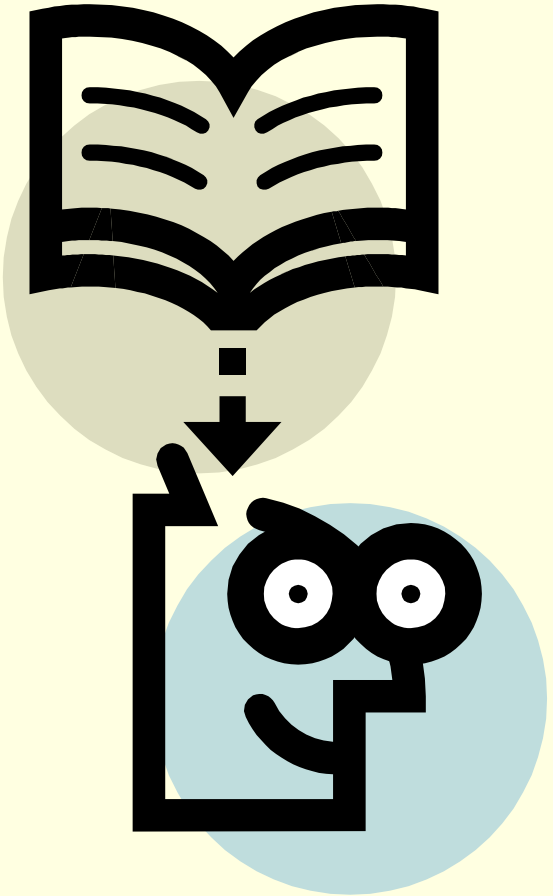
第10章 数据依赖与关系模式的规范化

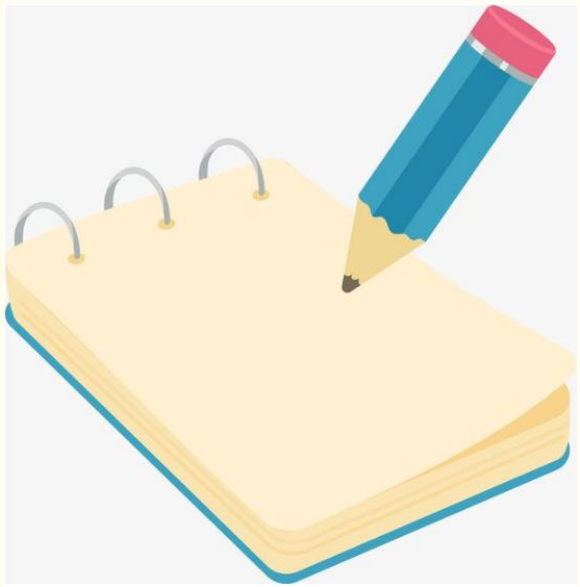
Chapter 10 Data Dependencies & Schema Normalization

目录 Contents



10.1 关系模式规范化概述

- 10.1.1 关系模式的设计
- 10.1.2 “不好的”（bad）关系模式
- 10.1.3 如何设计“好的”（good）关系模式
- 10.1.4 权衡：规范化 vs. 性能



10.1.1 关系模式的设计

- 关系模式的设计：成功开发DB应用的关键
  - DB用于支持数据密集型应用 （data intensive apps）
  - 数据密集型应用的核心问题是： DB设计
  - DB设计——结构方面：
    - 采用数据模型（data model）, e.g. relational model
      - 面向过程的方法（process-oriented）, e.g. SA/SD
      - 面向数据的方法（data-oriented）, e.g. IEM √
    - 设计数据模式（data schemas）, e.g. a collection of relation(al) schemas
  - DB设计的目标：设计“好的”（good）关系模式——属于数据库的逻辑设计
    - 理论依据：函数依赖（functional dependencies）与规范化（normalization）理论

10.1.2 “不好的”关系模式

- 模式设计的不同方案
  - 同一个关系数据库可以有多种不同的模式设计方案
  - 假设一个学生数据库中有以下属性：
    学号（SNO），课程号（CNO），成绩（G），
    主讲教师（T），教师所在系（DEPT）。
  - 上述属性具有下列数据语义：
    - 学号（SNO）是学生的唯一标识，课程号（CNO）是课程的唯一标识；
    - 每位学生所修的每门课程都有一个成绩（G）；
    - 每门课程只有一位主讲教师（T），但一位教师可以主讲多门课程；
    - 教师中没有重名，每位教师只属于一个系（DEPT）。
  - 有多种模式设计方案

10.1.2 “不好的”关系模式

- 方案一
  - 一个关系
    - R (SNO, CNO, G, T, DEPT)
- 方案二
  - 三个关系
    - R1 (SNO, CNO, G)
    - R2 (CNO, T)
    - R3 (T, DEPT)

10.1.2 “不好的”关系模式

- 方案一
  - 关系模式 R(SNO, CNO, G, T, DEPT) 的一个实例：

R

SNO	CNO	G	T	DEPT
95601	C01	85	张乐	计算机
95602	C01	90	张乐	计算机
95801	C01	95	张乐	计算机
95801	M03	91	王丽	数理
95802	M03	88	王丽	数理

10.1.2 “不好的”关系模式

- 方案二
  - 关系模式 R1 (SNO, CNO, G)，R2 (CNO, T)，R3 (T, DEPT) 的一个实例：

R1

SNO	CNO	G
95601	C01	85
95602	C01	90
95801	C01	95
95801	M03	91
95802	M03	88

R2

CNO	T
C01	张乐
M03	王丽

R3

T	DEPT
张乐	计算机
王丽	数理

10.1.2 “不好的”关系模式

- 2. 不同模式设计方案的比较
  - 不同的模式设计方案对数据库的影响是否相同？——否！
  - 我们从下面几个方面来比较前面两种数据库设计方案：
    - 数据冗余度
    - 数据更新操作：元组插入；元组删除；元组修改

10.1.2 “不好的”关系模式

- 经过比较发现，方案一有如下问题：
  - 数据冗余（data redundancy）
    - 重复数据：
      - 例如，“C01”课的主讲教师是“张乐”；“张乐”是“计算机”系的教师。
      - “M03”课、“王丽”等情况类似
    - 方案二不存在数据冗余

R

SNO	CNO	G	T	DEPT
95601	C01	85	张乐	计算机
95602	C01	90	张乐	计算机
95801	C01	95	张乐	计算机
95801	M03	91	王丽	数理
95802	M03	88	王丽	数理

方案一

R1

SNO	CNO	G
95601	C01	85
95602	C01	90
95801	C01	95
95801	M03	91
95802	M03	88

R2

CNO	T
C01	张乐
M03	王丽

R3

T	DEPT
张乐	计算机
王丽	数理

方案二

10.1.2 “不好的”关系模式

- 经过比较发现，方案一有如下问题：
  - 更新异常（update anomalies）
    - A. 修改异常（modification anomalies）
      - “张乐”调到“土木”系，如果只改了其中一个元组的值，那么会出现数据不一致问题。
      - “M03”课的主讲教师换成“杨萍”，如果只改了其中一个元组的值，那么会出现数据不一致问题。
      - 对于方案二，只需分别修改R2和R3关系中的元组的值即可，不会出现数据不一致。

R

SNO	CNO	G	T	DEPT
95601	C01	85	张乐	计算机
95602	C01	90	张乐	计算机
95801	C01	95	张乐	计算机
95801	M03	91	王丽	数理
95802	M03	88	王丽	数理

方案一

R1

SNO	CNO	G
95601	C01	85
95602	C01	90
95801	C01	95
95801	M03	91
95802	M03	88

R2

CNO	T
C01	张乐
M03	王丽

R3

T	DEPT
张乐	计算机
王丽	数理

方案二

10.1.2 “不好的”关系模式

- 经过比较发现，方案一有如下问题：
  - 更新异常（update anomalies）
    - B. 删除异常（deletion anomalies）
      - “C01”课不开了，需删除R中所有相关元组（前3个），但“张乐”是“计算机”系教师的信息也同时被删。
      - 对于方案二，可仅在R1和R2中分别删除课程“C01”的相关元组，但不会同时删除“张乐”是“计算机”系教师的信息，其相应元组仍保留在R3中。

R

SNO	CNO	G	T	DEPT
95601	C01	85	张乐	计算机
95602	C01	90	张乐	计算机
95801	C01	95	张乐	计算机
95801	M03	91	王丽	数理
95802	M03	88	王丽	数理

方案一

R1

SNO	CNO	G
95601	C01	85
95602	C01	90
95801	C01	95
95801	M03	91
95802	M03	88

R2

CNO	T
C01	张乐
M03	王丽

R3

T	DEPT
张乐	计算机
王丽	数理

方案二

10.1.2 “不好的”关系模式

- 经过比较发现，方案一有如下问题：
  - 更新异常（update anomalies）
    - C. 插入异常（insertion anomalies）
      - 若需要新开设一门尚未有学生选修的课程(C02, 李四)，则无法构造出一个由SNO, CNO, G, T, DEPT属性值所组成的新元组，将其插入到R中。
      - 对于方案二，可以直接将元组(C02, 李四)插入到R2中。【若R3中尚没有元组(李四,计算机)，可以添加之。】

R

SNO	CNO	G	T	DEPT
95601	C01	85	张乐	计算机
95602	C01	90	张乐	计算机
95801	C01	95	张乐	计算机
95801	M03	91	王丽	数理
95802	M03	88	王丽	数理

方案一

R1

SNO	CNO	G
95601	C01	85
95602	C01	90
95801	C01	95
95801	M03	91
95802	M03	88

R2

CNO	T
C01	张乐
M03	王丽

R3

T	DEPT
张乐	计算机
王丽	数理

方案二

10.1.2 “不好的”关系模式

- 上述例子表明，不同的模式设计方案确有好坏的区分。
  好的设计方案应该是：数据冗余度低，且能避免更新异常。

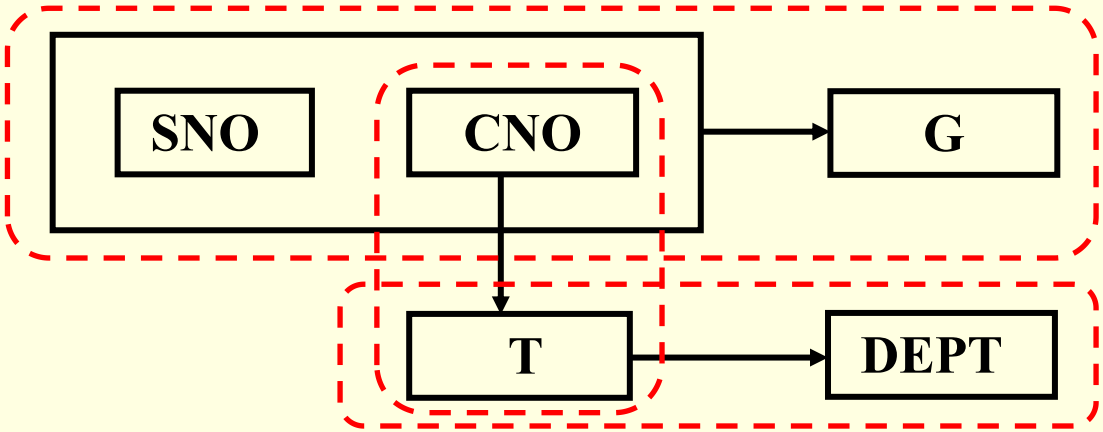
- 设计出“不好的”关系模式的原因：
  关系模式中数据的“语义”不单纯。在此，“语义”专指问题空间中固有的、相对稳定的数据依赖（data dependency, DD）关系。
- 数据依赖（DD）：是现实世界中属性之间相互联系的一种抽象，是语义的一种体现。主要包括：
  - 函数依赖（functional dependency, FD）
    - 一个/组属性X的值是否决定另一个/组属性Y的值。
  - 多值函数依赖（multi-valued dependency, MVD）【不学】
  - 连接依赖（join dependency, JD）【不学】

10.1.2 “不好的”关系模式

- 由于上例模式R中存在不好的数据依赖（FD），引起数据冗余和更新异常。【解决方法】通过分解关系模式R来消除其中不合适的数据依赖，这样的模式分解过程称为关系模式规范化。
- 上例模式R(SNO, CNO, G, T, DEPT)中存在以下三个函数依赖：
  - 1. SNO,CNO → G　　2. CNO → T　　3. T → DEPT

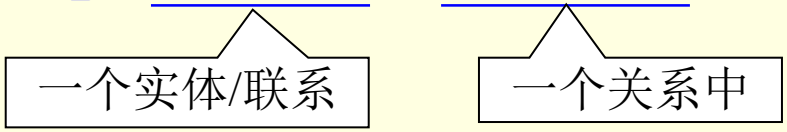
相应地表示了三个事实（fact），为何不用三个模式来描述呢？

R1(SNO,CNO,G)　2. R2(CNO,T)　3. R3(T,DEPT)【方案二】



10.1.3 如何设计“好的”关系模式

- 设计“好的”关系模式：模式分解，规范化 & 范式
  - 规范化（normalization）
    - 将一个关系模式按“语义单纯化”的原则进行合理分解——称模式分解（decomposition of schema），以最终达到“一事一地（One Fact in One Place）”。



  - 模式分解的条件 / 准则
    - 起码的条件：无损分解（lossless decomposition）：分解前、后要等价，即对分解前关系、分解后诸关系的任何相同的查询总能产生相同的结果。（可通过“连接”分解后的诸关系来重构分解前关系）。
    - 理想的分解：保持依赖的分解（dependency-preserving decomposition）：这需进一步论述。

10.1.3 如何设计“好的”关系模式

- 范式（normal forms）
  - 规范化（即模式分解）程度的一种测度。根据对属性间所存在的内在语义联系要求的不同，又可将关系模式的规范化分为若干个级别，称为范式。

低 ↓ 高

第一范式 (1NF)	
第二范式 (2NF)	函数依赖 (FD) 范畴 √
第三范式 (3NF)	
Boyce-Codd 范式 (BCNF)	
第四范式 (4NF)	多值函数依赖 (MVD) 范畴 ×
第五范式 (5NF)	连接依赖 (JD) 范畴 ×

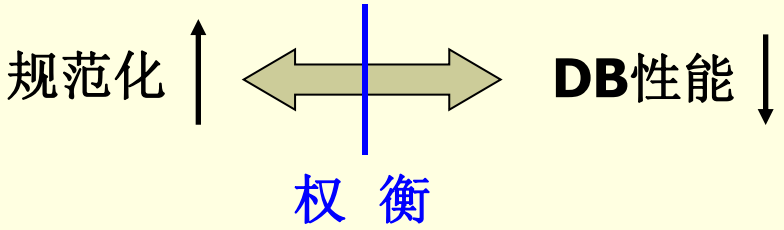
  - 一个关系模式R达到x范式的程度，英文称：R is in xNF，记为：$R \in xNF$；否则，称：R violates xNF condition, 记为：$R \notin xNF$。

10.1.3 如何设计“好的”关系模式

- 数据依赖理论的起源
  - 函数依赖（functional dependency, FD）
    - 1970年，E. F. Codd提出
    - 1972~1974年，Codd, Casey, Bernstein, Armstrong
      - 完全/部分FD，平凡/非平凡FD，直接/传递FD
      - 键（key）
    - 1974年，Armstrong公理系统
      - FD的逻辑蕴涵
      - FD的形式化推理规则集
  - 多值依赖（Multi-Valued Dependency, MVD）
    - 1976~1978年，Zaniolo, Fagin, Delobel

10.1.4 权衡：规范化 vs. 性能

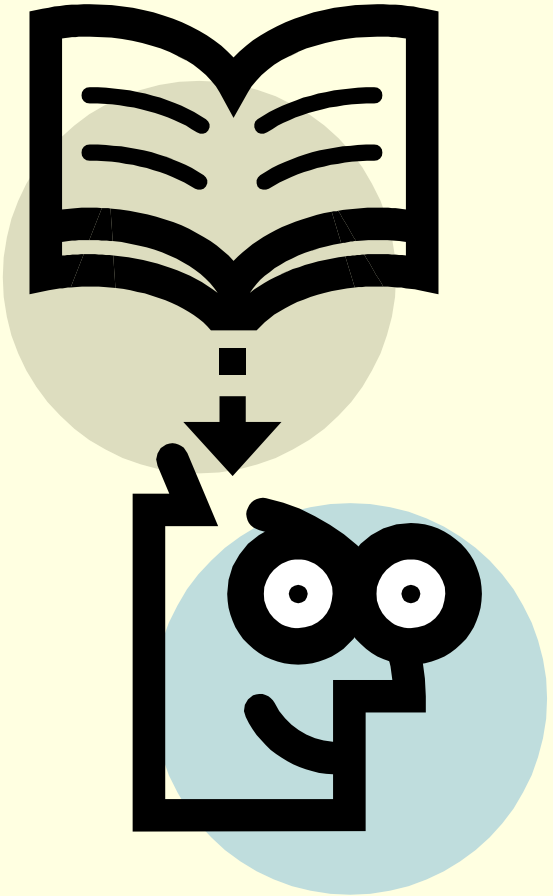
- 规范化程度并非越高越好



- 程度
  - 一般规范化到BCNF或3NF已足够了！
- 策略
  - 对更新频繁/查询较少的表：规范化↑ ------ 减少“异常”
  - 对查询频繁/更新较少的表：规范化↓ ------ 提高“性能”

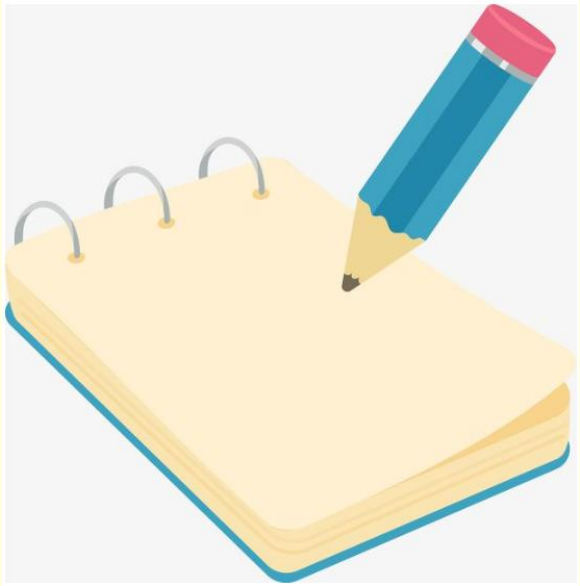
目录 Contents

- 10.1 关系模式规范化概述
- 10.2 函数依赖与范式
- 10.3 模式分解理论（简介）



10.2 函数依赖与范式

- 10.2.1 函数依赖（FD）
  - 完全/部分FD，平凡/非平凡FD，直接/传递FD
- 10.2.2 范式（函数依赖范畴内）
  - 1NF，2NF，3NF，BCNF



10.2.1 函数依赖

- 定义：在一个关系模式R(U)中，如果一部分属性Y⊂U的取值依赖于另一部分属性X⊂U的取值，则在属性集X和属性集Y之间存在着一种数据依赖关系，称之为函数依赖。

  - 例1：在学生关系模式 Student(SNO, Sname, Sage, Sdept) 中就存在下面的几个函数依赖关系：
    - {Sname} 的取值依赖于 {SNO} 的取值
    - {Sname, Sage} 的取值依赖于 {SNO} 的取值
    - {Sname, Sage, Sdept } 的取值依赖于 {SNO} 的取值

  - 例2：在选课关系模式SC(SNO, CNO, Grade)中：
    - {Grade} 的取值依赖于 {SNO, CNO} 的取值

10.2.1 函数依赖

- 形式定义：函数依赖 / 决定子（determinant）
  - 设有关系模式R(U)，U是其属性集，A、B是U的子集。若对于模式R(U)的任一关系实例 r 中的任意两个元组 t 和s，

    t [A] = s [A] $\Longrightarrow$ t [B] = s [B] 成立，

    则称B函数依赖于A或A函数决定B，记为：A → B。A称为决定子。亦可记为：A1, A2, … , An → B1, B2, … , Bm
    (Ai，Bj为单个属性)。
  - 注：A不函数决定B，记为A $\nrightarrow$ B。
    若A → B，B → A，则称两者一一对应，记为A ↔ B。
- 分裂/合并规则（the splitting/combining rule）
  - **X1, X2, … , Xn → Y1, Y2, … , Ym $\Longleftrightarrow$ X1, X2, … , Xn → Y1**

    …

    X1, X2, … , Xn → Ym

10.2.1 函数依赖

- 函数依赖反映的是同一个关系中的两个属性子集之间在取值上的依存关系，这种依存关系实际上也是一种数据完整性约束。

- 例：假设一个学生数据库中有以下属性：学号（SNO），课程号（CNO），成绩（G），主讲教师（T），教师所在系（DEPT）。这些数据具有下列语义：
  - 学号（SNO）是学生的唯一标识，课程号（CNO）是课程的唯一标识；
  - 一位学生所修的每门课程都有一个成绩（G）；
  - 每门课程只有一位主讲教师（T），但一位教师可以主讲多门课程；
  - 教师中没有重名，每位教师只属于一个系（DEPT）。
  - 有以下三个函数依赖：

    1. SNO,CNO → G　　2. CNO → T　　3. T → DEPT

10.2.1 函数依赖

- **定义：平凡依赖 / 非平凡依赖 / 完全非平凡依赖**

  设A、B是某个关系模式的两个属性子集，对函数依赖A → B，

  - 若B ⊆ A，则称此函数依赖为平凡依赖（trivial dependency）；
  - 若B − A ≠ Φ，则称此函数依赖为非平凡依赖（nontrivial dependency）；
  - 若B ∩ A = Φ，则称此函数依赖为完全非平凡依赖（completely nontrivial dependency）。
  - 例：在关系SC (Sno, Cno, Grade)中，

    非平凡函数依赖： (Sno, Cno) → Grade （也是完全非平凡依赖）

    非平凡函数依赖： (Sno, Cno) → (Cno, Grade)

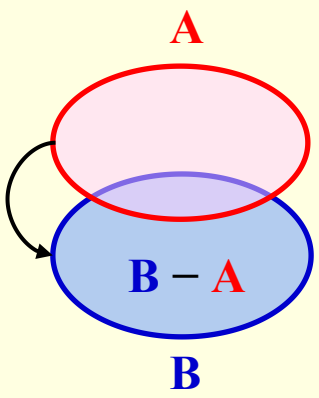
    平凡函数依赖： (Sno, Cno) → Sno

    (Sno, Cno) → Cno

    Sno → Sno
  - 任一关系模式中，平凡函数依赖都是必然存在的，它不反映有意义的数据语义，因此，若不特别声明，我们总是讨论非平凡函数依赖。

10.2.1 函数依赖

- 平凡依赖规则（the trivial-dependency rule）
  - $A \rightarrow B \iff A \rightarrow B - A$



- 定义: 完全依赖 / 部分依赖
  - 设A、B是某个关系模式的两个不同属性集，

    若有函数依赖$A \rightarrow B$，且不存在 $C \subset A$，使$C \rightarrow B$，则称依赖$A \rightarrow B$为完全依赖（full dependency），记为 $A \xrightarrow{f} B$；

    否则称为部分依赖（partial dependency），记为 $A \xrightarrow{p} B$。

- 例: 在关系SC (Sno, Cno, Grade)中，由于：Sno $\nrightarrow$ Grade，Cno $\nrightarrow$ Grade，因此：(Sno, Cno) $\xrightarrow{f}$ Grade

10.2.1 函数依赖

- 定义：传递依赖（transitive dependency）
  - A, B, C是某关系模式的三个不同属性集，若有：$A \rightarrow B$, $B \not\rightarrow A$, $B \rightarrow C$, 则称C传递函数依赖于A, 记为：$A \xrightarrow{t} C$。
  - 为了使得函数依赖在表示形式上的简单化，传递函数依赖与非传递函数依赖在表示形式上没有区别。

- 传递规则（the transitive rule）
  - $A \rightarrow B,\ B \rightarrow C \Longrightarrow A \rightarrow C$

- 例: 在关系Std (Sno, Sdept, Dean_name)中，有：

  Sno → Sdept，Sdept → Dean_name，所以, Dean_name传递函数依赖于Sno，即Sno → Dean_name

10.2.2 范式

- 回顾概念：
  - 键（key）：在关系模式R(U)中，若存在一个最小的属性（组）$K \subseteq U$ 满足$K \xrightarrow{f} U$, 则称K为R的侯选键（candidate key）或键：
    - 决定性条件：此属性（组）K的值唯一地决定了其他属性的值（因而也决定了整个元组）；
    - 最小性条件：此属性（组）K的任何真子集均不满足决定性条件。
  - 主键（primary key, PK）：在关系模式机器实现时，若关系模式有多个候选健，则选定其中的一个做为主键。其他键称为候补键（alternate key）。
  - 超键（superkey）：关系模式中包含键的属性（组）称为超键。
  - 全键（all key）：键由关系模式所有属性构成的，称为全键。
  - 主属性：关系模式的所有键中的属性；否则称为非主属性。

10.2.2 范式

- 范式（normal form）：

  是符合某种规范化程度的全体关系模式的一个集合。包括：

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

函数依赖范畴内

- 各种范式之间的关系：

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

10.2.2 范式

- 定义：1NF
  - 设有一个关系模式R, 若R的任一关系实例r中的属性值均是原子数据（即：属性都是不可再分的数据项）， 则称R属于第一范式（1NF），记为R∈1NF。
  - 注：
    - 1NF条件是传统关系数据库的基本要求，目前大多数关系数据库（RDBMS）均有此要求。突破此要求称为非第一范式（non-first normal form, NF^2）。
    - E-R图中的非原子属性在转化为关系时要这样处理：
      - 对集合属性：在表中“纵向”展开【如：选修课程】
      - 对元组属性：在表中“横向”展开【如：邮寄地址】
    - 满足1NF的关系模式并不一定是一个好的关系模式。

10.2.2 范式

- 定义：2NF
  - 设有一个关系模式R∈1NF, 若R的每个非主属性均完全函数依赖于键, 则称R属于第二范式（2NF），记为R∈2NF。
  - 注：
    - $R \in 2NF \Longrightarrow R \in 1NF$
    - 考察 R (**SNO, CNO**, G, T, DEPT) ∈ 1NF，

      因为 SNO, CNO → G,

      CNO → T, T → DEPT $\Longrightarrow$ CNO → DEPT ,

      于是 CNO → T, DEPT

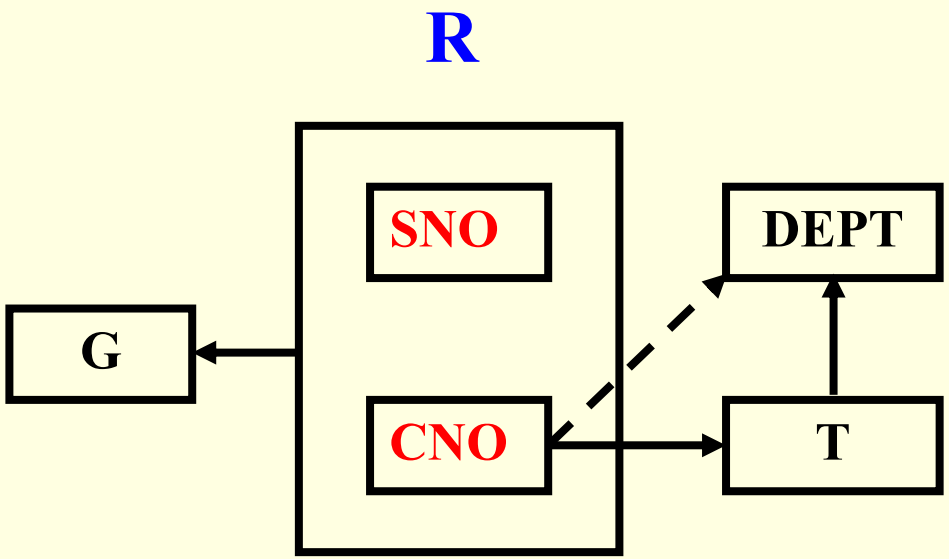
      故 Key = {SNO, CNO}。

      但 CNO → T, DEPT，即：Key $\xrightarrow{P}$ T, DEPT,

      故 R∉2NF 。

10.2.2 范式

- R (SNO, CNO, G, T, DEPT) 不是一个好的关系模式
  - 数据冗余
  - 更新异常
- 原因
  - T, DEPT部分函数依赖于键{SNO, CNO}。



10.2.2 范式

- 解决方法：
  - 模式分解：消除部分函数依赖。

    R1(**SNO, CNO**, G)∈2NF 和 R2(**CNO**, T, DEPT)∈2NF。
    - 但R2仍有问题：

      c01 t1 d1

      c02 t1 d1　冗余 & 异常

      c03 t1 d1

      c04 t2 d2
    - 原因：R2的Key={CNO}，
      存在T → DEPT , 而T不是超键，DEPT又不是主属性。

- 上述例子说明：采用模式分解法将一个1NF的关系模式分解为多个2NF的关系模式，可以在一定程度上减轻原关系模式中存在的数据冗余和更新异常问题，但并不能完全消除数据冗余和更新异常。

10.2.2 范式

- 定义：3NF
  - 设有一个关系模式 $R \in 1NF$，若R的任一非平凡函数依赖 $X \rightarrow A$ 满足下列两个条件之一：(1) X是超键, (2) A是主属性，则称R属于第三范式（3NF），记为$R \in 3NF$。
  - 注：
    - 若$R \notin 3NF$，意味着：X非超键，并且A非主属性：
      - 若X是键的真子集：非主属性A部分依赖于键：$Key \xrightarrow{p} A$；
      - 若X既非键的真子集，又非超键：

        $Key \rightarrow X,\ X \rightarrow A \Longrightarrow Key \xrightarrow{t} A,$

        即：非主属性A传递依赖于键。

      上述表明，3NF消除了非主属性对键的部分依赖和传递依赖。
    - $R \in 3NF \Longrightarrow R \in 2NF$。

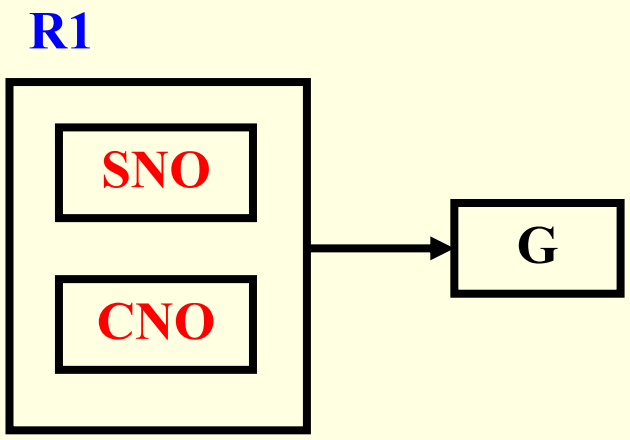
10.2.2 范式

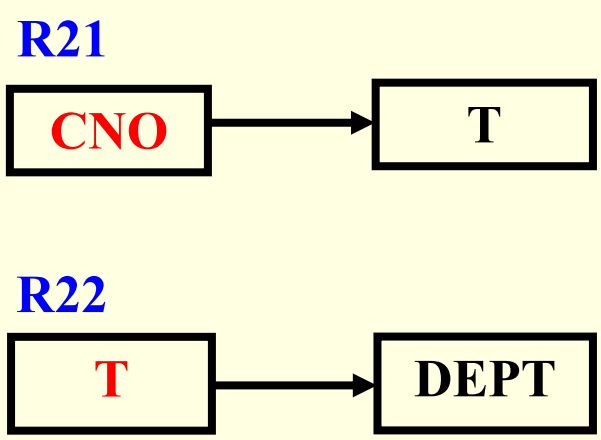
- 例：将关系模式R(SNO, CNO, G, T, DEPT)分解得到

  R1(**SNO, CNO**, G) ∈2NF 和 R2(**CNO**, T, DEPT)∈2NF，

  由于关系模式R2中存在非主属性DEPT对键的传递依赖，所以通过模式分解，把R2分解为两个关系模式，以消除传递函数依赖（ CNO → T，T → DEPT ）：

  R21(**CNO**, T) ∈3NF，　R22(**T**, DEPT) ∈3NF





10.2.2 范式

- 若R∈3NF，则R的每一个非主属性既不部分函数依赖于（候选）键，也不传递函数依赖于键。
- 将一个2NF的关系模式分解为多个3NF的关系模式，可以在一定程度上消除原关系模式中存在的数据冗余和更新异常问题，但并不能完全消除数据冗余和更新异常。

10.2.2 范式

- 定义：BCNF
  - 设有一个关系模式R∈1NF，若R的任一非平凡函数依赖 X → A均满足下列条件：决定子X必是超键，则称R属于Boyce–Codd范式（BCNF），记为R∈BCNF。
  - 注：
    - **A.** R∈BCNF ⟹ R∈3NF。
    - **B.** BCNF消除了(非主/主)属性对键的部分依赖和传递依赖。
    - **C.** 关系模式达到BCNF后，在函数依赖范畴内已彻底规范化了，并消除了数据冗余和更新异常。
    - **D. 任何全键关系模式必属于BCNF**
    - **E. 任何两属性关系模式必属于BCNF**
    - **F. 关系模式无损分解成BCNF的方法？**

10.2.2 范式

- D. 任何全键关系模式必属于BCNF
  - 证明：设$R(A_1, A_2, \dots, A_n)$是全键关系模式, 即：
    $Key=\{ A_1, A_2, \dots, A_n \}$,

    反证法：

    假设$R \notin$BCNF，则根据BCNF的定义，必存在一个非平凡函数依赖$X \rightarrow A_i$, 其决定子X不是超键。

    注意, $X \subset \{ A_1, A_2, \dots, A_n \}$，且$A_i \notin X$；同时,

    $X \subset \{ A_1, A_2, \dots, A_n \} - \{A_i\} = \{ A_1, A_2, \dots, A_{i-1}, A_{i+1}, \dots, A_n \}$,

    因$X \rightarrow A_i$，故$Key = \{ A_1, A_2, \dots, A_{i-1}, A_{i+1}, \dots, A_n \}$。

    这与全键的条件矛盾！命题得证。

10.2.2 范式

- E. 任何两属性关系模式必属于BCNF
  - 证明：设R(A1, A2)是两属性关系模式，则有4种非平凡函数依赖的情形：（穷举法）
    - (1) 不存在任何非平凡的函数依赖：
      无冒犯BCNF 条件的函数依赖或R是全键，故R∈BCNF。
    - (2) A1 → A2，但A2 ↛ A1：
      Key=A1，唯一的决定子A1是超键，故R∈BCNF。
    - (3) A2 → A1，但A1 ↛ A2：
      Key=A2，唯一的决定子A2是超键，故R∈BCNF。
    - (4) A1 → A2，且A2 → A1：
      Key1=A1, Key2=A2，两个决定子均是超键，故R∈BCNF。

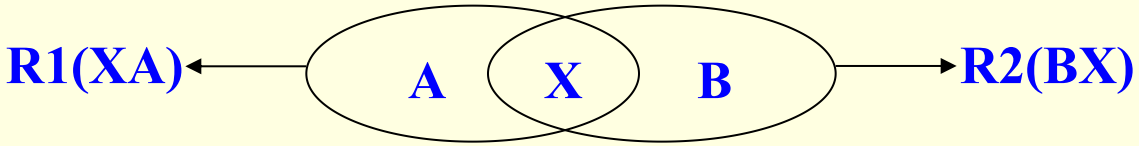
10.2.2 范式

- F. 关系模式无损分解成BCNF的启发式算法
  - 设关系模式R(XAB)，其中，X, A, B均是R的属性（子集），
    - (1) 针对冒犯BCNF 条件的某个非平凡函数依赖：

      $X \rightarrow A$ （即X不是超键），将R分解成两个模式：

      R1(XA) 和 R2(BX)。

      一般地，R1∈BCNF，而有可能R2∉BCNF。



本人俗称为“细胞分裂法”

    - (2) 针对不属于BCNF的模式R2（和R1）重复步骤(1)，直到全部模式属于 BCNF为止。

10.2.2 范式

- 例：M (title, year, length, color, star, star-gender, star-add, studio, studio-add, studio-class)
  - 根据通常的语义，有以下函数依赖：

    (1) star → star-gender, star-add

    (2) studio → studio-add, studio-class

    (3) title, year → length, color, studio

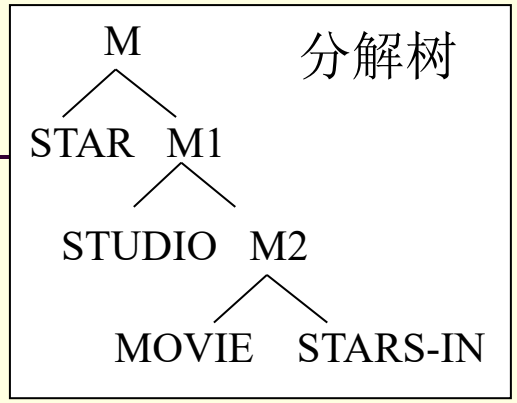
    故，M的键 Key = {title, year, star}。

    就是说，上述3个非平凡函数依赖的决定子都不是超键。

由于函数依赖(1), (2), (3)均冒犯BCNF 条件，故M ∉BCNF。

10.2.2 范式

(1) star → star-gender, star-add
(2) studio → studio-add, studio-class
(3) title, year → length, color, studio



- 用“细胞分裂法”进行模式分解：
- 对式(1)【 star → star-gender, star-add 】，
  将M分解成：

  STAR (star, star-gender, star-add) ∈BCNF

  M1 (title, year, length, color, studio, studio-add, studio-class, star)

  因M1的Key未变，且式(2), (3)仍成立，故M1∉BCNF。
- 对式(2)【studio → studio-add, studio-class 】，将M1分解成：

  STUDIO（studio, studio-add, studio-class）∈BCNF

  M2（title, year, length, color, star, studio）

  因M2的Key未变，且式(3)仍成立，故M2∉BCNF。
- 对式(3)【 title, year → length, color, studio 】，将M2分解成：

  MOVIE（title, year, length, color, studio）∈BCNF

  STARS-IN（title, year, star）∈BCNF ——全键关系模式。

故将M规范化成BCNF的无损分解：ρ ={STAR, STUDIO, MOVIE, STARS-IN}

关系模式规范化的基本步骤

- 函数依赖范畴内，关系模式规范化程度的各种测度（范式）：

1NF

↓ 消除非主属性对键的部分函数依赖

2NF

↓ 消除非主属性对键的传递函数依赖

3NF

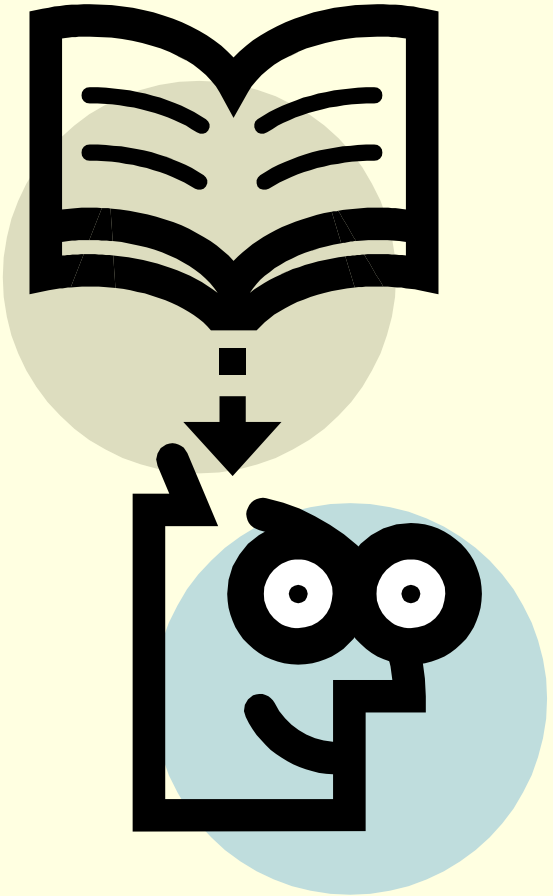
↓ 消除主属性对键的部分和传递函数依赖

BCNF

注：1NF ⊃ 2NF ⊃ 3NF ⊃ BCNF

目录 Contents

- 10.1 关系模式规范化概述
- 10.2 函数依赖与范式
- 10.3 模式分解理论（简介）



10.4 模式分解理论

- 模式分解理论（仅在函数依赖范畴内讨论）
  - [定义] 逻辑蕴涵（logical implication）
    - 设一关系模式R(U,F)，其中U为R的属性集，F为R的函数依赖集。若对任意的关系r∈R，函数依赖 $X \rightarrow Y$在r上成立，则称F逻辑蕴涵$X \rightarrow Y$，记为：$F \models X \rightarrow Y$。
    - e.g. 若 $F = \{A \rightarrow B, B \rightarrow C\}$, 则 $F \models A \rightarrow C$。
  - [定义]函数依赖集F的闭包（the closure of a set of functional dependencies F）：F^+
    - 函数依赖集F所逻辑蕴涵的函数依赖的全体，称F的闭包，记为F^+。即：$F^+ = \{X \rightarrow Y \mid F \models X \rightarrow Y\}$。

10.4 模式分解理论

- Armstrom公理（Armstrong's axioms——inference rules developed by William W. Armstrong in his 1974 paper）：三条推理规则
  - 自反律（reflexivity rule）：
    - If $Y \subseteq X \subseteq U$, Then $X \rightarrow Y$.
  - 扩展律（augmentation rule）：
    - If $X \rightarrow Y$ and $Z \subseteq U$, Then $X \cup Z \rightarrow Y \cup Z$.
  - 传递律（transitivity rule）：
    - If $X \rightarrow Y$ and $Y \rightarrow Z$, Then $X \rightarrow Z$.
- 定理：Armstrom公理是正确的（sound）【运用推理规则不会产生任何不正确的函数依赖】和完备的（complete）【 F+ 可由Armstrong公理的推理规则从F导出】.

10.4 模式分解理论

- [定义] 关系模式R的一个分解 / 关系r在U_i上的投影 / 函数依赖集F在U_i上的投影
  - 给定一个关系模式R(U, F), 若此关系模式R可用一个关系模式的集合$\{R_1(U_1), R_2(U_2), \dots, R_k(U_k)\}$，$\bigcup_{i=1}^{k} U_i = U$ 来取代，则此集合称为R的一个分解，记为：$\rho = \{ R_1, R_2, \dots, R_k \}$。
  - 关系$r \in R$在U_i上的投影定义为：$\Pi_{Ui}(r) = \{ t[U_i] \mid t \in r \}$。
  - 函数依赖集F在U_i上的投影定义为：

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

10.4 模式分解理论

- [定义] 无损分解（lossless decomposition）
  - 设ρ = { $R_1(U_1), R_2(U_2), \ldots, R_k(U_k)$ }是对关系模式R(U)的一个分解，若对任意的关系 r∈R 满足：
    $r = \Pi_{U1}(r) \bowtie \Pi_{U2}(r) \bowtie \ldots \bowtie \Pi_{Uk}(r)$，则称ρ是对关系模式R的一个无损分解。

- [定义] 保持依赖分解（dependency-preserving decomposition）
  - 设ρ = { $R_1(U_1), R_2(U_2), \ldots, R_k(U_k)$ }是关系模式R(U, F)的一个分解，若 $\Pi_{U1}(F) \cup \Pi_{U2}(F) \cup \ldots \cup \Pi_{Uk}(F) \models F$ 成立，则称ρ是对关系模式R的一个保持（函数）依赖分解。

10.4 模式分解理论

- 分解的条件 / 准则
  - 无损分解——起码：决定能否分解；
  - 保持依赖分解——理想：决定分解的好坏。

- 结论
  - 总有将一个关系模式分解成3NF的无损且保持依赖的分解。
  - 总有将一个关系模式分解成BCNF（甚至4NF）的无损分解【但不一定是保持依赖的分解】。

模式分解的综合题

- 给定包含五个原子属性的关系模式R(ABCDE)，且其函数依赖集$F_R = \{A \rightarrow BC, CD \rightarrow E\}$。试回答问题：

1）分别求出模式R的键、包含属性个数最少的超键、包含属性个数最多的超键，要求在推理过程中说明相应的推理规则和理由；

2）判断模式R最高属于第几范式，要求根据各种范式的定义分别说明理由；

3）给出将模式R经一次“模式分解”就规范化到BCNF的一个无损分解ρ，要求给出分解过程并说明分解后的模式属于BCNF的理由；进一步说明ρ不是一个保持（函数）依赖分解的理由；

4）给出将模式R经两次“模式分解”规范化到BCNF的一个无损分解φ，要求给出分解过程并说明分解后的模式属于BCNF的理由。

模式分解的综合题 $F_R = \{A \rightarrow BC, CD \rightarrow E\}$

1）分别求出模式R的键、包含属性个数最少的超键、包含属性个数最多的超键，要求在推理过程中说明相应的推理规则和理由；

根据扩展律，由 A→BC 可得 AD→BCD；

根据分裂规则，由 AD→BCD 可得 AD→BC 与 AD→CD；

根据传递规则，由 AD→CD 和 CD→E 可得 AD→E；

根据合并规则，由 AD→BC 和 AD→E 可得 AD→BCE，

即 AD 满足键的决定性条件。

AD 的任何真子集 A 或 D 均不能决定 E，即 AD 满足键的最小性条件。

根据键的定义（决定性、最小性条件）可知：模式 R 的**键=AD；**

键就是超键，故 R 中**包含属性个数最少的超键=AD；**

模式的全部属性是超键，故 R 中**包含属性个数最多的超键=ABCDE。**

模式分解的综合题 $F_R = \{A \rightarrow BC, CD \rightarrow E\}$

2）判断模式R最高属于第几范式，要求根据各种范式的定义分别说明理由；

因非平凡依赖 A→BC（或 CD→E）的决定子 A（或 CD）不是超键，故 R∉BCNF；

并且该函数依赖的被决定子 BC（或 E）不是主属性，故 R∉3NF；

又因 A→BC 意味着非主属性 BC 部分依赖于键 AD，故 R∉2NF；

根据题意，R 的全部属性均是原子属性，故 R∈1NF；

结论：R 最高属于 1NF。

【R∈BCNF的定义】模式R的任一非平凡函数依赖X→A满足决定子X必是超键。
【R∈3NF 的定义】模式R的任一非平凡函数依赖X→A 满足下列两个条件之一：
(1) 决定子X是超键, (2) 被决定子A是主属性。
【R∈2NF 的定义】模式R的每个非主属性均完全函数依赖于键。
【R∈1NF 的定义】模式R的任一关系实例r中的属性值均是原子数据。

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

知识回忆

模式分解的综合题 $F_R = \{A \rightarrow BC, CD \rightarrow E\}$

3）给出将模式R经一次“模式分解”就规范化到BCNF的一个无损分解ρ，要求给出分解过程并说明分解后的模式属于BCNF的理由；进一步说明ρ不是一个保持（函数）依赖分解的理由；

针对冒犯 BCNF 条件的 $A \rightarrow BC$，将模式 R 无损分解为：R1(ABC) 和 R2(ADE)。

考察 R1(ABC)是否属于 BCNF：根据 F_R 可知，R1 的函数依赖集 $F_{R1}=\{A \rightarrow BC\}$，

故 R1 的键为 A（也是超键）；由 BCNF 定义可知：$R1 \in BCNF$。

考察 R2(ADE)是否属于 BCNF：从题 1 可知 $AD \rightarrow E$（或：F_R 逻辑蕴含 $AD \rightarrow E$），

R2 的函数依赖集 $F_{R2}=\{AD \rightarrow E\}$，R2 的键为 AD（也是超键），故 $R2 \in BCNF$。

故将 R 经一次模式分解规范化到 BCNF 的无损分解：$\rho = \{R1(ABC), R2(ADE)\}$。

又从上述分析可知：$\Pi_{ABC}(F_R) = F_{R1} = \{A \rightarrow BC\}$，且 $\Pi_{ADE}(F_R) = F_{R2} = \{AD \rightarrow E\}$，

故 $\Pi_{ABC}(F_R) \cup \Pi_{ADE}(F_R) = \{A \rightarrow BC, AD \rightarrow E\}$，但它并不逻辑蕴含分解前关系模式

R 的函数依赖集 F_R 中原有的函数依赖 $CD \rightarrow E$，即：$\Pi_{ABC}(F_R) \cup \Pi_{ADE}(F_R) \not\models F_R$，

故 ρ 不是一个保持（函数）依赖的分解。

模式分解的综合题 $F_R = \{A \rightarrow BC, CD \rightarrow E\}$

4）给出将模式R经两次“模式分解”规范化到BCNF的一个无损分解φ，要求给出分解过程并说明分解后的模式属于BCNF的理由。

针对冒犯 BCNF 条件的 CD→E，将 R 无损分解为：R1(CDE) 和 R2(ABCD)。

考察 R1(CDE)是否属于 BCNF：根据 F_R 可知，$F_{R1} = \{CD \rightarrow E\}$，

故 R1 的键为 CD（也是超键）；由 BCNF 定义可知：R1∈BCNF。

考察 R2(ABCD)是否属于 BCNF：因(ABCD)上仅有 A→BC，即 $F_{R2} = \{A \rightarrow BC\}$，

故 R2 的键为 AD；但 A→BC 的决定子 A 不是超键，故 R2∉BCNF。

针对冒犯 BCNF 条件的 A→BC，进一步将 R2 无损分解为：R21(ABC) 和 R22(AD)。

考察 R21(ABC)是否属于 BCNF：因(ABC)上仅有 A→BC，即 $F_{R21}=\{A \rightarrow BC\}$，

故 R21 的键为 A（也是超键）；决定子 A 是超键，故 R21(ABC)∈BCNF。

考察 R22(AD)是否属于 BCNF：只有两个属性的关系模式必定属于 BCNF，

故 R22(AD)∈BCNF。

故将 R 经两次模式分解成 BCNF 的无损分解为：φ = {R1(CDE), R21(ABC), R22(AD)}。

The End

- 教材Page 221： 习题10中的第8题(1) - (7)
  【补充】将(7)中的关系模式R无损分解到BCNF，并说明此分解不能保持依赖的理由。
- **提醒：请在截止时间（11月26日23:59）之前提交答案！**