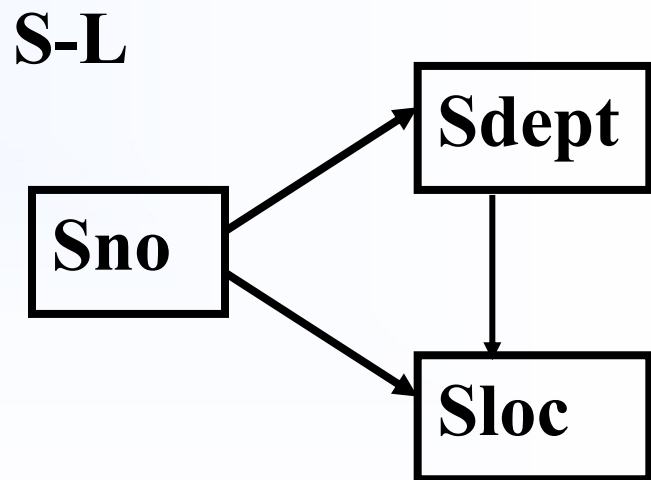
$Sdept \rightarrow Sloc$

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， $S-L \notin 3NF$

函数依赖图：



3NF (续)

■ 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)

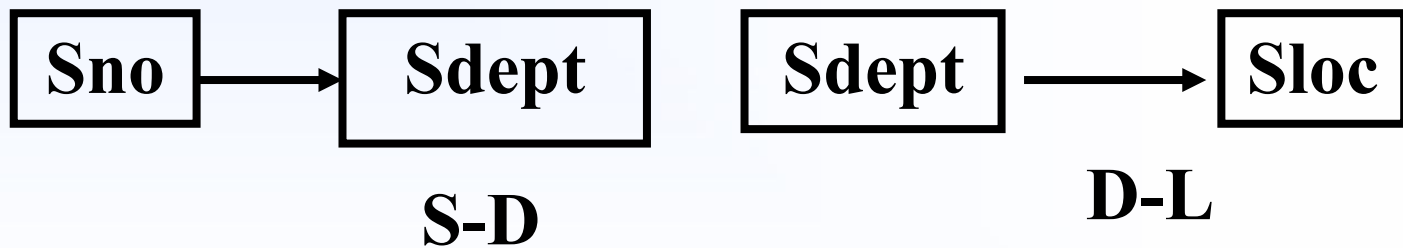
D-L (Sdept, Sloc)

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-D(Sno, Sdept) \in 3NF

D-L(Sdept, Sloc) \in 3NF



3NF (续)

■ 异常的情况得到改善:

| S-D | |
|-----|-------|
| Sno | Sdept |
| | |
| | |

| D-L | |
|-------|------|
| Sdept | Sloc |
| | |
| | |

(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了，只是删除**S-D**关系中的相应元组，**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) 当学校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。



3NF (续)

- 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，并不能完全消除关系模式中的各种异常情况和数据冗余。
- 3NF只限制了非主属性对键的依赖关系，而没有限制主属性对键的依赖关系。如果发生这种依赖关系，仍有可能存在各种异常情况和数据冗余。



一个例子

例：关系模式**STC** (**S**, **T**, **C**) 中，**S**表示学生，**T**表示教师，**C**表示课程
语义：

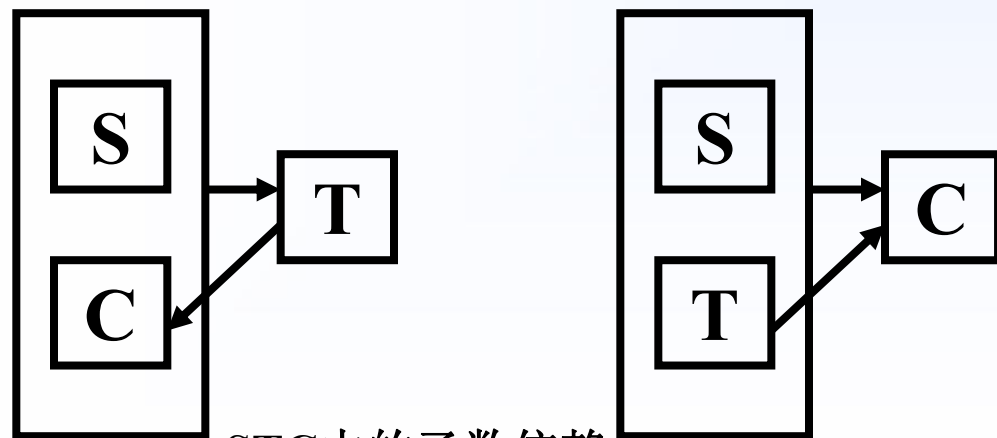
假设每一教师只教一门课 $T \rightarrow C$

每门课由若干教师教，但某一学生选定某门课，就确定了一个固定的教师

$(S, C) \rightarrow T$

某个学生选修某个教师的课就确定了所选课的名称 $(S, T) \rightarrow C$

$(S, C) \rightarrow T, (S, T) \rightarrow C, T \rightarrow C$



STC中的函数依赖

1. 候选码：(**S**, **C**) 和 (**S**, **T**)
2. **S** **T** **C** 都是主属性，不存在非主属性对码的部分函数依赖 和传递依赖
3. **STC** \in **3NF**



一个例子

虽然 $STC(S, T, C) \in 3NF$ ，但它仍存在增删改等异常，还不是一个理想的关系模式。

存在的问题：

(1) 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，则有关信息也无法存入数据库中。

(2) 删除异常

如果选修过某门课程的学生全部毕业了，则删除这些学生元组的同一时间相应教师开设该门课程的信息也同时丢掉了。

(3) 数据冗余度大

虽然一个教师只教一门课，却要记录这一信息。

(4) 修改复杂

某个教师开设的某门课程改名后，所有选修了该教师该课程的元组都要进行相应修改。

思考题：

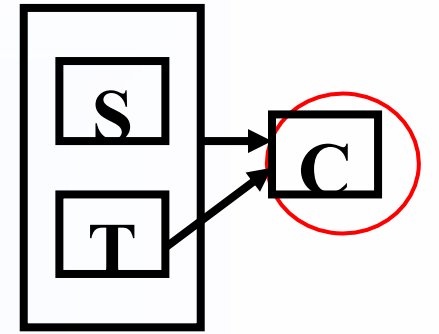
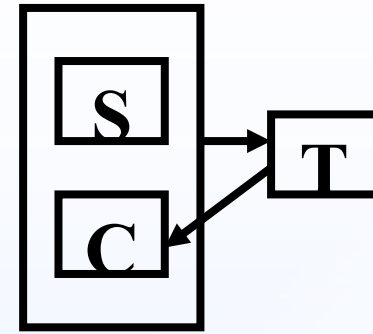
列出一些数据，验证这里列出的异常情况



一个例子

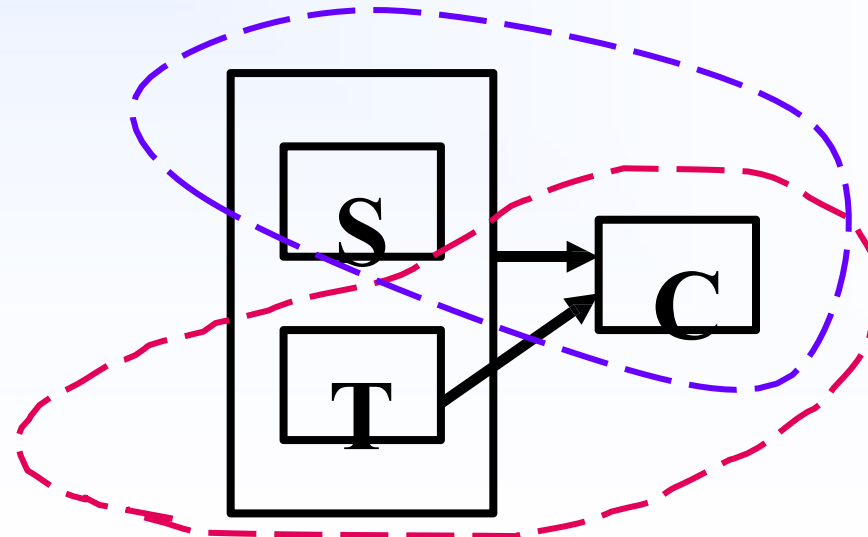
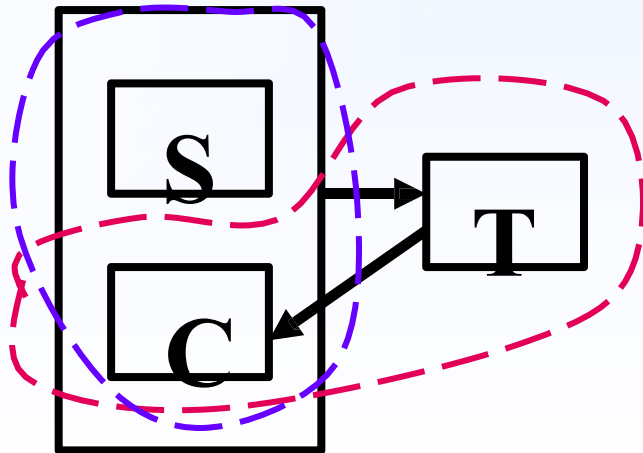
原因:

主属性C部分依赖于码: $(S,T) \xrightarrow{P} C$



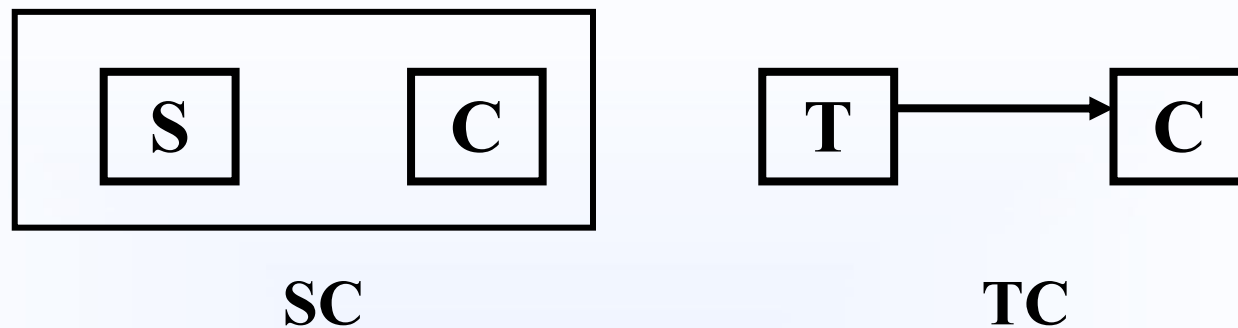
解决方法:

采用投影分解法, 将STC分解为二个关系模式: $SC(S, C)$; $TC(T, C)$



一个例子

❖ **ST的码为 (S, C)，TC的码为T。**



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) **TC**关系中可以存储所开课程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了，只是删除**SC**关系中的相应元组，不会影响**TC**关系中相应教师开设该门课程的信息。
- (3) 关于每个教师开设课程的信息只在**TC**关系中存储一次。
- (4) 某个教师开设的某门课程改名后，只需修改**TC**关系中的一个相应元组即可。



6.2.6 BC范式 (BCNF)

鲍依斯-科得范式 (Boyce Codd Normal Form -BCNF是3NF的改进形式-扩展的第三 范式。)

- **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: 每一个决定属性因素都包含码。

若关系模式 R 是**第一范式**, 且每个属性都不传递依赖于 R 的**候选键**。这种关系模式就是**BCNF**模式。即在第三范式的基础上, 数据库表中如果不存在任何字段对任一候选关键字段的传递函数依赖则符合**BCNF**。



BCNF (续)

例: $STC (S, T, C) \in 3NF$

$SC (S, C) \in BCNF$

$TC (T, C) \in BCNF$

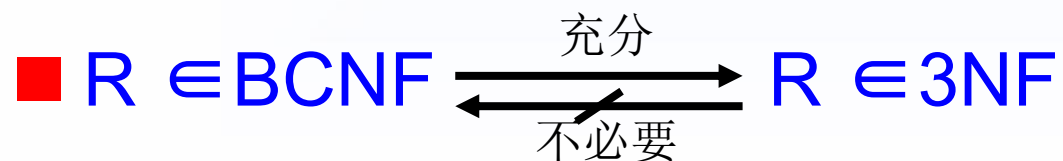
$T \rightarrow C$, $(S, C) \rightarrow T$, $(S, T) \rightarrow C$

SC 的码为 (S, C) , all-key

TC 的码为 T , $T \rightarrow C$

■ 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



BCNF (续)

- **性质1:** 若 $R \in \text{BCNF}$ ，将消除任何属性(主属性或非主属性)对键的部分函数依赖和传递函数依赖。也就是说，如果 $R \in \text{BCNF}$ ，则 $R \in 3\text{NF}$ 。

证明：采用反正法。设 R 不是 3NF ，则必然存在如下条件函数依赖， $X \rightarrow Y (Y \rightrightarrows X)$ ， $Y \rightarrow Z$ ，其中 X 是键属性， Y 是任意属性组， Z 是非主属性， $Z \notin Y$ ，则 $Y \rightarrow Z$ 函数依赖的决定因素 Y 可以不包含键，这与 BCNF 定义矛盾。命题得证。

- **性质2:** 若 $R \in 3\text{NF}$ ，则 R 不一定是 BCNF 。

思考



BCNF(续)

■ 例:

S_H_M (仓库ID, 存储物品ID, 管理员ID, 数量)且有一个管理员只在一个仓库工作; 一个仓库可以存储多种物品。这个数据库表中存在如下决定关系:

- (仓库ID, 存储物品ID) \rightarrow (管理员ID, 数量)
- (管理员ID, 存储物品ID) \rightarrow (仓库ID, 数量)

满足3NF, 但不满足BCNF: 存在关键字段决定关键字段

- (仓库ID) \rightarrow (管理员ID)
- (管理员ID) \rightarrow (仓库ID)
- 存在问题: (1) 删除异常: 当仓库被清空后, 所有"存储物品ID"和"数量"信息被删除的同时, "仓库ID"和"管理员ID"信息也被删除了。

(2) 插入异常: 当仓库没有存储任何物品时, 无法给仓库分配管理员。

(3) 更新异常: 如果仓库换了管理员, 则表中所有行的管理员ID都要修改

S_M (仓库ID, 管理员ID); S_h (仓库ID, 存储物品ID, 数量)



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SCP (S, C, P)

S-学生 C-课程 P-名次

语义：每个学生选修每门课程的成绩有一定的名次

每门课程的名次只有一个学生（无并列）

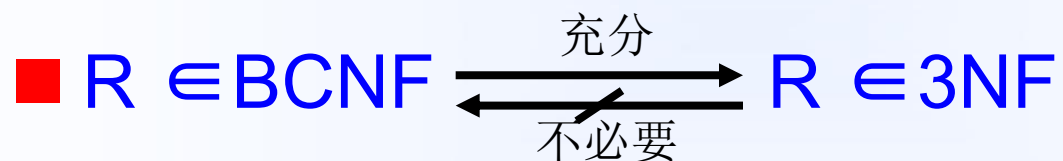
- 函数依赖： $(S, C) \rightarrow P$; $(C, P) \rightarrow S$
- (S, C) 与 (C, P) 都可以作为候选码,属性相交
- $SCP \in 3NF$,
- $SCP \in BCNF$



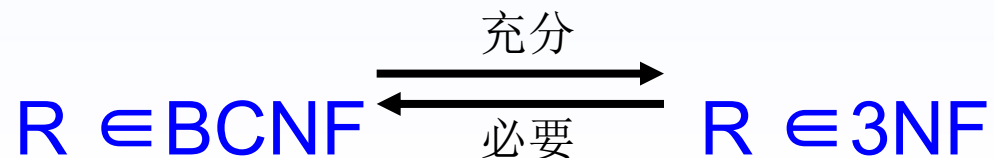
3NF与BCNF的关系

■ 3NF很大程度上消除了插入异常和删除异常，但可能存在主属性对候选键的部分依赖和传递依赖，因此关系模式的分解仍不够。

■ 在函数依赖范畴内，BCNF已经实现了模式的彻底分解，达到了最高的规范化程度，消除了插入异常和删除异常，而且数据的冗余也减少到了极小程度。



■ 如果 $R \in 3\text{NF}$ ，且 R 只有一个候选码



6.2.9 规范化小结

6.2.1 函数依赖

6.2.2 码

6.2.3 范式

4. 2NF

5. 3NF

6. BC范式 (BCNF)

*6.2.7 多值依赖

*6.2.8 4NF

6.2.9 规范化小结



6.2.9 规范化小结

- 一个关系模式只要其分量都是不可分的数据项，它就是规范化的关系模式，但这只是最基本的规范化。
- 规范化程度过低的关系模式不一定能够很好地描述现实世界，可能会存在插入异常、删除异常、修改复杂、数据冗余等问题，解决方法就是对其进行规范化，转换成高级范式。
- 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集合，这种过程就叫关系模式的规范化。
- 关系数据库的规范化理论是数据库逻辑设计的工具。



规范化小结（续）

■ 关系模式规范化的基本步骤



■ 规范化的基本思想:

- 逐步消除数据依赖中不合适的部分，使模式中的各关系模式达到某种程度的“分离”
- 采用“一事一地”的模式设计原则
 - 让一个关系描述一个概念、一个实体或者实体间的一种联系。
 - 若多于一个概念就把它“分离”出去。

■ 规范化实质上是概念的单一化



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



练习1

■ 问：关系模式R中的属性全部是主属性,则R的必定是_____。
3NF



练习2

- 如下表所示的关系R，下列选项中关于该关系模式属于第几范式判断正确的是（ ）。

| 零件号 | 单价 |
|-----|----|
| P1 | 25 |
| P2 | 8 |
| P3 | 25 |
| P4 | 9 |

D

- A、不是3NF B、是3NF但不是2NF
- C、是3NF但不是BCNF
- D、是BCNF

任何一个二元关系必定属于基于函数依赖最高范式BCNF。



下课了。。。。

研究



休息一会儿。。。。

