



# UNIT 11 规范化与函数依赖

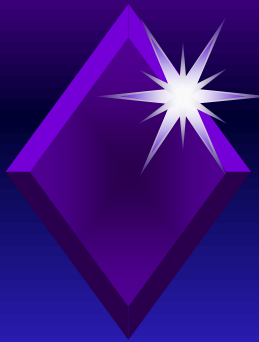


# 本讲主要目标



学完本讲后，你应该能够了解：

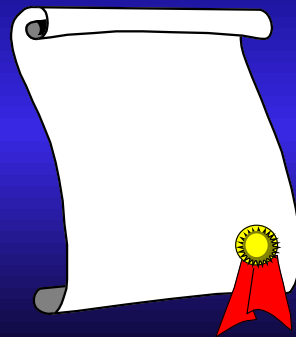
- 1、**规范化**是关系数据库逻辑设计的一种方法；
- 2、**一个不好的数据库设计存在更新异常**：插入异常、删除异常、数据冗余和修改复杂；而导致更新异常的原因是：在一个关系模式中存在某些函数依赖。
- 3、**规范化方法**就是将一个关系模式分解成多个模式，使得这些函数依赖不出现在同一个关系模式中的过程；
- 4、函数依赖、部分函数依赖、传递函数依赖的定义；
- 5、1NF、2NF、3NF、BCNF的定义，以及范式之间的包含关系；
- 6、对于一个关系模式，**判断**其最高属于第几**范式**，并使用基本的模式分解方法，将1NF分解为2NF，将2NF分解为3NF；
- 7、使用规范化方法设计数据库的逻辑结构时，并不是要使得到的数据库模式都达到最高范式，而是还需要平衡查询效率和更新代价。



# 本讲主要内容

---

- 一. 规范化的概念
- 二. 不好的数据库设计中的异常
- 三. 函数依赖
- 四. 部分函数依赖
- 五. 传递函数依赖
- 六. 候选键的形式定义
- 七. 范式：2NF、3NF和BCNF



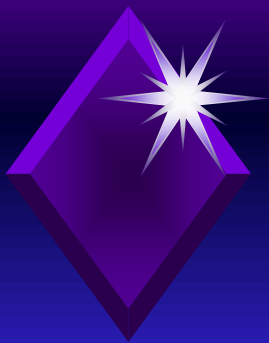


# 一、规范化的概念

---

(参见教材P180)

- 应用系统中数据库的设计就是要将现实世界表达成适当的模式——逻辑设计
- 层次、网状模型主要凭经验，无规则 and 理论
- 关系模型有严格数学理论基础，通过关系规范化理论来辅助逻辑设计
- 规范化理论：即将不规范的设计变成规范的设计



# 一、规范化的概念

## ◆ 规范化 (Normalization) (参见教材P192)

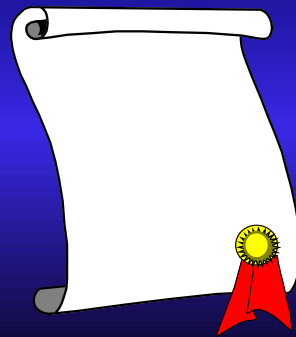
- 规范化是将属性分配给一个实体的过程，用以减少数据冗余和减少更新异常。
- 规范化过程实际上就是将一个低一级范式的关系模式，通过模式分解转换为若干个高一级范式的关系模式的集合的过程。
- 范式的转换过程是通过分析和消除属性间的数据依赖关系来实现的。
- 属性可分为主属性和非主属性。
  - 2NF, 3NF考察非主属性和键的关系,
  - BCNF考察主属性和键的关系。
- 属性间的依赖关系包括函数依赖和多值依赖。
  - 1NF, 2NF, 3NF, BCNF考察了函数依赖关系;
  - 4NF考察了多值依赖。

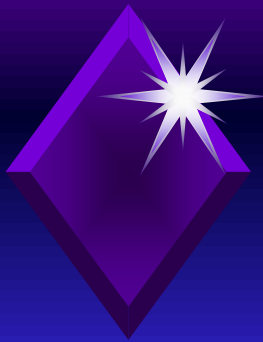


# 本讲主要内容

---

- 一. 规范化的概念
- 二. 不好的数据库设计中的异常
- 三. 函数依赖
- 四. 部分函数依赖
- 五. 传递函数依赖
- 六. 候选键的形式定义
- 七. 范式：2NF、3NF和BCNF





## 二、不好的数据库设计中的异常

---

举例1：现在要建立一个数据库来描述学生的借书情况，有这样一些属性：

借书证号 (S#)

姓名 (SN)

所在系 (SD)

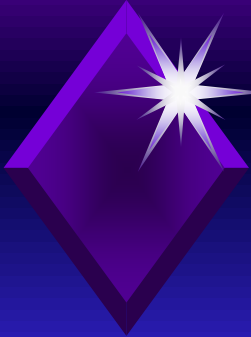
电话 (PHONE)

借阅的图书号 (B#)

借阅图书名 (BN)

借阅日期 (DATE)

可以用一个关系模式 S (S#, SN, SD, PHONE, B#, BN, DATE) 表示。



## 二、不好的数据库设计中的异常

假定有这样一个关系：

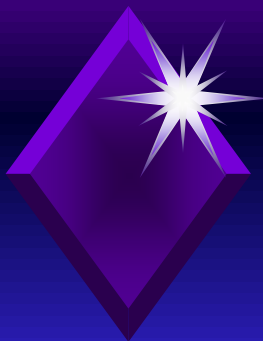
S#	SN	SD	PHONE	B#	BN	DATE
199813001	王铭	CS	87654321	B001	数据结构	20020403
199813001	王铭	CS	87654321	B002	数据库	20020403
199813001	王铭	CS	87654321	B003	离散数学	20020406
199813001	王铭	CS	87654321	B004	操作系统	20020406
199813001	王铭	CS	87654321	B005	心理学	20020420
199813001	王铭	CS	87654321	B006	人工智能	20020420
200112001	李力	MA	12345678	C001	哲学	20020508
199812002	张成	MA	12345678	B001	数据结构	20020509

设定主键为 (S#, B#, DATE)

该关系  
模式好  
吗？

?



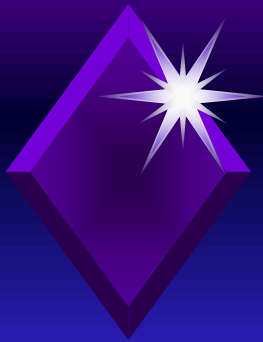


## 二、不好的数据库设计中的异常

S#	SN	SD	PHONE	B#	BN	DATE
199813001	王铭	CS	87654321	B001	数据结构	20020403
199813001	王铭	CS	87654321	B002	数据库	20020403
199813001	王铭	CS	87654321	B003	离散数学	20020406
199813001	王铭	CS	87654321	B004	操作系统	20020406
199813001	王铭	CS	87654321	B005	心理学	20020420
199813001	王铭	CS	87654321	B006	人工智能	20020420
200112001	李力	MA	12345678	C001	哲学	20020508
199812002	张成	MA	12345678	B001	数据结构	20020509

### 如果有更新操作：

- 一个学生借多本书
  - “王铭”转系
  - 新生注册但没借书
  - 还掉了借的所有书
- 冗余太大
  - 更新异常
  - 插入异常
  - 删除异常



## 二、不好的数据库设计中的异常

(内容参见教材P181-182)

### ➤ 数据冗余

—— 重复存储，浪费大量存储空间

### ➤ 更新异常 (修改复杂)

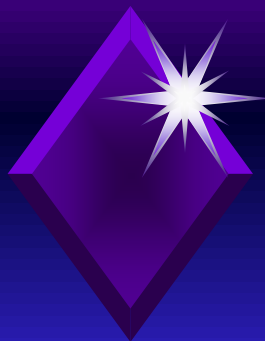
—— 如果更改表所对应的某个实体实例或者关系实例的单个属性时，需要进行多行更新，那么就说这个表存在更新异常

### ➤ 插入异常

—— 无法插入某个实体实例 (因为缺少某些属性的值)

### ➤ 删除异常

—— 如果删除表的某一个实体实例或关系实例时，导致另一个不同实体实例或关系实例的信息丢失。



## 二、不好的数据库设计中的异常

**解决办法** ----- 将该模式分解下列成三个模式：

S (S#,SN, SD,PHONE) , B (B#,BN) , SB (S#, B#, DATE)

S#	SN	SD	PHONE
199813001	王铭	CS	87654321
200112001	李力	MA	12345678
199812002	张成	MA	12345678

B#	BN
B001	数据结构
B002	数据库
B003	离散数学
B004	操作系统
B005	心理学
B006	人工智能
C001	哲学

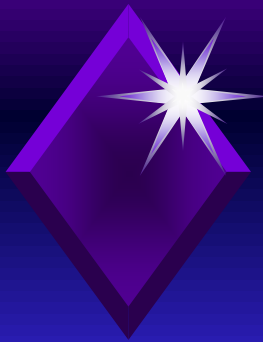
S#	B#	DATE
199813001	B001	20020403
199813001	B002	20020403
199813001	B003	20020406
199813001	B004	20020406
199813001	B005	20020420
199813001	B006	20020420
200112001	C001	20020508
199812002	B001	20020509

解决前面的  
问题了吗？

- 一个学生借多本书
- “王铭” 转系
- 新生注册但没借书
- 还掉了借的所有书



能保证不再  
存在其它的  
更新异常吗？



## 二、不好的数据库设计中的异常

---

### 举例2：

用一个关系模式S-L-C (S#, SD, SL, C#, G) 来描述学生的有关信息，

S#为学生的学号

SD为学生所在系

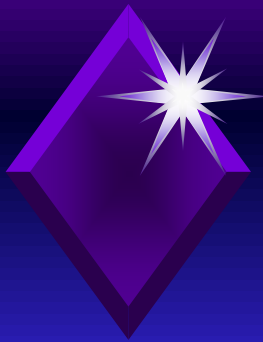
SL为学生的住处，每个系的学生住在同一个地方

C#为课程号

G为成绩

这里键为 (S#, C#) 。

**关系模式 S-L-C (S#, SD, SL, C#, G)**

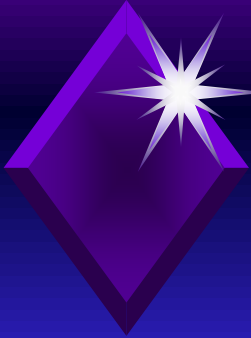


## 二、不好的数据库设计中的异常

---

分析关系模式S-L-C (S#, SD, SL, C#, G) 中的更新异常:

- |                            |      |
|----------------------------|------|
| ➤ 一个学生修很多课程                | 冗余太大 |
| ➤ 一个学生转系                   | 修改复杂 |
| ➤ 插入一个尚未选课的学生信息            | 插入异常 |
| ➤ 一个学生只选修了一门课,<br>但现在决定不选了 | 删除异常 |



## 二、不好的数据库设计中的异常

解决办法：将关系模式S-L-C (S#, SD, SL, C#, G)  
分解为两个：S-L (S#, SD, SL) 和S-C (S#, C#, G)

。。。S-C和S-L中消除了前面的更新异常了吗？

- 一个学生修很多课程
- 一个学生转系
- 插入一个尚未选课的学生信息
- 一个学生只选修了一门课，  
但现在决定不选了





## 二、不好的数据库设计中的异常

解决办法：将关系模式S-L-C (S#, SD, SL, C#, G) 分解为两个：S-L (S#, SD, SL) 和S-C (S#, C#, G)

。。。 S-C和S-L中去掉了所有更新异常吗？

分析模式S-L (S#, SD, SL) :

- 一个系有很多学生，且同系学生住在一个地方 ×
- 一个学生转系 ×
- 一个新系创建但新生尚未注册 ×
- 一个系所有学生毕业了 ×

## 二、不好的数据库设计中的异常

将模式S-L分解为：S-D (S#, SD) 和 D-L (SD, SL)

S-D和D-L中消除了S-L  
中的更新异常了吗？

- 一个系有很多学生，且同系学生住在一个地方 ✓
- 一个学生转系 ✓
- 一个新系创建但新生尚未注册 ✓
- 一个系所有学生毕业了 ✓



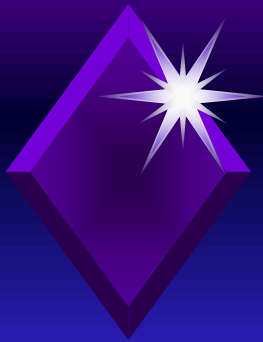
## 二、不好的数据库设计中的异常

将模式S-L进一步分解为：S-D (S#, SD) 和 D-L (SD, SL)

S-D和D-L中还存在其它更新异常吗？

不存在了，因为S-D与D-L都只是二目关系，已经足够小了

一般情况下，你能判断一个关系模式是否存在更新异常吗？



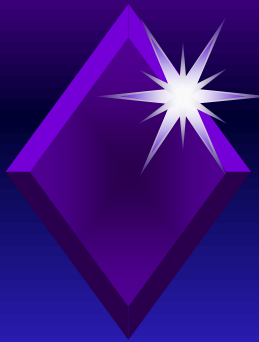
## 二、不好的数据库设计中的异常

---

出现更新异常的原因？

---- 在这单个模式中某些不好的数据依赖，  
如函数依赖，多值依赖

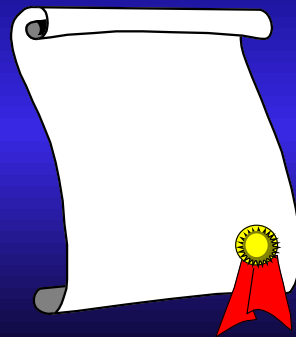
规范化方法就是将一个关系模式分解成多个模式，使得这些函数依赖不出现在同一个关系模式中的过程；



# 本讲主要内容

---

- 一. 规范化的概念
- 二. 不好的数据库设计中的异常
- 三. 函数依赖
- 四. 部分函数依赖
- 五. 传递函数依赖
- 六. 候选键的形式定义
- 七. 范式：2NF、3NF和BCNF

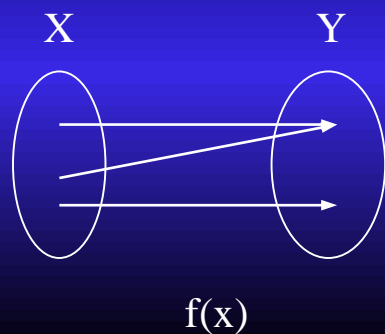


### 三、函数依赖 (参见教材P180-183)

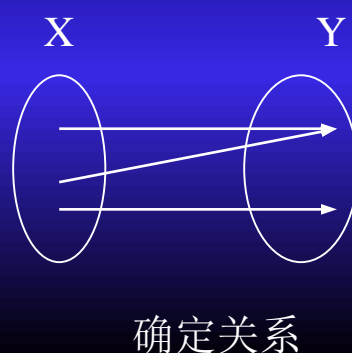
#### 1、函数依赖 (Functional dependency)

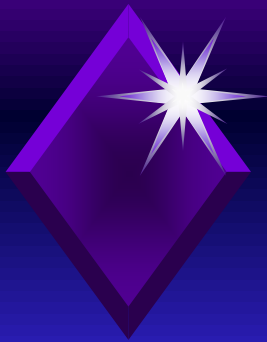
**函数依赖的定义：** 设 $R(U)$  是属性集 $U$ 上的关系模式。 $X, Y$ 是 $U$ 的子集。若对于 $R(U)$  的任意一个可能的关系 $r$ ,  $r$ 中不可能存在两个元组在 $X$ 上的属性值相等, 而在 $Y$ 上的属性值不等, 则称 $X$ 函数确定 $Y$ 或 $Y$ 函数依赖于 $X$ , 记作 $X \rightarrow Y$ 。

数学中的函数关系



规范化中的函数依赖



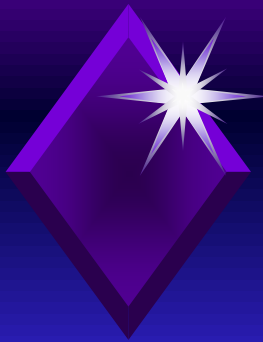


### 三、函数依赖

---

#### 2、函数依赖与属性间的关系有：

- 若 $X, Y$ 是1-1关系,  
则存在  $X \rightarrow Y$  或  $Y \rightarrow X$ 。如学号与借书证号
- 若 $X, Y$ 是  $m-1$ 关系,  
则存在  $X \rightarrow Y$  但  $Y \nrightarrow X$ 。如学号与姓名
- 若 $X, Y$ 是  $m-n$ 关系,  
则 $X, Y$ 间不存在函数依赖关系。如姓名与课程



### 三、函数依赖

#### 3、函数依赖另一种定义：

若R的任意关系有：对X中的每个属性值，在Y中都有惟一的值与之对应，则称Y函数依赖于X。

例：指出关系R中的函数依赖：

A	B	C	D
a1	b1	c1	d1
a1	b1	c1	d2
a1	b2	c2	d1
a2	b1	c3	d1

$AB \rightarrow C$  ?

$A \rightarrow C$  ?

$C \rightarrow A$  ?

$AB \rightarrow D$  ?

- ① 函数依赖是语义范畴的概念，只能通过语义确定
- ② 函数依赖要求对关系模式的所有关系都成立

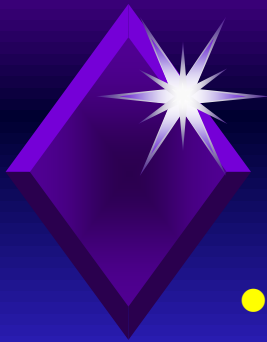


### 三、函数依赖

---

例：试指出学生关系S(SNO, SNAME, CLASS, CNO, TNO, TNAME, TAGE, ADDRESS, GRADE)中存在的函数依赖关系。

- SNO→SNAME (每个学号只能有一个学生姓名)
- SNO→CLASS (每个学号只能有一个班级)
- CNO→TNO (每门课程只有一个教师任教, 一个教师可教多门课)
- TNO →TNAME (每个教师只能有一个姓名)
- TNO→TAGE (每个教师只能有一个年龄)
- TNO→ADDRESS (每个教师只能有一个地址)



### 三、函数依赖

---

- $(SNO, CNO) \rightarrow GRADE$  (每个学生学习一门课只能有一个成绩)
- $(SNO, CNO) \rightarrow SNAME$
- $(SNO, CNO) \rightarrow CLASS$
- $(SNO, CNO) \rightarrow CNO$
- $(SNO, CNO) \rightarrow TNAME$
- $(SNO, CNO) \rightarrow TAGE$ 、
- $(SNO, CNO) \rightarrow ADDRESS$





### 三、函数依赖

---

#### 4、平凡、非平凡函数依赖

##### 决定因素

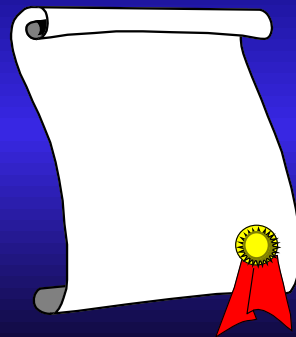
- $X \rightarrow Y$ , 但  $Y \not\subseteq X$  则称  $X \rightarrow Y$  是非平凡的函数依赖。
- $X \rightarrow Y$ , 但  $Y \subseteq X$  则称  $X \rightarrow Y$  是平凡的函数依赖。
- 若  $X \rightarrow Y$ , 则  $X$  叫做决定因素。
- 若  $X \rightarrow Y$ ,  $Y \rightarrow X$ , 则记作:  $X \rightleftarrows Y$ 。
- 如果  $Y$  不函数依赖于  $X$ , 则记做  $X \not\rightarrow Y$ 。



# 本讲主要内容

---

- 一. 规范化的概念
- 二. 不好的数据库设计中的异常
- 三. 函数依赖
- 四. 部分函数依赖
- 五. 传递函数依赖
- 六. 候选键的形式定义
- 七. 范式：2NF、3NF和BCNF





## 四、部分函数依赖

### 1、部分函数依赖 (Partial dependency)

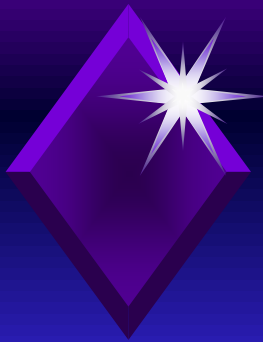
- **完全函数依赖**：在R (U) 中，如果 $X \rightarrow Y$ ，并且对于X的任何真子集 $X'$ ，都有 $X' \not\rightarrow Y$ ，则称Y对X完全函数依赖(full dependency)，记作：

$$X \xrightarrow{f} Y$$

- **部分函数依赖**：若 $X \rightarrow Y$ ，且存在X的真子集 $X'$ ，有 $X' \rightarrow Y$ ，记作：

$$X \xrightarrow{p} Y$$

- 左部为单属性的函数依赖一定是完全函数依赖。



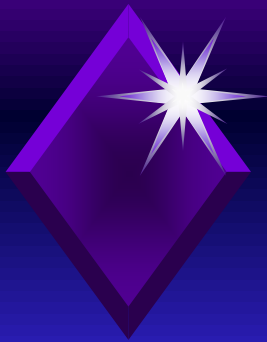
## 四、部分函数依赖

- 左部为多属性的函数依赖，如何判断其是否为完全函数依赖？

方法：取真子集，看其能否决定右部属性。

例如：试指出学生关系S(SNO, SNAME, CLASS, CNO, TNO, TNAME, TAGE, ADDRESS, GRADE)中存在的完全函数依赖和部分函数依赖。

- $SNO \rightarrow SNAME, SNO \rightarrow CLASS, TNAME \rightarrow TAGE, TNAME \rightarrow ADDRESS, CNO \rightarrow TNAME$  都是完全函数依赖。
- $(SNO, CNO) \rightarrow GRADE$  是一个完全函数依赖，  
因为  $SNO \not\rightarrow GRADE, CNO \not\rightarrow GRADE$ 。



## 四、部分函数依赖

---

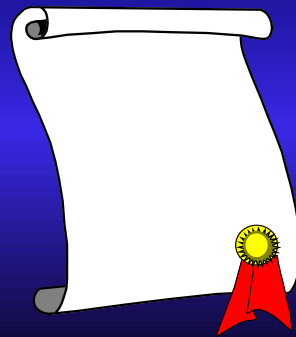
- $(SNO, CNO) \rightarrow SNAME, (SNO, CNO) \rightarrow CLASS,$   
 $(SNO, CNO) \rightarrow TNAME, (SNO, CNO) \rightarrow TAGE,$   
 $(SNO, CNO) \rightarrow ADDRESS$  都是部分函数依赖,  
因为:  $SNO \rightarrow SNAME, SNO \rightarrow CLASS, CNO \rightarrow TNAME$   
 $CNO \rightarrow TAGE, CNO \rightarrow ADDRESS$ 。



# 本讲主要内容

---

- 一. 规范化的概念
- 二. 不好的数据库设计中的异常
- 三. 函数依赖
- 四. 部分函数依赖
- 五. 传递函数依赖
- 六. 候选键的形式定义
- 七. 范式：2NF、3NF和BCNF





## 五、传递函数依赖

### 传递函数依赖的定义：

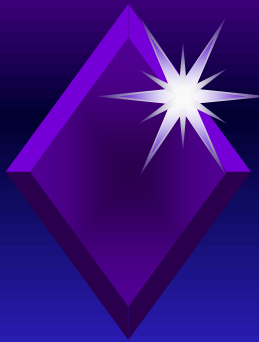
在 $R(U)$ 中，如果 $X \rightarrow Y$  ( $Y \not\subseteq X$ )， $Y \twoheadrightarrow X$ ，且 $Y \rightarrow Z$ ，  
则称 $Z$ 对 $X$ 传递函数依赖，记作：

$$X \xrightarrow{t} Z$$

例如：试指出学生关系 $S(SNO, SNAME, CLASS, CNO, TNO, TNAME, TAGE, ADDRESS, GRADE)$ 中存在的传递函数依赖。

解：因为 $CNO \rightarrow TNO$ ， $TNO \twoheadrightarrow TNAME$ ，所以 $CNO \rightarrow TNAME$ 是一个传递函数依赖。

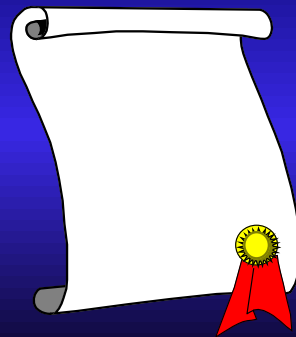
类似地， $CNO \rightarrow ADDRESS$ 也是一个传递函数依赖。



# 本讲主要内容

---

- 一. 规范化的概念
- 二. 不好的数据库设计中的异常
- 三. 函数依赖
- 四. 部分函数依赖
- 五. 传递函数依赖
- 六. 候选键的形式定义
- 七. 范式：2NF、3NF和BCNF







# 关系模式的形式定义

- ◆ 关系的描述称为关系模式可以形式化地表示为：

$$R(U, D, \text{dom}, F)$$

其中：

- R为关系名，
- U为组成该关系的属性名集合，
- D为属性组U中属性所来自的域，
- dom为属性向域的映象集合，
- F为属性间数据的依赖关系集合。

- ◆ 简化表示： $R(U, F)$

- ◆ 简化表示： $R(U)$  或  $R(A_1, A_2, \dots, A_n)$

其中： $A_1, A_2, \dots, A_n$ 为各属性名。



## 六、候选键的形式定义

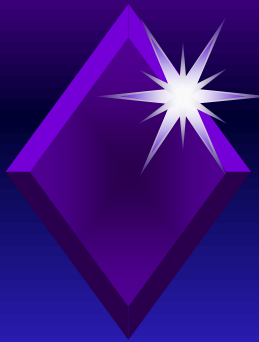
### ◆ 用函数依赖的概念来定义键：

设 $X$ 为 $R(U, F)$ 中的属性或属性组，若  $X \xrightarrow{f} U$   
则 $X$ 为 $R$ 的候选键。 (P183)

说明： $X \xrightarrow{f} U$

- $X \rightarrow U$   $X$ 能决定整个元组
- $X' \not\rightarrow U$   $X$ 中无多余的属性
- 存在性与不唯一性

思考，若  $X \xrightarrow{p} U$  则 $X$ 为 $R$ 的？



## 六、候选键的形式定义

---

### ◆ 术语：

- **主键**：选定候选键中的一个作为主键，一个关系的主键是唯一的
- **主属性**：包含在任何一个**候选键**中的属性
- **非主属性**：不包含在任何一个候选键中的属性
- **全键**：整个属性组为键

例：R(顾客，商品，日期)

S (S#, SN, SD, PHONE, B#, BN, DATE)

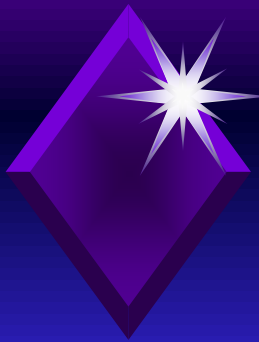


## 六、候选键的形式定义

**例：指出学生关系中的候选键、主属性和非主属性**

学号	姓名	性别	年龄	系名
S1	刘力	男	19	计算机
S2	李军	男	18	计算机
S3	王林	女	20	电子
S4	沈国立	男	19	电子

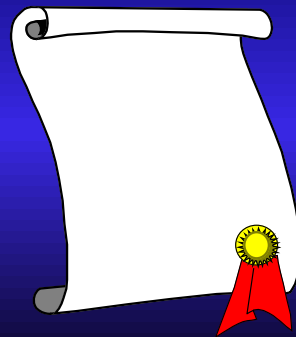
- 学号、姓名均可作为候选键
- 学号、姓名为主属性
- 性别、年龄、系名为非主属性

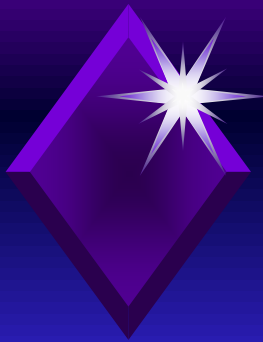


# 本讲主要内容

---

- 一. 规范化的概念
- 二. 不好的数据库设计中的异常
- 三. 函数依赖
- 四. 部分函数依赖
- 五. 传递函数依赖
- 六. 候选键的形式定义
- 七. 范式：2NF、3NF和BCNF





## 七、范式：2NF、3NF和BCNF

(参见教材P183-192)

### 1、范式(Normal Forms)

- **范式的定义：** 关系数据库中符合某一级别的关系模式的集合。所谓“第几范式”，是表示关系的某一种级别，R为第几范式就可以写成

$$R \in xNF$$

- **各范式之间的联系有**

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

- **并不总是需要达到最高范式**

## 七、范式：2NF、3NF和BCNF

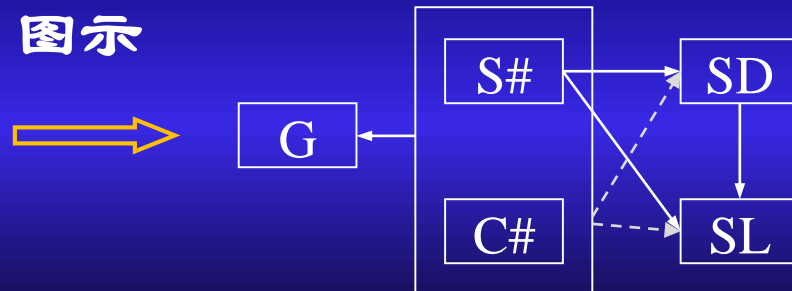
### 2、实例

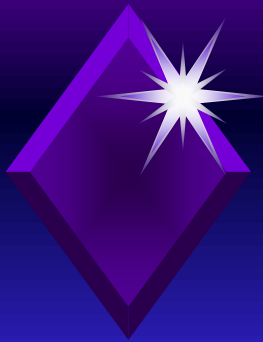
例如，有一个关系模式S-L-C (S#, SD, SL, C#, G)，其中S#为学生的学号，SD为学生所在系，SL为学生的住处，并且每个系的学生住在同一个地方，C#为课程号，G为成绩。这里键为 (S#, C#)。

函数依赖有：

$S\# \rightarrow SD,$	$(S\#, C\#) \xrightarrow{F} G$
$S\# \rightarrow SL,$	$(S\#, C\#) \xrightarrow{P} SD$
$SD \rightarrow SL$	$(S\#, C\#) \xrightarrow{P} SL$
	$S\# \xrightarrow{t} SL$

图示





## 七、范式：2NF、3NF和BCNF

### 3、第一范式 (First Normal Form, 简称为1NF)

—— 如果一张表不含有多值属性 (有时称为重复字段) 和内部结构 (比如记录类型) 的列, 则称该表为第1范式

- 关系模式  $S-L-C(\underline{S\#}, SD, SL, \underline{C\#}, G) \in 1NF$

- 关系模式  $S-L-C$  存在更新异常

- 一个学生修很多课程

冗余太大

- 一个学生转系

修改复杂

- 插入一个尚未选课的学生信息

插入异常

- 一个学生只选修了一门课,

删除异常

但现在决定不选了

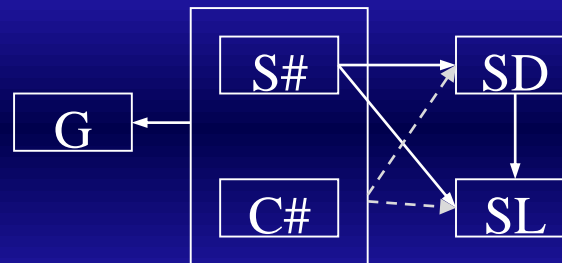


## 七、范式：2NF、3NF和BCNF

### 4、第2范式 (Second Normal Form, 简称2NF)

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

S-L-C:



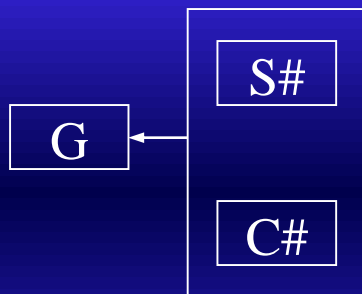
- 分析：存在非主属性对键的部分函数依赖
- 结果： $S-L-C \in 1NF$ ，但  $S-L-C \notin 2NF$



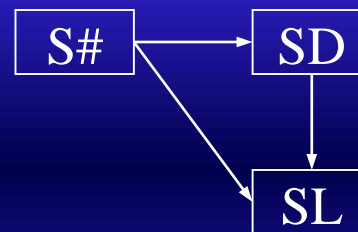
## 七、范式：2NF、3NF和BCNF

- 分解：将S-L-C分解为：S-L(S#, SD, SL)，和S-C(S#, C#, G)

S-C图示

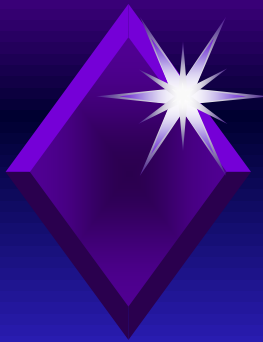


S-L图示



- 分析：S-C的键为 (S#, C#)，S-L的键为S#，不存在非主属性对键的部分函数依赖。
- 结果：S-C  $\in$  2NF；S-L  $\in$  2NF
- 问题：

S-C和S-L中消除了1NF中的某些更新异常，但仍然存在更新异常



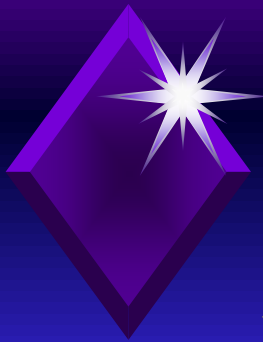
## 七、范式：2NF、3NF和BCNF

在S-L (S#, SD, SL) 和S-C (S#, C#, G) 中

- ☆ 一个学生修很多课程 ✓
- ☆ 一个学生转系 ✓
- ☆ 插入一个尚未选课的学生信息 ✓
- ☆ 一个学生只选修了一门课，但现在决定不选了 ✓

- ☆ 一个系有很多学生，且同系学生住在一个地方 ✗
- ☆ 一个学生转系 ✗
- ☆ 一个新系创建但新生尚未注册 ✗
- ☆ 一个系所有学生毕业了 ✗

结论：2NF可以消除一些1NF中存在的更新异常，但不能彻底消除更新异常

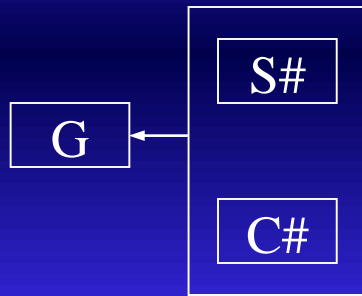


## 七、范式：2NF、3NF和BCNF

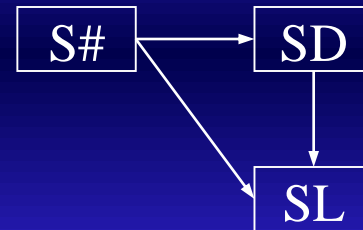
### 5、第3范式 (Third Normal Form, 简称3NF)

—— 若 $R \in 2NF$ ，且每一个非主属性不传递函数依赖于键，则 $R \in 3NF$ 。

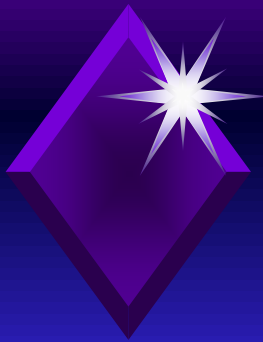
S-C图示



S-L图示



- **分析：** S-C的键为 (S#, C#)，不存在非主属性对键的传递函数依赖。S-L的键为S#，存在非主属性对键的传递函数依赖  $S\# \rightarrow SL$ 。
- **结果：**  $S-C \in 3NF$  ;  $S-L \notin 3NF$



## 七、范式：2NF、3NF和BCNF

- 投影分解法：将S-L分解为：S-D (S#, SD) 和 D-L (SD, SL)

S-D图示

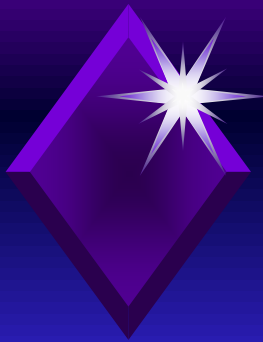


D-L图示



- 分析：S-D的键为S#，D-L的键为SD，不存在非主属性对键的传递函数依赖。
- 结果：S-D  $\in$  3NF ; D-L  $\in$  3NF
- 问题：

S-D和D-L中消除了前面的更新异常了吗？



## 七、范式：2NF、3NF和BCNF

在S-D (S#, SD) 和 D-L (SD, SL) 中：

- ☆ 一个系有很多学生，且同系学生住在一个地方 ✓
- ☆ 一个学生转系 ✓
- ☆ 一个新系创建但新生尚未注册 ✓
- ☆ 一个系所有学生毕业了 ✓

3NF还存在更新异常吗？



## 七、范式：2NF、3NF和BCNF

例如 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。

由语义可以得到函数依赖

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

键为  $(S, J)$  ,  $(S, T)$

分析：没有非主属性，不存在非主属性对键的部分与传递依赖，因此， $STJ \in 3NF$



## 七、范式：2NF、3NF和BCNF

---

在STJ (S, T, J) 中：

- ☆ 多个学生选同一门课 ×
- ☆ 某门课程改名 ×
- ☆ 一个教师开设某门课，但还没有学生选 ×
- ☆ 选修某门课的学生全部毕业 ×



3NF不能  
彻底消  
除更新  
异常

结论：3NF可以消除一些2NF中存在的更新异常，但不能彻底消除更新异常





## 七、范式：2NF、3NF和BCNF

---

### 6、Boyce-Codd范式

(Boyce-Codd Normal Form, 简称BCNF)

——  $R \in 3NF$ , 且每一个决定因素都包含键, 则  $R \in BCNF$ 。

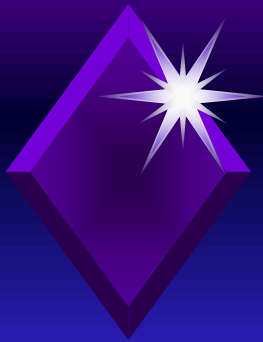
在3NF关系  $S-D (\underline{S\#}, SD)$  和  $D-L (\underline{SD}, SL)$  中:

$S\# \rightarrow SD, \quad SD \rightarrow SL$



$S-D \in BCNF, \quad D-L \in BCNF$

所有的3NF都是BCNF?



## 七、范式：2NF、3NF和BCNF

---

在  $STJ (\underline{S}, T, \underline{J})$  中,

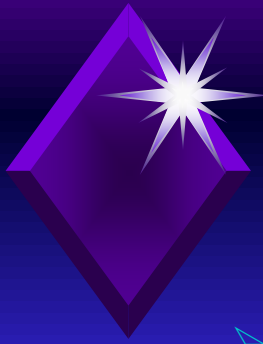
$T \rightarrow J$  ;  $(S, J) \rightarrow T$  ;  $(S, T) \rightarrow J$

键为  $(S, J)$  ,  $(S, T)$

分析:

- 没有非主属性, 所以, 不存在非主属性对键的部分与传递依赖, 因此,  $STJ \in 3NF$
- 在  $T \rightarrow J$  中, 决定因素  $T$  不包含键, 因此,  $STJ \notin BCNF$

结论: 并非所有的 3NF 都是 BCNF



## 七、范式：2NF、3NF和BCNF

---

- 分解：将STJ (S, T, J) 分解为 S-J (S, J) 和 T-J (T, J)

结果：S-J (S, J) 和 T-J (T, J) 都是BCNF

S-J和T-J中消除了前面的更新异常了吗？

∞

## 七、范式：2NF、3NF和BCNF

如果一个模式属于BCNF，在函数依赖的范畴内，彻底消除了更新异常吗？

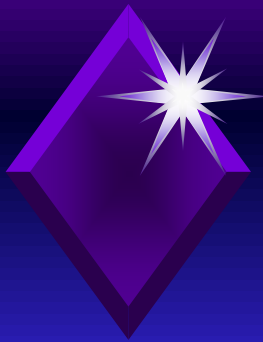
Yes

如果一个模式属于BCNF，在数据依赖的范畴内，彻底消除了更新异常吗？

No

其它的数据依赖也会产生更新异常

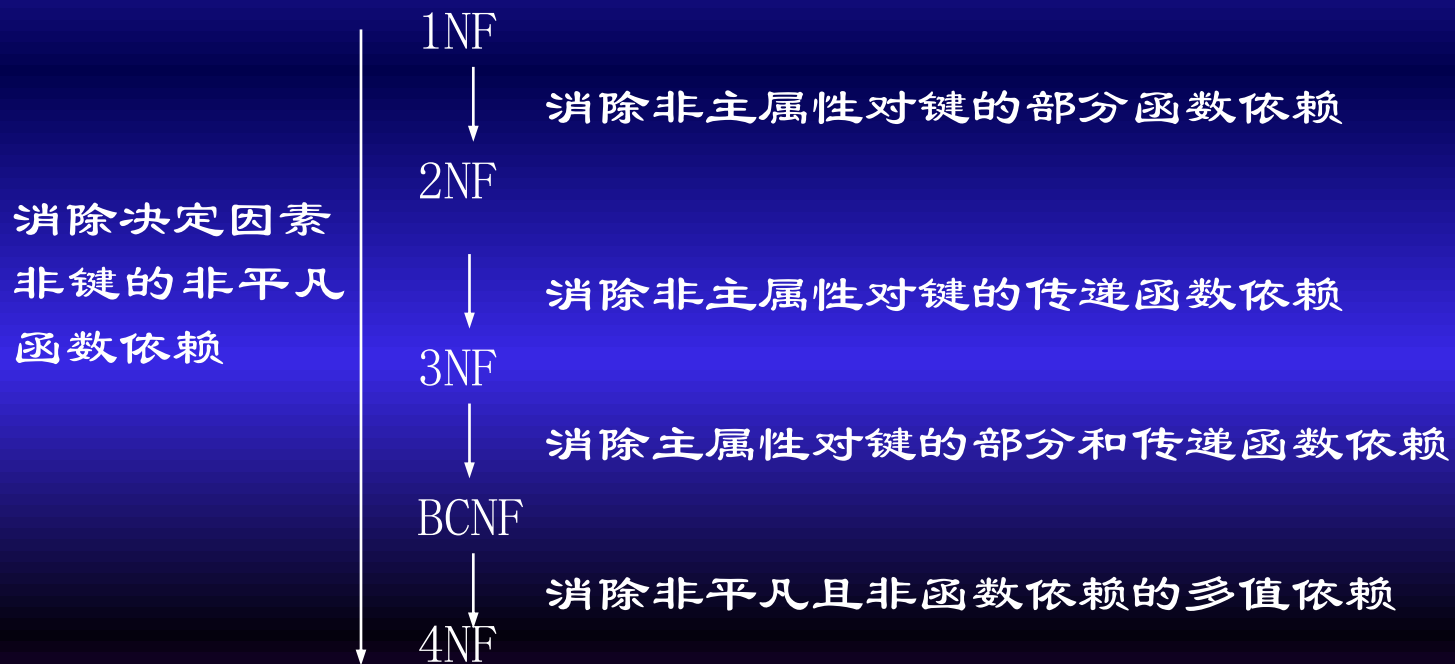




## 七、范式：2NF、3NF和BCNF

### 规范化过程

规范化过程是通过对关系模式的分解来实现的。

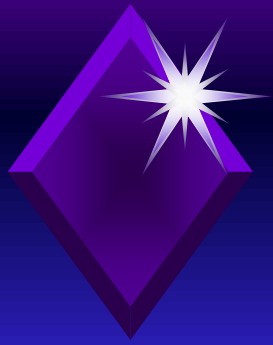




## 其它设计考虑

---

- 使用规范化方法设计数据库的逻辑结构时，**并不是要使得到的数据库模式都达到最高范式，而是还需要平衡查询效率和更新代价。**
- **反规范化设计**
  - 对一些特定的应用不规范化，而是通过使用冗余来改进性能
  - 反规范化设计是为了提高查询效率
  - 进行反规范化设计后，需要采取措施，处理可能出现的更新异常



*Questions?*





# 本讲主要目标



学完本讲后，你应该能够了解：

- 1、**规范化**是关系数据库逻辑设计的一种方法；
- 2、**一个不好的数据库设计存在更新异常**：插入异常、删除异常、数据冗余和修改复杂；而导致更新异常的原因是：在一个关系模式中存在某些函数依赖。
- 3、**规范化方法**就是将一个关系模式分解成多个模式，使得这些函数依赖不出现在同一个关系模式中的过程；
- 4、函数依赖、部分函数依赖、传递函数依赖的定义；
- 5、1NF、2NF、3NF、BCNF的定义，以及范式之间的包含关系；
- 6、对于一个关系模式，**判断**其最高属于第几**范式**，并使用基本的模式分解方法，将1NF分解为2NF，将2NF分解为3NF；
- 7、使用规范化方法设计数据库的逻辑结构时，并不是要使得到的数据库模式都达到最高范式，而是还需要平衡查询效率和更新代价。



# 问题讨论

1、实际上，进行规范化的目的是为了**避免更新异常和提高更新代价**；而有时规范化产生的高范式会降低有些查询的代价。这种说法对吗？你在设计数据库模式时，会采取什么方法？



# 问题讨论

2、在关系模式R (A, B, C, D) 中, 存在函数依赖关系  $\{(A, B) \rightarrow C, B \rightarrow D\}$ , 则其候选键是什么? 最高达到第几范式?

3、在关系模式R (A, B, C, D) 中, 存在函数依赖关系  $\{A \rightarrow B, A \rightarrow C, A \rightarrow D, (B, C) \rightarrow A\}$ , 则其候选键是什么? 最高达到第几范式?

4、在关系模式R (D, E, G) 中, 存在函数依赖关系  $\{E \rightarrow D, (D, G) \rightarrow E\}$ , 则其候选键是什么? 最高达到第几范式





# 练习

---

教材：《数据库系统原理教程》（第2版）

P196-197

- 1) 2
- 2) 3\*
- 3) 5
- 4) 6

