# 第10章 关系数据库设计理论

北京理工大学 计算机学院
张文耀
zhwenyao@bit.edu.cn

# 主要内容

- 1 关系模型的存储异常
- 2 函数依赖
- 3 函数依赖公理
- 4 模式分解
- 5 关系模式的规范化
- 6 多值依赖和4NF
- 7 连接依赖和投影-连接范式(Project-Join NF)
- 8 小结

# 1.关系模型的存储异常

- 关系
  - 关系模型的唯一数据结构
  - 可以描述实体，也可以描述实体间的联系
- 关系模式
  - 一个关系对应一个关系模式
- 关系数据库
  - 一组关系
- 关系数据库模式
  - 一组关系模式

- 关系数据库模式设计
  - 由于认识或看问题的方法不同，一个数据库可以设计出不同的数据库模式。
  - 数据库模式的设计是数据库应用系统开发的核心问题。
  - 数据库模式设计的好坏是系统成败的关键所在。
  - 关系数据库模式设计（即数据库的逻辑设计），是从各种可能的关系模式组合中选取一组关系模式来构成一个数据库。
- 如何设计一个好的关系数据库模式？
  - 先得解决关系模式设计的性能好坏问题

- 关系模式设计示例
  - 需求：某学校图书馆要建一个图书数据库，其中借阅管理包括借书证号(CARDNO)、借书人姓名(NAME)、借书人所在单位(DEPT)、所在单位负责人(MN)、图书编号(BNO)、借阅日期(DATE)等信息。
  - 关系模式：借阅图书登记可以用如下关系模式来描述：
    **BORROW（CARDNO，NAME，DEPT，MN，BNO，DATE）**

- 关系—BORROW表

| CARDNO | NAME | DEPT | MN | BNO | DATE |
|---|---|---|---|---|---|
| R001 | 李晓鹏 | 计算机 | 张宏军 | TP31-125 | 20070123 |
| R001 | 李晓鹏 | 计算机 | 张宏军 | TP32-007 | 20061112 |
| R001 | 李晓鹏 | 计算机 | 张宏军 | TP12-233 | 20070405 |
| R002 | 王一鸣 | 计算机 | 张宏军 | TP51-211 | 20080209 |
| R002 | 王一鸣 | 计算机 | 张宏军 | TP31-254 | 20080209 |
| R003 | 刘明川 | 无线电 | 范和平 | TP23-126 | 20071011 |
| R003 | 刘明川 | 无线电 | 范和平 | TP23-023 | 20080321 |
| R003 | 刘明川 | 无线电 | 范和平 | TP31-125 | 20080321 |

- **BORROW表好不好？**
  - 存在大量的数据冗余
    - 借阅人的一些基本信息被重复存储
    - 造成存储空间的浪费
    - 存在潜在的数据不一致问题
  - 存在插入异常问题
    - 没有借书的读者信息不能存入数据库。因为关键字是由CARDNO和BNO组成的联合键，根据实体完整性规则，关键字不能为空。

- 存在删除异常问题
  - 当借书人归还所借的书后，需要从借阅关系中删除相关信息
  - 如果借阅人把所借的书全部还清了，则连同借阅人及所在单位的所有信息都将被一起删除
- 存在更新异常问题
  - 如果所在单位或单位负责人变了，需要修改该借书人在**DEPT**或**MN**属性上的值。但由于数据冗余，有关借书人所有元组中的单位或单位负责人的信息都要修改。
  - 这不仅增加了更新代价，而且可能出现部分数据被修改，而另外的没有修改，导致不一致问题。

- **E.F.Codd**将数据冗余、插入异常、删除异常、更新异常统称为存储异常。
- 为什么会出现存储异常？
  - 因为在数据间存在着一定的依赖关系。如借书关系中，
    - 借阅人和借阅人所在单位
    - 借阅人所在单位与单位负责人
    - 借书证号、书号与借阅日期之间

    都存在着依存关系。
  - **BORROW**模式没有很好地反映这些关系，因此不是一个好的设计。

- 在现实世界中，实体和实体间，以及实体内部的属性值之间存在着相互依赖又相互制约的关系，这种关系称为数据依赖。
- 设计一个好的关系模式，应了解数据间存在哪些数据依赖。
- 最重要的数据依赖有：
  - 函数依赖
  - 多值依赖
  - 连接依赖

# 2.函数依赖

- 函数依赖(Functional Dependency, FD)是现实世界中最广泛存在的一种数据依赖，
  - 是现实世界属性间相互联系的抽象；
  - 是数据内在的性质；
  - 它表示了关系中属性间的一种制约关系。
- 函数依赖的相关概念
  - 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
  - 传递函数依赖与直接函数依赖

- **定义10.1**　设关系模式**R(U)**，**X,Y⊆U**，**r**是**R(U)**上的任一关系。对任意元组**t1** 、**t2∈r**，如果**t1**、**t2** 在**X**上的属性值相等，**t1**、**t2**在**Y**上的属性值亦相等，则称**X**函数决定**Y**,或**Y**函数依赖于**X**，记为**FD X→Y**
  - 称**X**为决定因素，或称**X**为函数依赖的左部，**Y**为函数依赖的右部。

函数依赖是一个关系模式上的（TABLE级）

- **BORROW关系的函数依赖**
  - **CARDNO→NAME；**
    每个借书证号可以唯一确定一个读者
  - **CARDNO→DEPT；**
    每个借书证号可以唯一确定读者所在单位
  - **CARDNO→MN；**
    借书证号可以唯一确定读者所在单位负责人
  - **DEPT→MN；**
    单位可以唯一确定单位负责人
  - **(CARDNO,BNO)→DATE ；**
    借阅日期由证号和图书编号共同决定

一一对应或者多对一

单变量和多变量

- 函数依赖是指R的所有关系实例均要满足的约束条件，而不是指关系模式R的某个或某些关系实例满足的约束条件。
- 函数依赖是语义范畴的概念，只能根据现实世界中数据间的语义确定函数依赖 翻译：凭感觉
  - 如函数依赖：姓名→年龄，只有规定不允许有同名同姓的人的情况下才成立。
  - 为满足这种语义约束，当装入元组到数据库时，就要检查这种约束条件，使相应元组上的属性值满足规定的函数依赖。
  - 设计者必须非常清楚这种语义约束，才能设计出满足要求的数据库模式

- 定义**10.2** 设**FD X→Y**，如果**Y⊄X**，则称**FD X→Y**为非平凡的函数依赖；否则，若**Y⊆X**，称**FD X→Y**为平凡的函数依赖。

不一定一一对应，就是传统意义上的函数映射

  - 平凡的函数依赖是对模式**R**上的所有关系都成立的函数依赖。
  - 平凡函数依赖不反映新的语义
- 例：在关系**SC(Sno, Cno, Grade)**中，

  非平凡函数依赖：**(Sno, Cno) → Grade**

  平凡函数依赖：**(Sno, Cno) → Sno**

  **(Sno, Cno) → Cno**
- 若非特别声明总是讨论非平凡函数依赖。

平凡是自己对自己的一部分，没啥意义

- **定义10.3**
  - 设FD X→Y，如果对任意的X’ ⊂X，X’→Y都不成立，则称FD X→Y是完全函数依赖；
  - 若有X的真子集X’(X’⊂X)使得X’ →Y成立，则称FD X→Y是部分函数依赖，即Y函数依赖于X的一部分。
  - 完全函数依赖记为： $X \xrightarrow{f} Y$
  - 部分函数依赖记为： $X \xrightarrow{p} Y$

full
partly

  - 当X是单属性时，FD X→Y是完全函数依赖总是成立的；
  - 若X是属性集时，X→Y可能是完全函数依赖，也可能是部分函数依赖 。

- 在借阅关系**BORROW**中，存在
  - 完全函数依赖：

    **(CARDNO, BNO)→DATE**

    **CARDNO→NAME**

    **CARDNO→DEPT**

    **CARDNO→MN**

    **DEPT→MN**
  - 部分函数依赖：

    **(CARDNO, BNO)→NAME**

    **(CARDNO, BNO)→DEPT**

    **(CARDNO, BNO)→MN**

- **定义10.4** 设关系模式R，X、Y、Z是R的属性子集，若FD X→Y，Y $\not\rightarrow$ X，Y→Z，则必有FD X→Z，称FD X→Z为传递函数依赖。

X到Y可能多对一

- 传递函数依赖表示为： $X \xrightarrow{t} Z$
- 在借阅关系BORROW中，
  - 存在函数依赖CARDNO→DEPT，DEPT→MN，而DEPT $\not\rightarrow$ CARDNO，因此有传递依赖CARDNO→MN
- 注：如果X→Y，且Y→X，则X←→Y；此时，如果Y→Z，由此得出的X→Z，称为Z直接依赖于X。

一一对应特殊情况

  - 如果 CARDNO→NAME，NAME→CARDNO，NAME→DEPT，则CARDNO→DEPT为直接函数依赖

- 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
  - 传递函数依赖与直接函数依赖
- 分析关系中各个属性间的函数依赖，有助于了解数据本身的意义。
- 设计一个好的数据库模式，必须了解现实世界中数据间的依存和制约关系，正确确定属性间的函数依赖关系。

如何确定有哪些函数依赖？

# 函数依赖的逻辑蕴涵

- 一个关系模式**R**上的任一关系**r(R)**，在任意给定的时刻都有它所满足的一组函数依赖集**F**。
- 若关系模式**R**上的任一关系都能满足一个确定的函数依赖集**F**，则称**F**为**R**满足的函数依赖集。
- 对于给定的一组函数依赖，需要判断另外一些函数依赖是否成立。

  从已知一手函数依赖
  推导出更高级的依赖

  - 例如，已知关系模式**R**上的函数依赖集为**F**，**F**中有函数依赖**X→Y**，**X→Z**，问**X→YZ**是否成立？
  - 这就是函数依赖的逻辑蕴涵所要研究的问题

- **定义10.5** 设函数依赖集F和关系模式R(U)，属性集X，Y⊆U，关系模式R满足F 。
  - 如果关系模式R满足FD X→Y，则称F逻辑蕴涵FD X→Y，或称X→Y逻辑蕴涵于F。
  - 记为 F |= X→Y。

只要R满足一个FD，那么就确定DF可以放在F里，不管我们之前是否发现过

- **定义10.6** 设函数依赖集F，所有被F逻辑蕴涵的函数依赖称为F的闭包，记为$F^+$。
  - $F^+$ 可表示为：$F^+ = \{ X \rightarrow Y \mid$ 所有F 蕴涵的FD $X \rightarrow Y\}$
  - 显然$F \subseteq F^+$

判断是否蕴含，可以直接到闭包里找，，但是这种方法不太现实

- 例: F={AB→C, C→B}是R(A, B, C)上的一组函数依赖，则有：

$F^+$ ={A→A，AB→A，AC→A，ABC→A，
B→B，AB→B，BC→B，ABC→B，
C→C，AC→C，BC→C，ABC→C，
AB→AB，ABC→AB，
AC→AC，ABC→AC，
BC→BC，ABC→BC，
ABC→ABC，
AB→C，AB→AC，AB→BC，AB→ABC，
C→B，C→BC，AC→B，AC→AB，AC→ABC }

- 定义10.7 设关系模式R（U，F），U是R的属性全集，F是R的函数依赖集，X是U的子集。如果满足条件：

  —(1) $X \rightarrow U \in F^+$；

  —(2) 不存在$X' \subset X$且$X' \rightarrow U \in F^+$成立。

  则称X为模式R的一个候选键。

- 函数依赖及函数依赖的闭包概念，给关系模式的键下了一个较为严密的形式化定义。

- 要确定关系模式的键，需要确定模式中属性间的函数依赖情况。

- **第3章对键的形式化定义**

设关系模式R(U)，K⊆U，r是R上的任一关系，若对r中的任意二个不同的元组满足：

（1）$t_1[K] \neq t_2[K]$；

（2）不存在K′⊂K而使得$t_1[K'] \neq t_2[K']$成立；

则称K是R的键。

候选键是关系模式中所有能够起到标识作用的键

这个定义只描述了键的性质，没有给出识别键的方法。

# 3.函数依赖公理

- 给出函数依赖集F，
  如何确定F是否蕴涵FD X→Y？
  - 运用一组推理规则来进行确定
  - 这组推理规则是在1974年由Armstrong提出来的，称为Armstrong公理
- Armstrong公理
  - 一组推理规则
  - 用于推导函数依赖的逻辑蕴含
  - 理论上是有效的、完备的
  - 基本内容：3条公理＋3个推论

- **Armstrong公理**
  - 设关系模式**R(U，F)**，并且**X、Y、Z**和**W**是**U**的子集
  - **A1** 自反律**(Reflexivity)**

    若$Y \subseteq X \subseteq U$, 则 $F \models X \rightarrow Y$;
  - **A2** 增广律**(Augmentation)**

    若$X \rightarrow Y$且$Z \subseteq U$，则 $F \models XZ \rightarrow YZ$;
  - **A3** 传递律**(Transitivity)**

    若$X \rightarrow Y$, $Y \rightarrow Z$，则 $F \models X \rightarrow Z$.

- 自反律证明：若$Y \subseteq X \subseteq U$, 则 $F|=X \rightarrow Y$；
  - 若$t_1[X]=t_2[X]$，则X的任意子集在$t_1$、$t_2$上的属性值相同；因$Y \subseteq X$，所以有$t_1[Y] = t_2[Y]$。根据函数依赖的定义，$X \rightarrow Y$成立，因此 $F|=X \rightarrow Y$

- 在一个关系中，
  - 两个元组不可能在属性集X上的值相同、而在X的子集Y上的值不同。
  - 所以，公理A1是平凡的。

- 增广律证明：若$X \rightarrow Y$且$Z \subseteq U$，则 $F \models XZ \rightarrow YZ$
  - 设$t_1[XZ]=t_2[XZ]$，则有$t_1[X]t_1[Z]=t_2[X]t_2[Z]$，则$t_1[X]=t_2[X]$，$t_1[Z]=t_2[Z]$
  - 由已知条件$X \rightarrow Y$可知，$t_1[Y]=t_2[Y]$。
  - 因此，有$t_1[YZ]=t_1[Y]t_1[Z]=t_2[Y]t_2[Z]=t_2[YZ]$
  - 根据函数依赖的定义，若$t_1[XZ]=t_2[XZ]$，有$t_1[YZ]=t_2[YZ]$，故$XZ \rightarrow YZ$成立
- 增广律表明，在一个函数依赖的左部增加R的属性，所得函数依赖依然成立。（$XZ \rightarrow Y$）
- 公理A2还可以写为：
  若$X \rightarrow Y$且$W \subseteq Z \subseteq U$，则 $F \models XZ \rightarrow YW$

- 传递律证明：若$X \rightarrow Y, Y \rightarrow Z$，则 $F \models X \rightarrow Z$；
  - 已知$X \rightarrow Y$，则$t_1[X]=t_2[X]$时，有$t_1[Y]=t_2[Y]$。
  - 已知$Y \rightarrow Z$，则$t_1[Y]=t_2[Y]$时，有$t_1[Z]=t_2[Z]$。

  显然，$X \rightarrow Z$ 成立

- 传递律没有传递函数依赖那么严格。

- 根据A1，A2，A3这三条推理规则可以得到下面三条很有用的推论：
  - 合成规则

    可以将两个函数值绑定起来由X决定

    由X→Y， X→Z，则X→YZ （A2， A3）
  - 分解规则

    决定可以延伸到函数值的子集

    由X→Y及 Z⊆Y，则X→Z （A1， A3）
  - 伪传递规则

    由X→Y，YZ→W，则XZ→W （A2， A3）

- 推论的证明
  - 合成规则
    若**X→Y**，由公理**A2**，有**XX→XY**；若**X→Z**，由公理**A2**，有**XY→YZ**；由公理**A3**，有**XX→YZ**，即**X→YZ**。
  - 分解规则
    若**Z⊆Y**，由公理**A1**，有**Y→Z**；已知**X→Y**，所以**X→Z**。
  - 伪传递规则
    若**X→Y**，由公理**A2**，有**XZ→YZ**；已知**YZ→W**，根据公理**A3**，**XZ→W**成立。
- 根据合成规则和分解规则，可得

  $X \rightarrow A_1 A_2 \ldots A_k$ 的充分必要条件是 $X \rightarrow A_i (i=1,2,3,\ldots)$

- **Armstrong公理的正确性已经证明**
- **Armstrong公理是否完备？**
  - 如果能用公理从一组函数依赖**F**推导出它所蕴涵的所有依赖，则是完备的。
  - 要说明公理的完备性，可对已知函数依赖集**F**求$F^+$；如果可以用公理推导出$F^+$中所有函数依赖，则可以证明公理是完备。
  - 但是，由$F=\{X\rightarrow A_1,\ldots, X\rightarrow A_n\}$出发可以推导出$2^n$个不同的函数依赖。该问题属于**NP**完全问题。
  - 因此，由**F**求$F^+$是行不通的。
  - 为了说明公理的完备性，引入属性闭包的概念。

# 属性闭包

- 定义10.8 设关系模式R(U，F)，$U=A_1,A_2,\ldots,A_n$，$X\subseteq U$。所有用公理推出的函数依赖$X\rightarrow A_i$中$A_i$的属性集合称为属性集X关于函数依赖集F的闭包，记为$X_F^+$。

  $X_F^+=\{A_i \mid$ 所有用公理由F推出的$X\rightarrow A_i\}$。

  - 显然，由自反律知道$X\subseteq X_F^+$ 。
  - 例如，设R(A, B, D, E,H)，

    R上的函数依赖集F={A→D，AB→DE，E→H}
  - 若X=A，$(A)_F^+$ =AD。
  - 若X=AB，$(AB)_F^+$ =ABDEH。

- **定理10.1** 设关系模式R(U,F)，X，Y⊆U；能够由Armstrong公理从F导出X→Y成立的充要条件是 $Y \subseteq X_F^+$

  【证明】设Y= $A_1,A_2,\ldots,A_k$，$A_i \in U(i=1,2,\ldots,k)$。

  - 先证充分性。设$Y \subseteq X_F^+$，由$X_F^+$的定义知，X→ $A_i$ (i=1,2, … , k)是由Armstrong公理从F中导出的，根据合成规则，X→Y成立。
  - 证必要性。设X→Y是用Armstrong公理从F推导出的。根据分解规则，有X→ $A_i$ (i=1,2, … ,k)成立，由$X_F^+$的定义知，$A_i \in X_F^+$ (i=1,2, … ,k)，即$Y \subseteq X_F^+$ 。

  证毕。

- 有了属性闭包的概念，根据定理**10.1**可以将

  判定**X→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。

X可以推出闭包，如果Y属于闭包，则X可以退出Y

# 属性闭包算法

- 如何计算$X_F^+$——算法10.1
  - 输入：模式R的属性全集U，U上的函数依赖集F，属性集X
  - 输出：属性集X的闭包$X^+$

(1) 初值 $X^{(0)} = X$，i=0；

(2) $X^{(i+1)} = X^{(i)} \cup Z$；其中$Z=\{A \mid$ 存在$V \rightarrow W \in F$，$V \subseteq X^{(i)}$ 且 $A \in W$ 而 $A \notin X^{(i)}\}$

(3) 判断$X^{(i+1)} = X^{(i)}$或$X^{(i+1)} = U$是否成立，若成立转(5)

(4) i=i+1，转(2)；

(5) 输出$X^+$的结果$X^{(i+1)}$。

- 算法的实质：只要**F**中的函数依赖的左部属性包含在中间结果$X^{(i)}$中，就可以将没有出现在$X^{(i)}$中的右部属性**A**并入$X^{(i)}$中。**X→A**显然成立。
- 算法的循环次数是有限的，因为每次至少添加一个属性，最多计算**|U|-|X|+1**次。
- 算法是正确的。

【例】设关系模式R(U,F), U={A,B,C,D,E,G},
F={ AB→C,BC→D,ACD→B,D→EG,BE→C,CE→AG},
求 $(BD)^+$

- 令X=BD

(1) $(X)^{(0)}=BD$

(2) $(X)^{(1)}=BDEG$，因为D→EG；$X^{(0)} \neq X^{(1)}$。

(3) $(X)^{(2)}=BCDEG$，因为BE→C；$X^{(1)} \neq X^{(2)}$

(4) $(X)^{(3)}= ABCDEG$，因为CE→AG；$X^{(3)} \neq X^{(2)}$，
但是$X^{(3)}$ 已包含了R中的所有属性，结束。

(5) 输出结果：$(BD)^+ = ABCDEG$

**BD→ABCDEG**

# Armstrong公理的完备性证明

- 定理10.2 Armstrong公理是正确的，完备的
  - 正确性：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
  - 完备性的逆否命题：若存在函数依赖$X \rightarrow Y$不能由F从Armstrong公理导出，那么它必然不为F所蕴含。
  - 完备性的证明思路：证明逆否命题（存在性问题）。给定R(U,F)，如果能够证明以下两点，则完备性得证

    1）存在r(R)满足F中的所有函数依赖；

    2）不能用Armstrong公理推导出来的$X \rightarrow Y$，在r(R)上不成立。

## 1）构造 r(R)满足F中的所有函数依赖

- 构造一张二维表r，它由下列两个元组构成，可以证明r必是R(U,F)的一个关系，即满足F中的所有函数依赖 。

| | $X_F^+$ | $U-X_F^+$ |
|---|---|---|
| t1 | 11……1 | 00……0 |
| t2 | 11……1 | 11……1 |

- 设FD $W \rightarrow Z \in F$，W在r中有两种情况：

  (1) $W \subseteq X_F^+$ 。根据属性闭包的定义，有$X \rightarrow W$，又因$W \rightarrow Z$，由函数依赖的传递性得FD $X \rightarrow Z$成立。根据属性闭包的定义，有$Z \subseteq X^+$。由r的构造知$W \rightarrow Z$在r上成立。

  (2) $W \not\subseteq X_F^+$ ，则在r中有$t1[W] \neq t2[W]$，因此，$W \rightarrow Z$在r上总是成立的。

由(1)和(2)知，对F中的任意函数依赖在r上都是成立的，即r满足F。

2）证明不能用Armstrong公理推导出来的$X \rightarrow Y$，在r(R)上不成立

| | $X_F^+$ | $U-X_F^+$ |
|---|---|---|
| t1 | 11……1 | 00……0 |
| t2 | 11……1 | 11……1 |

- 因为$X \rightarrow Y$不能由F从公理推出，由定理10.1知，$Y \not\subseteq X^+$。
- t1[X]=t2[X]，而$Y \not\subseteq X^+$，则t1[Y]≠t2[Y]，即r不满足$X \rightarrow Y$。

证毕

定理10.1 设关系模式R(U,F)，X，$Y \subseteq U$；能够由Armstrong公理从F导出$X \rightarrow Y$成立的充要条件是$Y \subseteq X_F^+$

# Armstrong公理的完备性说明

- “蕴含” 和 “导出”的等价

  “通过F推导出的函数依赖” ≡ “F所蕴涵的函数依赖”

- 完备性证明用到了

  实际上的应用，前面的证明不用看。

  - 属性闭包$X_F^+$

    $X_F^+ = \{A_i \mid$ 所有$F \models X \rightarrow A_i\}$。

  - 函数依赖集的闭包$F^+$

    $F^+ = \{ X \rightarrow Y \mid$ 用公理从F导出的所有FD $X \rightarrow Y\}$

- 对于一个给定的关系模式R以及R的函数依赖集F，判断某一个函数依赖$X \rightarrow Y$是否被F所逻辑蕴涵，只需求$X_F^+$，然后判断$Y \subseteq X_F^+$是否成立；不用计算$F^+$，再判断判断$X \rightarrow Y$是否是$F^+$ 中的成员。

# 函数依赖集的等价和覆盖

- 定义10.9 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。
- $F^+ = G^+$ 的充分必要条件是：$F \subseteq G^+$同时$G \subseteq F^+$

证：必要性显然，只证充分性。

可以互相导出

（1）任取 $X \rightarrow Y \in F^+$ ，有 $Y \subseteq X_F^+$；

（2）若 $F \subseteq G^+$，则$X_F^+ \subseteq X_{G+}^+$ ；

（3）由（1）和（2）得 $Y \subseteq X_F^+ \subseteq X_{G+}^+$，

进而有 $X \rightarrow Y \in (G^+)^+ = G^+$。因此 $F^+ \subseteq G^+$；

（3）同理可证$G^+ \subseteq F^+$；

所以$F^+ = G^+$。

- 要判定$F \subseteq G^+$，只须逐一对F中的函数依赖$X \rightarrow Y$，考察 Y 是否属于$X_{G+}^+$ 就行了。

  这给出了判断两个函数依赖集等价的可行算法。

# 最小函数依赖集

- 定义**10.10**　如果函数依赖集**F**满足下列条件，则称**F**为一个最小函数依赖集 或 最小覆盖

  **(1)** **F**中的所有函数依赖其右部都是单属性；

  **(2)** 对**F**中的任一函数依赖**X→A**，**F–{X→A}**与**F**不等价；

  **(3)** 对**F**中的任一函数依赖**X→A**，**F–{X→A}∪{Z→A }**与**F**不等价，其中 **Z** 是 **X** 的真子集（**Z⊂X**）。

- 最小函数依赖的特点
  - 右部都是单属性
  - 没有多余的函数依赖
  - 左部都没有多余属性

【例】 对于关系模式S<U，F>，其中

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

F={ Sno→Sdept, Sdept→Mname, (Sno,Cno)→Grade}

设F'={Sno→Sdept, Sdept→Mname, Sno→Mname,
(Sno, Cno)→ Grade, (Sno, Sdept)→Sdept}

F是最小覆盖，而F'不是。

因为：F'-{Sno→Mname}与F'等价

F'-{(Sno，Sdept)→Sdept}与F'等价

F'-{(Sno，Sdept)→Sdept}

∪{Sno→Sdept} 与F'等价

- **定理10.3** 每一个函数依赖集F均等价于一个最小函数依赖集$F_m$

证明：构造性证明法，找出F的一个最小依赖集。

1)将F中所有函数依赖转换为右部为单属性的形式，令结果为G。G与F等价。

2)检查G中的每个函数依赖$X \rightarrow A$，若$G-\{X \rightarrow A\}$与G等价，则从G中去掉$X \rightarrow A$。G仍然与F等价

3)检查G中的每个函数依赖$X \rightarrow A$的左部属性，若$Z \subset X$且$G-\{X \rightarrow A\} \cup \{Z \rightarrow A\}$与G等价，则用$Z \rightarrow A$取代$X \rightarrow A$。G仍然与F等价。

最后得到的就是F的最小依赖集$F_m$。证毕。

- 1)的等价性
  $X \rightarrow Y$ ($Y=A_1A_2 \ldots A_k$) 等价于 $\{X \rightarrow A_j \mid j=1,2,\ldots,k\}$

- 2)判别G-{X→A}与G等价
  令H=G-{X→A}，计算$X_H^+$；
  若$A \in X_H^+$，则H=G-{X→A}与G等价。

- 3)判别H=G-{X→A}∪{Z→A}与G等价
  计算$Z_G^+$；若$A \in Z_G^+$，则两者等价。因为
  $A \in Z_G^+$，说明$G \mid = Z \rightarrow A$;
  另外，$Z \rightarrow A \in H$，$Z \subset X$，有$X \rightarrow A \in H$

- 函数依赖集F的最小函数依赖集$F_m$不是唯一的。
  - $F_m$的结果与F中各个函数依赖的考察顺序有关。
  - 若最后计算得到的$F_m$与原来的F相同，说明F本身就是一个最小依赖集
- 定理10.3的证明过程
  - 给出了一个计算最小函数依赖集的有效算法
  - 也可以用来验证F是否为最小依赖集

【例】由关系模式R(U,F)，U={A,B,C,D,E}，
F={AB→C，B→D，C→E，EC→B，AC→B}，求$F_m$。

（1）将F中的所有函数依赖转换为右部为单属性的形式G。
G=F，因为F已经是满足条件的形式

（2）去掉多余的函数依赖。考察各个函数依赖是否可以去掉。
令H=G-{X→Y}，计算$X_H^+$；若$Y \in X_H^+$，则去掉X→Y

- H=F-{AB→C}，$AB_H^+$ =ABD，不包含C，AB→C保留
- H=F-{B→D}，$B_H^+$ =B，不包含D，B→D保留
- H=F-{C→E}，$C_H^+$=C，不包含E，C→E保留
- H=F-{EC→B}，$EC_H^+$ =EC，不包含B，EC→B保留
- H=F-{AC→B}，$AC_H^+$ =ABCDE，包含B，AC→B去掉

得到 G={AB→C，B→D，C→E，EC→B}

（3）去掉函数依赖左部多余的属性

**G={AB→C，B→D，C→E，EC→B}**

对于Z⊂X，X→Y∈G，令H=G-{X→Y}∪{Z→Y}，计算$Z_G^+$；若Y∈$Z_G^+$，则用Z→Y取代X→Y。

- 检查AB→C (X=AB, Y=C)
  - 令Z=A，$A_G^+$={A}，不包含C，AB→C中B不能去掉
  - 令Z=B，$B_G^+$={B,D}，不包含C，AB→C中A不能去掉
- 检EC→B (X=EC, Y=B)
  - 令Z=E，$E_G^+$={E}，不包含B，EC→B中C不能去掉
  - 令Z=C，$C_G^+$={CEB}，包含B，EC→B中E可以去掉（用C→B取代EC→B）

最后：$F_m$={AB→C，B→D，C→E，C→B}

# 4.模式分解

- 一个大的模式在属性间可能会存在复杂的数据依赖关系，带来存储异常等问题。因此需要将一个大的关系模式用几个较小的模式代替，即进行模式分解
- 定义10.11 设关系模式$R(U)$，$\rho=\{R_1(U_1), R_2(U_2), \ldots, R_k(U_k)\}$是一个关系模式的集合，若$\cup_{i=1}^{k}U_i=U$，则称$\rho$是关系模式$R(U)$的一个分解。

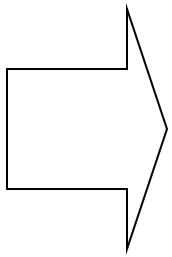
- 一个关系模式可以有多种不同的分解。
  - 例如关系模式**E(EmpNo, Title, Salary)**

    $\rho_1$={$E_1$ (EmpNo, Title ), $E_2$ (Title , Salary)}

    $\rho_2$={$E_1$ (EmpNo, Title ), $E_2$(EmpNo, Salary)}

    $\rho_3$={$E_1$ (EmpNo, Salary), $E_2$ (Title, Salary)}
- 模式分解不是任意的，分解后的模式应该与原模式等价。等价包括两方面的含义：
  - 分解后的模式不损失任何信息，即：对于同样的数据，前后查询结果应该一样。
  - 分解后的模式应该保持原来的函数依赖，即：不丢失数据间的约束 (语义)。

| EMPNO | TITLE | SALARY |
|---|---|---|
| E001 | 教授 | 2500 |
| E002 | 副教授 | 2200 |
| E003 | 研究员 | 2500 |
| E004 | 讲师 | 2000 |
| E005 | 副教授 | 2200 |
| E006 | 副教授 | 2200 |



| EMPNO | TITLE |
|---|---|
| E001 | 教授 |
| E002 | 副教授 |
| E003 | 研究员 |
| E004 | 讲师 |
| E005 | 副教授 |
| E006 | 副教授 |

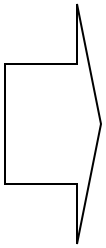
| TITLE | SALARY |
|---|---|
| 教授 | 2500 |
| 副教授 | 2200 |
| 研究员 | 2500 |
| 讲师 | 2000 |

$\rho_1$

分解前后等价

更新时更优

| EMPNO | TITLE | SALARY |
|---|---|---|
| E001 | 教授 | 2500 |
| E002 | 副教授 | 2200 |
| E003 | 研究员 | 2500 |
| E004 | 讲师 | 2000 |
| E005 | 副教授 | 2200 |
| E006 | 副教授 | 2200 |



| EMPNO | SALARY |
|---|---|
| E001 | 2500 |
| E002 | 2200 |
| E003 | 2500 |
| E004 | 2000 |
| E005 | 2200 |
| E006 | 2200 |

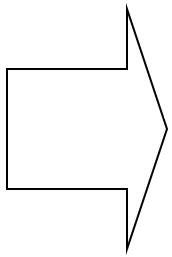
| EMPNO | TITLE |
|---|---|
| E001 | 教授 |
| E002 | 副教授 |
| E003 | 研究员 |
| E004 | 讲师 |
| E005 | 副教授 |
| E006 | 副教授 |

$\rho_2$

不损失任何信息

但没有保持函数依赖

| EMPNO | TITLE | SALARY |
|---|---|---|
| E001 | 教授 | 2500 |
| E002 | 副教授 | 2200 |
| E003 | 研究员 | 2500 |
| E004 | 讲师 | 2000 |
| E005 | 副教授 | 2200 |
| E006 | 副教授 | 2200 |



| EMPNO | SALARY |
|---|---|
| E001 | 2500 |
| E002 | 2200 |
| E003 | 2500 |
| E004 | 2000 |
| E005 | 2200 |
| E006 | 2200 |

| TITLE | SALARY |
|---|---|
| 教授 | 2500 |
| 副教授 | 2200 |
| 研究员 | 2500 |
| 讲师 | 2000 |

$\rho_3$

有信息损失

是一个不好的分解

- 将一个模式分解为多个模式时必须遵循一定的准则，使分解后的模式不损失原有的信息，同时保持数据间原有的函数依赖关系。
- 怎么做到无损？
  ——采用无损连接分解
- 怎么保持函数依赖？
  ——是否保持，只能分解完了再判断
  ——可以采用一定的算法，保持某些分解的函数依赖

# 无损连接分解

- 分解后的关系通过自然连接如果能够恢复为原来的关系，即保证连接后的关系与原关系完全一致，则称该分解为无损连接分解。
- 定义**10.12**　设关系模式**R(U)**，**F**是**R**上的函数依赖集，$\rho=\{R_1, R_2, \ldots, R_K\}$是**R**的一个分解，如果对**R**的任一满足**F**的关系**r**有：

$$r=\Pi_{R1}(r) \bowtie \Pi_{R2}(r) \bowtie \cdots \bowtie \Pi_{Rk}(r)$$

则称$\rho$是满足**F**的无损连接**(Lossless join)**分解。

- r在$\rho$上的投影连接记为$m_\rho(r)$，则

  $m_\rho(r) = \bowtie_{i=1}^{k} \prod_{Ri}(r)$
- 满足无损连接的条件可表示为

  $r = m_\rho(r)$
- 例：SL(Sno,Sdept,Sloc)分解为

  $\rho_1$={NL(Sno, Sloc), DL(Sdept, Sloc)}

  $\rho_2$={ND(Sno, Sdept), NL(Sno, Sloc)}

| Sno | Sdept | Sloc |
|---|---|---|
| 95001 | CS | A |
| 95002 | IS | B |
| 95003 | MA | C |
| 95004 | IS | B |
| 95005 | PH | B |

SL

| Sno | Sloc |
|---|---|
| 95001 | A |
| 95002 | B |
| 95003 | C |
| 95004 | B |
| 95005 | B |

NL

| Sdept | Sloc |
|---|---|
| CS | A |
| IS | B |
| MA | C |
| PH | B |

DL

| Sno | Sdept | Sloc |
|---|---|---|
| 95001 | CS | A |
| 95002 | IS | B |
| 95002 | PH | B |
| 95003 | MA | C |
| 95004 | IS | B |
| 95004 | PH | B |
| 95005 | IS | B |
| 95005 | PH | B |

NL ⋈ DL

增加了三个元组，而且出现了数据的不一致性

$\rho_1$不是无损连接分解

| Sno | Sdept | Sloc |
|---|---|---|
| 95001 | CS | A |
| 95002 | IS | B |
| 95003 | MA | C |
| 95004 | IS | B |
| 95005 | PH | B |

SL

| Sno | Sdept | Sloc |
|---|---|---|
| 95001 | CS | A |
| 95002 | IS | B |
| 95003 | MA | C |
| 95004 | IS | B |
| 95005 | PH | B |

ND ⋈ NL=SL

| Sno | Sdept |
|---|---|
| 95001 | CS |
| 95002 | IS |
| 95003 | MA |
| 95004 | IS |
| 95005 | PH |

ND

| Sno | Sloc |
|---|---|
| 95001 | A |
| 95002 | B |
| 95003 | C |
| 95004 | B |
| 95005 | B |

NL

$\rho_2$是一个无损连接分解

- 如何判断一个分解是否是无损连接分解？
  - 通过实际关系来验证是不可靠的。
  - 必须有一个具体的可操作的方法——算法10.2
- 算法10.2 判断一个分解是否是无损连接分解
  - 输入：关系模式$R(A_1,A_2,\ldots,A_n)$，R上的函数依赖集F，以及R的一个分解$\rho=\{R_1, R_2, \ldots, R_K\}$。
  - 输出：分解$\rho$是否具有无损连接性。
  - 方法：

    (1) 构造一个k行n列的表T，每一行对应分解后的一个子模式，每行中列的值为$T_{ij}$

$$T_{ij}=\begin{cases} a_j & (A_j \in R_i) \\ b_{ij} & (A_j \notin R_i) \end{cases}$$

(2) 逐个考察F中的每个函数依赖X→Y，在表中寻找在X的属性上相等的行，若在这些行上属性Y的值不相等，则将Y的值改为相等。具体改法为：若其中某一列上有$a_j$，则改相应行的列值为$a_j$，否则将其都改为这些行中具有最小下标的$b_{ij}$的值

- 反复考察F中的每个函数依赖，直到不能使表T中的值改变为止

(3) 检查结果表，如果表中有$a_1, a_2, \ldots, a_n$的行，则分解$\rho$具有无损连接性；否则，分解$\rho$不是无损的。

- 算法10.2的示例（P220－221）
  - R(U,F)
  - U={A,B,C,D,E}
  - F={A→C，B→C，C→D，DE→C，CE→A}
  - $\rho$ ={$R_1$, $R_2$, $R_3$, $R_4$, $R_5$}，
    $R_1$=AD，$R_2$=AB，$R_3$=BE，$R_4$=CDE，$R_5$=AE

- 构造表格T

| | A | B | C | D | E |
|---|---|---|---|---|---|
| R1=AD | a1 | b12 | b13 | a4 | b15 |
| R2=AB | a1 | a2 | b23 | b24 | b25 |
| R3=BE | b31 | a2 | b33 | b34 | a5 |
| R4=CDE | b41 | b42 | a3 | a4 | a5 |
| R5=AE | a1 | b52 | b53 | b54 | a5 |

- 考察A→C, B→C, C→D, DE→C , CE→A

| | A | B | C | D | E |
|---|---|---|---|---|---|
| R1=AD | a1 | b12 | b13 | a4 | b15 |
| R2=AB | a1 | a2 | b23/b13 | b24/a4 | b25 |
| R3=BE | b31/a1 | a2 | b33/b13/a3 | b34/a4 | a5 |
| R4=CDE | b41/a1 | b42 | a3 | a4 | a5 |
| R5=AE | a1 | b52 | b53/b13/a3 | b54/a4 | a5 |

- **算法10.2**
  - 能正确判断一个分解$\rho$是否是无损分解。证明见教材。
  - 适应于一般的模式分解检验。对于仅含两个关系模式的分解，可以采用定理10.5来检验。
- **定理10.5** 设关系模式R的一个分解$\rho=\{R_1,R_2\}$，F是R上的函数依赖集。如果

$$F|=(R_1\cap R_2)\rightarrow(R_1-R_2)$$

或 $F|=(R_1\cap R_2)\rightarrow(R_2-R_1)$，

则$\rho$具有无损连接性

证明：用算法**10.2** 构造一个二行三列的矩阵如下：

| | $R_1 \cap R_2$ | $R_1 - R_2$ | $R_2 - R_1$ |
|---|---|---|---|
| $R_1$ | aaa…a | aaa…a | bbb…b |
| $R_2$ | aaa…a | bbb…b | aaa…a |

只要$(R_1 \cap R_2) \rightarrow (R_1 - R_2)$或$(R_1 \cap R_2) \rightarrow (R_2 - R_1)$在**F**中，都会出现全为**a**的一行，即$\rho$具有无损连接性。

# 分解的保持依赖性

- 模式分解
  - 数据被分解
  - 函数依赖也被分解
    - 分解后的函数依赖是否与原函数依赖集等价，这就是分解的保持依赖性问题。
- 例：**SL(Sno,Sdept,Sloc)**分解为

**$\rho_1$={NL(Sno, Sloc), DL(Sdept, Sloc)}**

**$\rho_2$={ND(Sno, Sdept), NL(Sno, Sloc)}**

**$\rho_3$={ND(Sno, Sdept), DL(Sdept, Sloc)}**

**F={Sno→Sdept, Sdept→Sloc, Sno→Sloc}**

| Sno | Sdept | Sloc |
|---|---|---|
| 95001 | CS | A |
| 95002 | IS | B |
| 95003 | MA | C |
| 95004 | IS | B |
| 95005 | PH | B |

SL

| Sno | Sloc |
|---|---|
| 95001 | A |
| 95002 | B |
| 95003 | C |
| 95004 | B |
| 95005 | B |

NL

| Sdept | Sloc |
|---|---|
| CS | A |
| IS | B |
| MA | C |
| PH | B |

DL

| Sno | Sdept | Sloc |
|---|---|---|
| 95001 | CS | A |
| 95002 | IS | B |
| 95002 | PH | B |
| 95003 | MA | C |
| 95004 | IS | B |
| 95004 | PH | B |
| 95005 | IS | B |
| 95005 | PH | B |

NL ⋈ DL

增加了三个元组，而且出现了数据的不一致性

$\rho_1$不是无损连接分解

SL

| Sno | Sdept | Sloc |
|---|---|---|
| 95001 | CS | A |
| 95002 | IS | B |
| 95003 | MA | C |
| 95004 | IS | B |
| 95005 | PH | B |

ND

| Sno | Sdept |
|---|---|
| 95001 | CS |
| 95002 | IS |
| 95003 | MA |
| 95004 | IS |
| 95005 | PH |

NL

| Sno | Sloc |
|---|---|
| 95001 | A |
| 95002 | B |
| 95003 | C |
| 95004 | B |
| 95005 | B |

$ND \bowtie NL = SL$ $\rho_2$是一个无损连接分解

F={Sno→Sdept, Sdept→Sloc, Sno→Sloc}

这种分解没有保持原关系中的函数依赖，SL中的函数依赖 Sdept→Sloc 没有投影到关系模式ND、NL上

SL

| Sno | Sdept | Sloc |
|---|---|---|
| 95001 | CS | A |
| 95002 | IS | B |
| 95003 | MA | C |
| 95004 | IS | B |
| 95005 | PH | B |

ND

| Sno | Sdept |
|---|---|
| 95001 | CS |
| 95002 | IS |
| 95003 | MA |
| 95004 | IS |
| 95005 | PH |

DL

| Sdept | Sloc |
|---|---|
| CS | A |
| IS | B |
| MA | C |
| PH | B |

$\rho_3$是一个无损连接分解

ND ⋈ DL=SL

这种分解方法既具有无损连接性，
又保持了函数依赖。

F={Sno→Sdept, Sdept→Sloc, Sno→Sloc}

- 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。
  - 保持函数依赖的分解也不一定具有无损连接性。

- 定义10. 13 设关系模式R的一个分解 $\rho=\{R_1, R_2, \ldots, R_p\}$，F是R上的函数依赖集。令

$$\Pi_{Ri}(F)=\{X\rightarrow Y \mid X\rightarrow Y\in F^{+}\text{且}\ XY\subseteq R_i\},\ (1\leqslant i\leqslant p),$$

称$\Pi_{Ri}(F)$为F在$R_i$ 上的投影。

- 若$\Pi_{Ri}(F)$是F在$R_i$上的投影，则称$\Pi_{Ri}(F)$在$R_i$上是可施加的。

- 定义10. 14 设关系模式R的一个分解 $\rho=\{R_1, R_2, \ldots, R_p\}$，F是R上的函数依赖集。令F在$R_i$ (1≤i≤p)上投影的集合

$$G=\bigcup_{i=1}^{p} \Pi_{Ri}(F)$$

若$G^+=F^+$，则称分解ρ保持函数依赖集F。

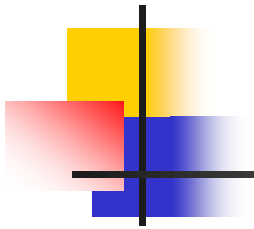
- 要判断一个分解是否保持函数依赖，
  - 首先计算F在分解ρ的每个关系模式上的投影$\Pi_{Ri}(F)$
  - 然后判断$F^+=G^+$是否成立
  - 实际上仅需检查G|= F是否成立就足够了

- **算法10.3** 检验一个分解是否保持函数依赖。
  - 输入：分解$\rho=\{R_1, R_2, \ldots, R_p\}$及其函数依赖集F。
  - 输出：分解$\rho$是否保持F。
  - 方法：

    (1) 计算$\prod_{Ri}(F)$，$1 \le i \le p$。

    (2) 令$G=\cup_{i=1}^{p} \prod_{Ri}(F)$，对F中的每个FD $X \rightarrow Y$，在G中计算$X^+$。若$Y \not\subset X^+$，检查结束，输出为假，分解$\rho$不具有保持依赖性。

    (3) 输出为真，分解$\rho$具有保持依赖性。



- **算法10.3示例**
  **P223（e248）【例10-7】**

  另请判断该分解是否有损

# 5.关系模式的规范化

- 设计不好的模式会出现存储异常，影响数据库的性能。
- 为了设计好的模式，人们研究了规范化理论
  - **1971**年，**E.F.Codd**首先提出了规范化的问题，给出了范式**(Normal Form, NF)**的概念，根据关系模式所达到的不同约束提出了**1NF**、**2NF**、**3NF**；
  - **1974**年，**Codd**和**Boyce**提出**BCNF(Boyce-Codd Normal Form)**。
  - 之后又研究了**4NF**、**5NF**。

- 规范化理论可以用来改造关系模式
  通过分解关系模式，消除不合适的数据依赖，以解决插入异常、删除异常、更新异常和数据冗余等问题
- 规范化理论的基本概念——范式
  - 范式就是规范化的关系模式，也就是满足一定要求的关系模式。
  - 满足不同程度要求的关系模式为不同的范式。
  - 数据库的范式从低到高有：
    **1NF、 2NF、3NF、BC范式(BCNF)、4NF。**
  - 不同的范式其规范化程度是不同的。

- 一个数据库模式可以通过模式分解，从低一级的范式转化为若干个高一级的范式。
- 从低一级的范式通过分解达到高一级范式的过程称为关系模式的规范化。
- 规范化理论的核心
  ——关系模式的规范化

# 第一范式(1NF)

- 如果关系模式R的每一个属性对应的域值都是不可再分的，称模式R属于第一范式，简记为R∈1NF。
- 第一范式是最低级别的关系模式
- 如果数据库模式R中的每个关系模式都属于1NF，则数据库模式R∈1NF。
- 一个数据库系统中的关系至少应该是1NF的，这是关系作为二维表的基本要求。
  - 不满足1NF的数据库模式不能称为关系数据库
  - 满足1NF的关系模式不一定是好的关系模式

# 第二范式(2NF)

- 设关系模式R，A是R中的属性，F是R上的函数依赖集。如果A包含在R的某个候选键中，称A为主属性，否则称A为非主属性。
- 如果一个关系模式R∈1NF，且所有非主属性都完全依赖于R的每一个候选键，则R∈2NF。
- 如果数据库模式R中的每个关系模式都属于2NF，则数据库模式R∈2NF。

- 判定一个关系模式是否属于**2NF**，
  - 需要了解关系模式的属性间存在哪些依赖
  - 根据数据依赖关系，找出关系模式的候选键，
  - 确定哪些属性是主属性，哪些属性是非主属性，
  - 确定非主属性与候选键之间是否存在完全函数依赖关系，以判定该模式是否属于**2NF**

- 借书关系

  **BORROW(CARDNO, NAME, DEPT, MN, BNO, DATE)**

  - 是**1NF**，不是**2NF**
  - **BORROW**的候选键：**(CARDNO、BNO)**
  - 函数依赖：**CARDNO→NAME**，**CARDNO→DEPT**，
    **CARDNO→MN**，**DEPT→MN**，
    **(CARDNO、BNO)→DATE**
  - 属性**NAME**、**DEPT**和**MN**部分依赖于**BORROW**的候选键。因此，**BORROW∉2NF**，存在插入异常、删除异常、数据冗余、修改复杂等问题。（问题的原因在于部分依赖）

- 解决方法：**BORROW**分解为两个关系模式，以消除这些部分函数依赖
  - **LOANS(CARDNO，NAME，DEPT，MN) ∈2NF**
  - **BORROW′ (CARDNO，BNO，DATE) ∈2NF**

| CARDNO | NAME | DEPT | MN |
|---|---|---|---|
| R001 | 李晓鹏 | 计算机系 | 张宏军 |
| R002 | 王一鸣 | 计算机系 | 张宏军 |
| R003 | 刘明川 | 无线电系 | 范和平 |

| CARDNO | BNO | DATE |
|---|---|---|
| R001 | TP31-125 | 20070123 |
| R001 | TP32-007 | 20061112 |
| R001 | TP12-233 | 20070405 |
| R002 | TP51-211 | 20080209 |

- 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。
- **2NF**不能完全消除关系模式中的各种存储异常
  - 例如：

| CARDNO | NAME | DEPT | MN |
|---|---|---|---|
| R001 | 李晓鹏 | 计算机系 | 张宏军 |
| R002 | 王一鸣 | 计算机系 | 张宏军 |
| R003 | 刘明川 | 无线电系 | 范和平 |

# 第三范式(3NF)

- **LOANS存在存储异常的原因**
  **LOANS(CARDNO，NAME，DEPT，MN) ∈2NF**
  - 存在传递函数依赖：
    **CARDNO→DEPT，DEPT→MN，且DEPT↛ CARDNO，**
    因此，有**FD CARDNO→MN**，一个传递函数依赖
- 如果将**LOANS**进一步分解，消除传递函数依赖，即可消除仍然存在的存储异常

| CARDNO | NAME | DEPT |
|---|---|---|
| R001 | 李晓鹏 | 计算机系 |
| R002 | 王一鸣 | 计算机系 |
| R003 | 刘明川 | 无线电系 |

| DEPT | MN |
|---|---|
| 计算机系 | 张宏军 |
| 无线电系 | 范和平 |

- 定义**10.18** 设**R∈1NF**，若在**R**中没有非主属性传递依赖于**R**的候选键，则关系模式**R∈3NF** 。如果数据库模式**R**中每一关系模式都是**3NF**，则数据库模式**R∈3NF**。
- 一个**2NF**的关系模式不一定属于**3NF**。**2NF**仅消除了非主属性和主属性之间的部分依赖，如果存在传递依赖还需进一步规范化。
- 一个关系模式若是**3NF**的，则一定属于**2NF**。
- 一个关系模式若不是**2NF**的，则一定也不属于**3NF**。

- **定理10.6** 若任一关系模式R∈3NF，则R∈2NF。

  证明：用反证法。令R上的键为K。假设R∈3NF但R∉2NF，则有R中非主属性A部分依赖于关键字K。那么，存在K的真子集K’，使得K’→A。由于K’⊂K，有K→K’，但K’→ K。于是,/ K→K’, K’→ K，K’→A，并且A∉K，因而A传递依赖于K，即R∉3NF，与假设矛盾。

  证毕。

- 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候选键。

# Boyce-Codd范式(BCNF)

- 第三范式
  - 消除了非主属性和主属性间的部分函数依赖和传递函数依赖；
  - 在一定程度上解决了存储异常问题；
  - 没有考虑主属性间的函数依赖问题；
  - 在主属性间存在部分函数依赖和传递函数依赖，也会导致存储异常问题

- 【例】关系模式R(City, Street, Zip)，
  - R上的函数依赖为 {(City,Street)→Zip, Zip →City}
  - 候选键为(City, Street)或(Street, Zip)
  - R中没有非主属性，不存在非主属性与主属性间的部分函数依赖和传递函数依赖，因此R∈3NF
  - 由于Zip →City，因此 (Street, Zip)→ City为部分函数依赖
    - 造成City值多次重复，会引起更新异常
      如若要修改北京对应的邮政编码为110000，必须修改北京地区的所有邮政编码

- 针对3NF的问题，Boyce和Codd提出了修正的第三范式——BCNF。
- 定义10.19　若R∈1NF，而且R中没有任何属性传递依赖于R中的任一关键字，则关系模式R属于Boyce-Codd范式（BCNF）。
- 如果数据库模式R中的每个关系模式R都属于BCNF，则数据库模式R∈BCNF。
- BCNF不但排除了非主属性对主属性的传递依赖，也排除了主属性间的传递依赖。

- 对于非**BCNF**的关系模式可以通过模式分解达到**BCNF**
  - 例如：**3NF**关系模式**R(City, Street, Zip)**，可以分解为：
    **R1(Zip,City)**，**R2(Street,Zip)**
  - **R1**和**R2**都属于**BCNF**

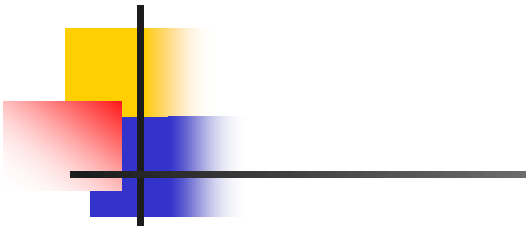
思考题：上面的分解是否无损？

## 一个更直观的等价的BCNF定义

- 定义10.20 设关系模式R∈1NF，F是R上的函数依赖集，对于F中的每一个函数依赖X→Y，必有X是R的一个超键（候选键），则R∈BCNF
  - 如果R∈BCNF，则R上的每一个函数依赖中的决定因素都包含（或者是）候选键。
  - 所有函数依赖都是关于超键（候选键）的依赖。

思考题：怎么判断X是否包含候选键？

- **3NF和BCNF的关系**
  - 若**R ∈ BCNF**，则一定有**R ∈ 3NF**。
    - 每一个决定属性集（因素）都包含（候选）键
    - **R**中的所有属性**(**主，非主属性**)**都完全函数依赖于键
  - 若**R ∈ 3NF**，则不一定有**R ∈ BCNF**
    - 如果**R ∈ 3NF**，且**R**只有一个候选键，则**R**必属于**BCNF**
  - **3NF**仅消除了非主属性的存储异常，
    - **3NF**可能存在主属性对键的部分依赖和传递依赖
  - **BCNF**消除了整个关系模式的存储异常。



- **3NF**和**BCNF**是在函数依赖的条件下对模式分解所能达到的分离程度的测度。
- 在函数依赖的范围内，**BCNF**已达到了关系模式的最大分离，已经消除了插入、删除异常，是函数依赖范围内能够达到的最高范式。
- 在实际应用中，**3NF**和**BCNF**的关系模式一般都能满足用户对数据的处理要求。
- 问题：如何使非**3NF**和非**BCNF**的关系模式达到**3NF**和**BCNF**？

# 模式分解算法

- **算法10.4** 生成3NF的算法（合成算法）

  给定关系模式R(U,F)

  1)寻找没有出现在F中的R的属性，将这些属性单独组成一个关系模式，并从R中去掉；

  2)令F:=F∪{U→Z},Z是没有出现在U中的附加属性；

  3)计算F的最小函数依赖集，结果仍记为F；

  4)若有X、Y为函数依赖的左部且X↔Y，则将这些函数依赖分为一组，其中X和Y可以相同；

  5)将每组函数依赖组成一个关系模式，并将附加属性Z去掉；

  6)若有一个关系模式所含属性与R的属性相同，输出该模式；否则输出这些模式。算法结束。

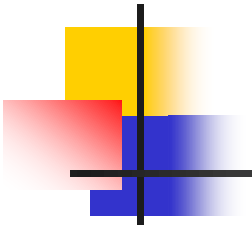
- 【例】已知R(U, F)，U=ABCDE，
  F={ A→CD，B→E，AC→D }

(1) 关系模式R的所有属性都在F中。
令F=F∪{ABCDE→Z};

(2) 计算F的最小依赖集。结果为
G={ A→C，B→E，A→D，AB→Z };

(3) G中依赖按照左部等价分组为：
{ A→C，A→D }；{B→E}；{AB→Z };

(4) 去掉外部属性Z后，R的3NF的分解为：
$R_1$=ACD，$R_2$=BE，$R_3$=AB。



- 算法10.4生成的分解是3NF的，且具有无损连接性和保持依赖性。
- 对于BCNF则没有类似的算法。
- 如果要求模式分解具有无损连接性和保持依赖性，可以达到3NF，但不一定能够达到BCNF。
- 如果要求模式分解仅具有无损连接性，则可以达到BCNF。即：规范化到BCNF，不一定能保持函数依赖。

- **算法10.5** 生成BCNF的算法

  输入：关系模式R（U，F）。

  输出：达到BCNF的R的一个无损分解。

  1)设ρ={R(U，F)}。

  2)检查ρ中的各关系模式是否为BCNF，若是，则算法终止。

  3)若ρ中有$R_i(U_i，F_i)$不属于BCNF，即$F_i$中有FD $X \rightarrow Y$，而X不是$R_i$的键，分解$R_i$为$R_{i1}=XY$，$R_{i2}=R_i-Y$。

  4) 用$\{R_{i1}，R_{i2}\}$代替ρ中的$R_i$，返回(2)

- 【例】关系模式R(U, F)，U=ABC，
  F={A→C，B→C}。
  - R∉BCNF，因有FD A→C，A不是R的键。
  - 根据FD A→C，将R分解为{AC，AB}
    分解后的二个模式都属于BCNF，损失函数依赖。
  - 若先检查FD B→C，则分解结为{BC，AB}。

R属于哪一级NF？R的键是什么？

R的3NF分解结果是什么样的？

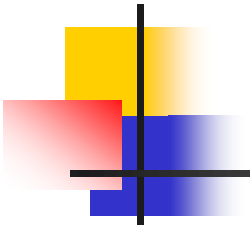
# 6.多值依赖和4NF

- 属于BCNF的关系模式是否完美？
- 例: 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。关系模式Teaching(C, T, B)

| 课程C | 教师T | 参考书B |
|---|---|---|
| 物理 | 李 勇<br>王 军 | 普通物理学<br>光学原理<br>物理习题集 |
| 数学 | 李 勇<br>张 平 | 数学分析<br>微分方程<br>高等代数 |

- 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 |
| … | … | … |



- **Teaching具有唯一候选键(C, T, B)，即全键**
  **Teaching∈BCNF**
- **Teaching模式中存在的问题**
  - 数据冗余度大：有多少名任课教师，参考书就要存储多少次；
  - 插入操作复杂：当某课程增加一名任课教师时，该课程有多少本参照书，就必须插入多少个元组；
  - 删除操作复杂：某一门课要去掉一本参考书，该课程有多少名教师，就必须删除多少个元组；
  - 修改操作复杂：某一门课要修改一本参考书，该课程有多少名教师，就必须修改多少个元组；

- **Teaching存在问题的原因——存在多值依赖**
- **定义10.21** 设**R(U)**是一个关系模式，**X**和**Y**是**U**的子集，**Z=U-X-Y**。对于关系**r(R)**中任意两个元组**s**和**t**，若**s[X]=t[X]**，在**r**中存在元组**w**和**v(w**、**v**可以与**s**、**t**相同)，使得：

  **w[X]=s[X]，w[Y]=t[Y]，w[Z]=s[Z]；**

  **v[X]=t[X]， v[Y]=s[Y]， v[Z]=t[Z]；**

  **(**即交换**s**和**t**元组的**Y**值所得的两个新元组必在**r**中**)**，则关系**r(R)**满足多值依赖**(MVD) X→→Y**，称**X**多值决定**Y**或**Y**多值依赖于**X**。

- 多值依赖示例

| | 课程X | 教员Y | 参考书Z |
|---|---|---|---|
| w | 物 理 | 李 勇 | 普通物理学 |
| s | 物 理 | 李 勇 | 光学原理 |
| | 物 理 | 李 勇 | 物理习题集 |
| t | 物 理 | 王 军 | 普通物理学 |
| v | 物 理 | 王 军 | 光学原理 |
| | 物 理 | 王 军 | 物理习题集 |

| | X | Y | Z |
|---|---|---|---|
| s | x | y1 | z1 |
| t | x | y2 | z2 |
| w | x | y2 | z1 |
| v | x | y1 | z2 |

**Teaching(C,T,B)有多值依赖：C →→T**

**X=C;**

**Y=T;**

**Z=B;**

- 等价的多值依赖的另一个定义
  定义10.22 设R(U)是一个关系模式，X和Y是U的子集，Z=U-X-Y，r是R上的一个关系。多值依赖X→→Y成立，当且仅当对于给定的一个x值，有一组y的值，且这组y值与r中的其它属性Z无关。此时，称X多值决定Y或Y多值依赖于X。
- Teaching（C, T, B）存在多值依赖
  - C→→T
    每一个C值，T有一组值与之对应，而不论B取何值
  - C→→B
    每一个C值，B有一组值与之对应，而不论T取何值

- 平凡多值依赖和非平凡的多值依赖
  - 若**X→→Y**，而**Z= Φ** ，则称**X→→Y**为平凡的多值依赖
  - 否则称**X→→Y**为非平凡的多值依赖

- 同函数依赖类似，多值依赖也有逻辑蕴涵性。
  - 从一组已知的多值依赖可以推导出它所蕴涵的多值依赖。

# 多值依赖的推理公理

- r是模式R上的关系，且W、X、Y、Z是R的子集
  - M1：自反律 若$Y \subseteq X$ 则$X \rightarrow\rightarrow Y$。
  - M2：增广律 若$X \rightarrow\rightarrow Y$，$W \subseteq Z$，则$XZ \rightarrow\rightarrow YW$。
  - M3：相加律 若$X \rightarrow\rightarrow Y$、$X \rightarrow\rightarrow Z$，则$X \rightarrow\rightarrow YZ$。
  - M4：投影律 若$X \rightarrow\rightarrow Y$、$X \rightarrow\rightarrow Z$，
    则$X \rightarrow\rightarrow Y \cap Z$、 $X \rightarrow\rightarrow Y-Z$。
  - M5：传递律 若$X \rightarrow\rightarrow Y$、$Y \rightarrow\rightarrow Z$，则$X \rightarrow\rightarrow Z-Y$。
  - M6：伪传递律 若$X \rightarrow\rightarrow Y$、$YW \rightarrow\rightarrow Z$，
    则$XW \rightarrow\rightarrow Z-(YW)$
  - M7：互补律 若$X \rightarrow\rightarrow Y$、$Z=R-(XY)$，则$X \rightarrow\rightarrow Z$
  - M8：重复律 若$X \rightarrow Y$，则$X \rightarrow\rightarrow Y$。
  - M9：结合律 若$X \rightarrow\rightarrow Y$，$Z \rightarrow W$，
    其中$W \subseteq Y$和$Y \cap Z=\Phi$，则$X \rightarrow W$。

- 多值依赖具有对称性
  若**X→→Y**，则**X→→Z**，其中**Z＝U－X－Y**
  多值依赖的定义中不仅涉及属性组 **X**和 **Y**，而且涉及**U**中其余属性**Z**

- 函数依赖是多值依赖的特殊情况。
  若**X→Y**，则**X→→Y**。(反之不成立)
  函数依赖属性间的依赖关系仅与**XY**有关，与其余属性无关

# 4NF

- 若关系模式存在多值依赖，还需要进一步规范化
- 定义10.23 设关系模式R∈1NF，F是R上的FD和MVD集。如果对于R上的任何一个非平凡的多值依赖X→→Y，X是R的一个超键，则R∈4NF。如果数据库模式*R*中的每一个关系模式R都属于4NF，那么，数据库模式*R*∈4NF。
- X是R的一个超键，X包含R的候选键， X→Y
- 4NF的属性之间不存在非平凡的多值依赖。
  - 若关系模式R上存在非平凡的多值依赖，该多值依赖只能是函数依赖，且该函数依赖的决定因素应是R的超键。

- 如果**R ∈ 4NF，则R ∈ BCNF**

  不允许有非平凡且非函数依赖的多值依赖

  允许平凡的多值依赖

  允许决定因素包含超键的函数依赖

- 可采用分解的方法消去非平凡且决定因素不包含超键的的多值依赖
  - 例：**Teaching(C,T,B) ∉ 4NF**，因为存在
    非平凡的多值依赖**C→→T**，且**C**不是候选键
  - 可把**Teaching**分解为如下两个关系模式：
    **CT(C, T) ∈ 4NF**，（**C→→T**平凡多值依赖）
    **CB(C, B) ∈ 4NF**，（**C→→B**平凡多值依赖）

- **算法10.6** 通过分解生成4NF的算法。
  - 输入：关系模式R，函数依赖和多值依赖集F。
  - 输出：达到4NF的R的一个无损分解。$\rho=\{R_1, R_2, \ldots, R_K\}$，$R_i \in 4NF$（$1 \le i \le k$）
  - (1) 若$R \in 4NF$，算法终止，$\rho=\{R\}$。
  - (2)若$\rho$中有$R_i \notin 4NF$，即有MVD $X \rightarrow\rightarrow Y$，$XY \neq R_i$ 且$X \not\rightarrow R_i$，则分解$R_i$为：$R_{i1}=R_i-Y$和$R_{i2}=XY$，用$R_{i1}$和$R_{i2}$代替$\rho$中的$R_i$。
  - (3) 若$\rho$中所有$R_i \in 4NF$，输出$\rho$，否则转(2)，继续进行分解，直到使所有的关系模式都成为4NF。

【例】关系模式R(A, B, C, D, E, I)，R的依赖集F={A→→BCD,B→AC,C→D}，将R规范到4NF。

1）A→→BCD是非平凡的，A不是R的超键，分解R为：
$R_1$ =ABCD，$\Pi_{R1}(F)$={B→AC,C→D,A→→BCD }；
$R_2$ = AEI，　$\Pi_{R2}(F)$={A→→EI }

2）$R_2$属于4NF，因A→→EI是平凡的

3）$R_1$不属于4NF，在$R_1$中，A→→BCD 是平凡的，但函数依赖C→D中的C不是$R_1$的超键。$R_1$还需要继续分解。

4）分解$R_1$为：$R_{11}$ = ABC，　$R_{12}$ = CD。
$R_{11}$、$R_{12}$都是4NF，因B和C分别是$R_{11}$和$R_{12}$的键。

5）分解结果：$\rho$={$R_{11}$，$R_{12}$，$R_2$}

# 7.连接依赖和投影-连接依赖

- **PJNF**
  - 略

# 规范化小结

- 关系数据库的规范化理论是关系数据库逻辑设计的工具。
  - 一个关系只要其分量都是不可分的数据项，它就是规范化的关系，但这只是最基本的规范化。
  - 规范化程度可以有多个不同的级别，从低到高依次为：**1NF ⊃ 2NF ⊃ 3NF⊃ BCNF ⊃ 4NF ⊃ PJNF**
  - 规范化程度过低的关系不一定能够很好地描述现实世界，可能存在插入异常、删除异常、修改复杂、数据冗余等问题。
  - 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高一级范式的关系模式集合（关系模式的规范化）。

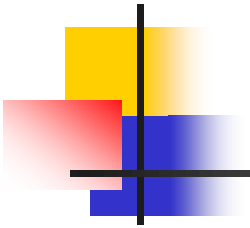
- 关系模式规范化的基本步骤
  - **1NF**
    ↓ 消除非主属性对候选键的部分函数依赖
  - **2NF**
    ↓ 消除非主属性对候选键的传递函数依赖
  - **3NF**
    ↓ 消除主属性对候选键的部分依赖和传递函数依赖
  - **BCNF**
    ↓ 消除非平凡且决定因素不包含候选键的多值依赖
  - **4NF**

- 规范化的基本思想
  - 通过模式分解，消除不合适的数据依赖，使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。
  - 一个关系模式分离程度越高，它所表示的概念就越清楚，结构就越简单，所存在的存储异常就消除得越多。
  - 因此，规范化实质上是概念的单一化。

- 不能说规范化程度越高的关系模式就越好
  - 分离程度越高，连接运算所花费的代价就越大，从而使效率降低；
  - 从**3NF**之后，规范化所得到的模式可能不保持某些数据依赖，因而与原模式不完全等价；
- 在设计中，模式究竟分解到哪一级范式取决于实际应用。
  - 规范化是范式的升高，连接是范式的降低；
  - 关系模式的规范化处理可以终止于任一级别；
  - 只考虑函数依赖，**BCNF**是关系模式规范化的最高程度
  - 如果考虑多值依赖，**4NF**是关系模式规范化的最高程度
  - 一般最常用的是**3NF**和**BCNF**。

# 本章小结

- 关系模型的存储异常
- 函数依赖
- 函数依赖公理
- 模式分解
- 关系模式的规范化
- 多值依赖和**4NF**
- 连接依赖和投影-连接范式**(Project-Join NF)**



- 关系数据库的规范化理论为数据库设计提供了理论上的指南和工具
  - 也仅仅是指南和工具
- 在模式设计中，不是规范化程度越高，所设计出的模式就越好
  - 必须结合应用环境和现实世界的具体情况，合理地选择数据库模式；
  - 一般，数据库模式规范化到**3NF**或**BCNF**就可以了。