第六章 关系数据理论

问题

- **如何才能构造一个良好的关系模式？要回答这个问题就必须要解决以下问题：**
  - 什么是不好的关系模式，一个不好的关系模式存在哪些弊病？
  - 区分一个关系模式设计的优劣程度的标准是什么？
  - 如何将一个不好的关系模式转换为一个好的关系模式？
- **关系数据理论借助于数学工具规定了一套关系数据库设计的理论和方法。是数据库逻辑设计的有力工具。**

关系数据库设计中存在的问题(I)

- 有关学生的关系模式**S(SNO，SNAME，DEPT，HEAD，CNO，G)**

SNO	SNAME	DEPT	HEAD	CNO	G
S01	杨明	D01	李一	C01	90
S02	李婉	D01	李一	C01	87
S01	杨明	D01	李一	C02	92
S03	李婉	D02	王二	C01	95
S04	安然	D02	王二	C02	78
S02	李婉	D01	李一	C03	81
S05	乐天	D03	赵三	C01	82

- 主键?

关系数据库设计中存在的问题(Ⅱ)

❖ 问题

- 插入异常：如果一个系刚成立没有学生，或者有了学生但尚未安排课程，那么就无法将这个系及其负责人的信息插入数据库。
- 删除异常：如果某个系的全部学生都毕业了，则删除该系学生及其选修课程的同时，把这个系及其负责人的信息也丢掉了。
- 数据冗余和更新异常：学生及其所选课程很多，而系主任只有一个，但其却要和学生及其所选课程出现的次数一样多。此外，如果某个系要更换系主任，就必须修改这个系学生所选课程的每个元组，修改其中的系主任信息。若有疏忽，就会造成数据的不一致，从而造成更新异常。

关系数据库设计中存在的问题(Ⅲ)

- 原因：把多个实体型用一个关系模式表示
- 解决之道：分解

DEPT	HEAD
D01	李一
D02	王二
D03	赵三

SNO	CNO	G
S01	C01	90
S02	C01	87
S01	C02	92
S03	C01	95
S04	C02	78
S02	C03	81
S05	C01	82

SNO	SNAME	DEPT
S01	杨明	D01
S02	李婉	D01
S03	李婉	D02
S04	安然	D02
S05	乐天	D03

函数依赖

- 一个实体型的诸属性之间具有内在的联系，通过对这些联系的分析，我们可以做到一个关系模式只表示一个实体型的信息，从而消除上述问题。在关系模型中，我们利用数据依赖来描述这些属性间的联系。
- 数据依赖是通过关系中属性间值的相等与否体现出来的数据间的相互关系，它是现实世界属性间相互联系的抽象，是数据内在的性质，是语义的体现。其中最重要的是函数依赖。

函数依赖

- 函数依赖极为普遍地存在于现实生活中。考察关系模式**S(SNO，SNAME，DEPT，HEAD，CNO，G)**，由于一个**SNO**只对应一个学生，而一个学生只能在一个系中学习。因而当**SNO**的值确定后，**SNAME**和**DEPT**也被唯一地确定了。就像自变量**x**确定后，相应的***f*(x)**也被确定了一样。我们说**SNO**函数决定（**SNAME**，**DEPT**），而（**SNAME**，**DEPT**）函数依赖于**SNO**。

函数依赖

SNO	SNAME	DEPT	HEAD	CNO	G
S01	杨明	D01	李一	C01	90
S02	李婉	D01	李一	C01	87
S01	杨明	D01	李一	C02	92
S03	李婉	D02	王二	C01	95
S04	安然	D02	王二	C02	78
S02	李婉	D01	李一	C03	81
S05	乐天	D03	赵三	C01	82

SELECT SNAME FROM S

WHERE SNO=‘S01’

SELECT SNO FROM S

WHERE SNAME=‘李婉’

SELECT SNAME FROM S

WHERE SNO=‘S02’

函数依赖

❖定义

- 函数依赖：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函数决定Y，或Y函数依赖于X，记作X→Y。

  如SNO → SNAME，（SNO，CNO）→ G

```
SELECT SNO, COUNT(DISTINCT SNAME)
FROM S
GROUP BY SNO
```

函数依赖

- 函数依赖是不随时间而变的。若关系模式**R**具有函数依赖**X→Y**，那么虽然关系模式**R**的关系实例**r**在**X**，**Y**上的取值各不相同，并且随时间而变化， 但**X**，**Y**在任一特定时刻都保持函数依赖**X→Y** 。
- 函数依赖不是指关系模式**R**的某个或某些关系满足的约束条件，而是指**R**的一切关系均要满足的约束条件。
- 函数依赖是语义范畴的概念，它反映了一种语义完整性约束，我们只能根据语义来确定一个函数依赖。

关系数据库设计中存在的问题(I)

SNO	SNAME	DEPT	HEAD	CNO	G
S01	杨明	D01	李一	C01	90
S02	李婉	D01	李一	C01	87
S01	杨明	D01	李一	C02	92
S03	李婉	D02	王二	C01	95
S04	安然	D02	王二	C02	78
S02	李婉	D01	李一	C03	81
S05	乐天	D03	赵三	C01	82

- G→SNO? G→SNAME?
- SNO→SNAME?

函数依赖

- **函数依赖于属性间的联系类型有关。**
  - 当X、Y之间是“1对1”联系时，则存在函数依赖X→Y和Y →X。
    - 如：学号和身份证号
  - 当X、Y之间是“多对1”联系时，则存在函数依赖X→Y。
    - 如：SNO和DEPT
  - 当X、Y之间是“多对多”联系时，则不存在函数依赖。
    - 如：SNO和CNO

函数依赖

- 平凡函数依赖：如果$X \rightarrow Y$，但$Y \not\subseteq X$，则称其为非平凡的函数依赖，否则称为平凡的函数依赖。如（SNO，SNAME）$\rightarrow$ SNAME是平凡的函数依赖
- 完全函数依赖：在R(U)中，如果$X \rightarrow Y$，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，记作$X \xrightarrow{f} Y$，否则称为Y对X部分函数依赖，记作$X \xrightarrow{p} Y$。

（SNO，CNO）$\xrightarrow{f}$ G，

（SNO，CNO）$\xrightarrow{p}$ SNAME

函数依赖

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

$$SNO \rightarrow DEPT，DEPT \rightarrow HEAD$$

函数依赖

A	B	C	D
a1	b1	c1	d1
a1	b2	c1	d2
a2	b2	c2	d2
a2	b3	c2	d3
a3	b3	c2	d4

- 检验：A→C？ C→A？ AB→D？

码

❖ 定义

- 候选码：设K为R< U , F >的属性或属性组合， 若K $\xrightarrow{f}$ U，则称K为R的一个候选码。若候选码多于一个，则选定其中一个作为主码。
- 超码：设K为R< U , F >的属性或属性组，若K→ U，则称K为R的超码。

码

- 主属性：包含在任何一个候选码中的属性，称作主属性。
- 非主属性：不包含在任何一个候选码中的属性，称作非主属性。
- 全码：关系模式的码由整个属性组构成。

如（P，W，A）

例子

关系模式S(SNO，SNAME，DEPT，HEAD，CNO，G)

主码：（SNO，CNO）

函数依赖：

$$(SNO，CNO) \xrightarrow{f} G$$

SNO→SNAME，$(SNO，CNO) \xrightarrow{p} SNAME$

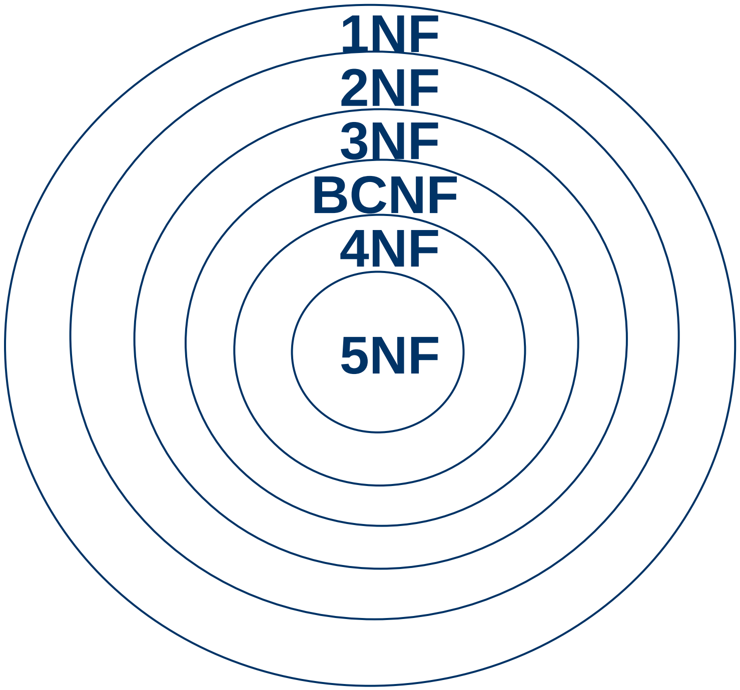
SNO → DEPT，$(SNO，CNO) \xrightarrow{p} DEPT$

SNO→HEAD，$(SNO，CNO) \xrightarrow{p} HEAD$

范式

❖ 定义

- 范式是对关系的不同数据依赖程度的要求。如果一个关系满足某个范式所指定的约束集，则称它属于某个特定的范式。



规范化

- 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级范式的关系模式的集合，这一过程称作规范化。

1NF

❖ 定义

关系中每一分量必须是原子的，不可再分。即不能以集合、序列等作为属性值。

SNO	CNO
S1	{C1，C2，C3}

不满足1NF的关系模式

SNO	CNO
S1	C1
S1	C2
S1	C3

满足1NF的关系模式

2NF(Ⅰ)

- 定义
  - 若R∈1NF，且每个非主属性完全依赖于码，则称R∈2NF
  - 将1NF的关系模式规范化为2NF的关系模式，其方法是消除1NF的关系模式中非主属性对码的部分依赖。
  - 思考：如果关系R的全体属性都是R的主属性，或者R的所有候选码都只含一个属性，那么，R是否属于第二范式？

2NF(II)

关系模式S(SNO, SNAME，DEPT，HEAD，CNO，G)

- **不良特性**
  - 插入异常：如果学生没有选课，关于他的个人信息及所在系的信息就无法插入。
  - 删除异常：如果删除学生的选课信息，则有关他的个人信息及所在系的信息也随之删除了。
  - 数据冗余：如果一个学生选修了k门课，则有关他的所在系的信息重复
  - 更新异常：如果学生转系，若他选修了k门课，则需要修改k次。

2NF(III)

- 原因：**S$\notin$2NF**，因为

（SNO，CNO）$\xrightarrow{f}$ G

SNO$\rightarrow$SNAME,	（SNO，CNO）$\xrightarrow{p}$ SNAME
SNO $\rightarrow$ DEPT,	（SNO，CNO）$\xrightarrow{p}$ DEPT
SNO$\rightarrow$HEAD,	（SNO，CNO）$\xrightarrow{p}$ HEAD

- 规范化

将S分解为：

S_SD(SNO, SNAME, DEPT, HEAD) $\in$2NF

SC(SNO , CNO , G)$\in$2NF

3NF（I）

S_SD(SNO, SNAME, DEPT, HEAD)

❖不良特性

- 插入异常：如果系中没有学生，则有关系的信息就无法插入。
- 删除异常：如果学生全部毕业了，则在删除学生信息的同时有关系的信息也随之删除了。
- 更新异常：如果学生转系，不但要修改DEPT，还要修改HEAD，如果换系主任，则该系每个学生元组都要做相应修改。
- 数据冗余：每个学生都存储了所在系的系主任的信息。

3NF(II)

❖定义

- 关系模式R< U , F >中，若不存在这样的码X，属性组Y及非主属性Z($Z \not\subseteq Y$)，使得下式成立，

$$X \rightarrow Y , Y \rightarrow Z , Y \not\rightarrow X$$

则称$R \in 3NF$。

- 将2NF的关系模式规范化为3NF的关系模式，其方法是消除2NF的关系模式中非主属性对码的传递依赖。

3NF(III)

- 原因： **S_SD $\notin$3NF**，因为
  - SNO→SNAME, SNO→DEPT
  - SNO $\xrightarrow{t}$ HEAD
  - 原因：SNO→DEPT，$\boxed{\text{DEPT→HEAD}}$
- 规范化
  - 将S分解为：
    - DEPT(DEPT , HEAD)
    - STUDENT(DEPT , SNAME , SNO)

BCNF(I)

❖示例

- STJ(S, T, J)，S表示学生，T表示教师，J表示课程。每位老师只教授一门课,每门课由若干教师教，某一学生选定某门课就确定了一个固定的教师，因此具有以下函数依赖：

T→J，（S，J）→T

（S，T），（S，J）为候选码。

BCNF(Ⅱ)

不良特性

- 插入异常：如果没有学生选修某位老师的任课，则该老师担任课程的信息就无法插入。
- 删除异常：删除学生选课信息，会删除掉老师的任课信息。
- 更新异常：如果老师所教授的课程有所改动，则所有选修该老师课程的学生元组都要做改动。
- 数据冗余：每位学生都存储了有关老师所教授的课程的信息。

BCNF(III)

❖定义

- 若关系模式R< U , F >$\in$1NF，若X$\rightarrow$Y，且Y$\nsubseteq$X时，X必含有码，则R< U , F > $\in$BCNF。
- 由BCNF的定义可以看到，每个BCNF的关系模式都具有如下三个性质：
  - 所有非主属性都完全函数依赖于每个候选码。
  - 所有主属性都完全函数依赖于每个不包含它的候选码。
  - 没有任何属性完全函数依赖于非码的任何一组属性。

BCNF(IV)

- **原因：**

  存在主属性对码的不良依赖，STJ $\notin$ BCNF，因为T $\to$ J，而T不含有码。

- **改造**

  将S分解为（S，T），（T，J）。

- **思考**

  (S , J , P)，表示学生选修课程的名次，有函数依赖(S，J)$\to$ P，(J，P) $\to$ S，它属于BCNF吗？

多值依赖(I)

- 关系模式TEACH（C#，P#，B#），一门课程由多个教师任教，一门课程使用相同的一套参考书。它的码是（C#，P#，B#），所以属于BCNF。

C#	P#	B#
物理	{张明, 张平}	{普通物理学, 光学原理}
化学	{张明, 王微}	{无机化学, 有机化学}

C#	P#	B#
物理	张明	普通物理学
物理	张明	光学原理
物理	张平	普通物理学
物理	张平	光学原理
化学	张明	无机化学
化学	张明	有机化学
化学	王微	无机化学
化学	王微	有机化学

多值依赖(Ⅱ)

不良特性

- 插入异常：当某门课程增加一名教师时，该门课程有多少本参考书就必须插入多少个元组；同样当某门课程需要增加一本参考书时，它有多少个教师就必须插入多少个元组。
- 删除异常：当删除一门课程的某个教师或者某本参考书时，需要删除多个元组。
- 数据冗余：同一门课的教师与参考书的信息被反复存储多次。
- 更新异常：当一门课程的教师或参考书作出改变时，需要修改多个元组。

多值依赖(III)

❖定义

- 设R(U)是属性集U上的一个关系模式，X、Y、Z是U的子集，并且Z = U – X – Y，关系模式R(U)中多值依赖X →→ Y成立，当且仅当对R(U)的任一关系r，给定的一对（x，z）值有一组Y的值，这组值仅仅决定于x值而与z值无关。
- 如在关系模式TEACH中，对(物理，普通物理学)有一组P#值(张明, 张平)，对(物理, 光学原理)也有一组P#值(张明, 张平)，这组值仅取决于C#的取值，而与B#的取值无关。因此，P#多值依赖于C#，记作C# →→ P#，同样有C# →→ B#。

多值依赖(IV)

- 形式化：在R(U)的任一关系r中，如果存在元组t，s使得t[x]=s[x]，那么就必然存在元组w，v∈r，（w，v可以与s，t相同），使得：

  w[X] = s[X] = v[X] = t[X]

  w[Y] = t[Y]，v[Y] = s[Y]

  w[Z] = s[Z]，v[Z] = t[Z]

  则称Y多值依赖与X，记作X →→ Y。

  若X →→ Y，而Z＝ϕ，则称X →→ Y为平凡的多值依赖。

多值依赖(VI)

❖ 性质

- 多值依赖具有对称性，即
  若X→→Y，则X→→Z，其中Z=U–X–Y。
- 函数依赖是多值依赖的特例，即
  若X→Y，则X→→Y。
- 若X→→Y， X→→Z，则X→→YZ
- 若X→→Y， X→→Z，则X→→Y∩Z
- 若X→→Y， X→→Z，则X→→Y－Z， X→→ Z－Y
- 若X→→Y， Y→→Z，则X→→ Z － Y

多值依赖 Vs 函数依赖(Ⅰ)

- **函数依赖有效性范围**
  - X→Y的有效性仅决定于X、Y属性集上的值，它在任何属性集W（XY $\subseteq$ W $\subseteq$ U）上都成立。

    若X→Y在R(U)上成立，则对于任何Y'$\subseteq$ Y，均有X→Y '成立。

多值依赖 Vs 函数依赖(Ⅱ)

❖ 多值依赖有效性范围

- $X \to\to Y$的有效性与属性集范围有关。

  $X \to\to Y$在U上成立 $\Rightarrow$ $X \to\to Y$在属性集W（$XY \subseteq W \subseteq U$）上成立。反之则不然。

  若在R(U)上，$X \to\to Y$在属性集W($XY \subseteq W \subseteq U$)上成立，则称$X \to\to Y$为R(U) 的嵌入式多值依赖。

  若$X \to\to Y$在R(U)上成立，则不能断言对于$Y' \subseteq Y$，是否有$X \to\to Y'$成立。

4NF

❖ **定义**

- 关系模式R< U , F > ∈ 1NF，如果对于R到每个非平凡的多值依赖X→→Y（$Y \nsubseteq X$），X都含有码，则称R∈4NF。

❖ 4NF就是限制关系模式的属性之间不允许有非平凡且非函数依赖的多值依赖。因为根据定义，对于每一个非平凡的多值依赖X→→Y，X都含有候选码，于是就有X→Y，所以4NF所允许的非平凡的多值依赖实际上是函数依赖。

4NF

- 如关系模式CPB，C#→→P#，C#→→B#，码为(C#, P#, B#)，所以CPB∉4NF。

  如果一门课Ci有m个教师，n本参考书，则关系中分量为Ci的元组共有m×n个，数据冗余非常大。

❖ 改造

将CPB分解为CP（C#，P#），CB（C#，B#）。

数据依赖的公理系统

- **定义**
  - 对于满足一组函数依赖F的关系模式R< U, F >，其任何一个关系r，若函数依赖X→Y都成立，则称F逻辑蕴涵X→Y。
- **Armstrong公理**

  X，Y，Z是属性集，
  - 自反律。若Y⊆X⊆U， 则X→Y为F所蕴含。
  - 增广律。若X→Y为F所蕴含， 且Z⊆U，则XZ → YZ为F所蕴含。
  - 传递律。若X→Y及Y→Z为F所蕴含，则X→Z为F所蕴含。

数据依赖的公理系统

- **在关系模式R< U, F >中，为F所逻辑蕴涵的函数依赖的全体称作F的闭包，记作F^+。**

  例：F={X→Y，Y→Z}，则{X→Z} ⊆ F^+。

- **Armstrong公理的有效性及完备性**
  - 有效性：由F出发根据Armstrong公理推导出来的函数依赖一定在F^+中。
  - 完备性：F^+中的每一个函数依赖都可以由F出发根据Armstrong公理从F中导出。

数据依赖的公理系统

❖ 关于Armstrong公理有效性的证明

- 设$Y \subseteq X \subseteq U$，R<U, F>上的任一关系r的两个元组t，s

  若t[X]=s[X]，由$Y \subseteq X$，有t[Y]=s[Y]，所以$X \to Y$成立，自反律得证。

- 设$X \to Y$为F所蕴含，且$Z \subseteq U$。设R<U, F>上的任一关系r的两个元组t，s

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

  由$X \to Y$，有t[Y]=s[Y]，所以t[YZ]=s[YZ]，因此$XZ \to YZ$为F所蕴含，增广律得证。

数据依赖的公理系统

- 设X→Y及Y→Z为F所蕴含。设R<U, F>上的任一关系r的两个元组t，s 。

若t[X]=s[X]，且由X→Y，有t[Y]=s[Y]

再由Y→Z，有t[Z]=s[Z]，所以X→Z为F所蕴含，传递律得证。

由Armstrong公理导出的推理规则

- 合并律。若X → Y，X → Z，则X → YZ。
- 分解律。若X → YZ，则X → Y，X → Z 。
- 伪传递律。若X → Y，WY → Z，则XW → Z

数据依赖的公理系统

❖示例

$R<U, F>$, $U = (A, B, C, G, H, I)$, $F = \{A \to B, A \to C, CG \to H, CG \to I, B \to H\}$,

- $A \to H$?
- $CG \to HI$?
- $AG \to I$?

数据依赖的公理系统

问题：有没有一般性的算法判定X→Y是否能由F根据Armstrong公理导出？

- **属性集的闭包**
  - 设F为属性集U上的一组函数依赖，X $\subseteq$ U，

    X_F^+ = {A | X→A能由F根据Armstrong公理导出}

    称 X_F^+ 为属性集X关于函数依赖集F的闭包。

数据依赖的公理系统

- 引理一：

  $X \to A_1 A_2 \ldots A_k$成立 $\Leftrightarrow$ $X \to A_i$成立$(i=1, 2, \ldots, k)$

- 引理二：

  设F为属性集U上的一组函数依赖，X，Y $\subseteq$U，$X \to Y$能由F根据Armstrong公理导出的充分必要条件是$Y \subseteq X_F^+$

数据依赖的公理系统

- **算法（求属性集的闭包）**

  Input：X，F

  Output：X_F^+

  步骤：

(1)令$X^{(0)}$＝X，i=0

(2)求B，B=$\{A|(\exists V)(\exists W)(V\rightarrow W \wedge V \subseteq X^{(i)} \wedge A \in W)\}$

(3) $X^{(i+1)} = B \cup X^{(i)}$

(4)判断$X^{(i+1)} = X^{(i)}$？

(5)若相等，或$X^{(i+1)}$＝U，则$X^{(i+1)}$就是X_F^+，算法终止。

(6)若否，则i=i+1，返回第二步。

数据依赖的公理系统

❖示例

R< U, F >, U = (A, B, C, D, E), F = {AB→C, B→D, C→E, CE→B, AC→B},计算 $(AB)_F^+$ 。

所用依赖	$(AB)_F^+$
AB→C	ABC
B→D	ABCD
C→E	ABCDE

$(AB)_F^+$ = ABCDE

数据依赖的公理系统

Armstrong公理完备性的证明

证明：用反证法，

假定存在函数依赖X→Y被F逻辑蕴涵，但X→Y不能用Armstrong公理从F中导出。

(1) 若V→W∈F，且V⊆ X_F^+，则W ⊆ X_F^+

证：因为V ⊆ X_F^+，所以有X→V；于是X →W成立，所以W ⊆ X_F^+。

数据依赖的公理系统

（2）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F中的全部函数依赖在r上成立。

r	X_F^+	$U - X_F^+$
t	11…1	00 …0
s	11…1	11…1

若r不是R(U, F)的关系，则必由于F中有函数依赖V→W在r上不成立所致。由r的构成可知，V必定是X_F^+的子集，而W不是X_F^+的子集，可是由第（1）步可知，W$\subseteq X_F^+$矛盾，所以r必是R(U, F)的一个关系。

数据依赖的公理系统

（3）若X→Y不能由F从Armstrong公理导出，则Y不是 X_F^+ 的子集，因此必有Y的子集Y'满足Y'⊆U− X_F^+ ，则X→Y在r中不成立，即X→Y必不为R(U, F)所蕴含。

数据依赖的公理系统

❖定义

- 函数依赖集的等价性
  - 函数依赖集F，G，若$F^+ = G^+$，则称F与G等价。
  - $F^+ = G^+ \Leftrightarrow F \subseteq G^+$，$G \subseteq F^+$
- 最小依赖集（最小覆盖）

  满足下列条件的函数依赖集F称为最小覆盖，记作F_m：
  - F中任一函数依赖的右部仅含有一个属性。
  - F中不存在这样的函数依赖$X \rightarrow A$，使得F与$F - \{X \rightarrow A\}$等价。
  - F中不存在这样的函数依赖$X \rightarrow A$，在X中有真子集Z，使得F与$F - \{X \rightarrow A\} \cup \{Z \rightarrow A\}$等价。

数据依赖的公理系统

❖ 算法—求解函数依赖集F的最小覆盖F_m

- 逐个检查F中各函数依赖FD_i：$X \to Y$，

  若$Y=A_1 A_2 \ldots A_k$，$k \geq 2$，则用诸$X \to A_i$代替Y。

- 逐个检查F中各函数依赖$X \to A$，

  令$G = F - \{X \to A\}$，若$A \in X_G^+$，则从F中去掉该函数依赖。

- 逐个检查F中各函数依赖$X \to A$，

  设$X = B_1 \ldots B_m$，逐个考查B_i，

  若$A \in (X - B_i)_F^+$，则以（$X - B_i$）取代X。

数据依赖的公理系统

❖**示例**

$F = \{A \to B，B \to A，A \to C，B \to C，C \to A\}$，求$F_m$。

$F_m = \{A \to B，C \to A，B \to C\}$

或者

$F_m = \{A \to B，B \to A，A \to C，C \to A\}$

模式分解中的问题

❖ **实例**

设R是一个关系模式，R的属性集合U={SNO，DEPT，HEAD），R的函数依赖集合F={SNO→DEPT，DEPT→HEAD}，由于R中存在传递函数依赖，因此存在异常，需要进行模式分解，其方式可以有：

分解一：（SNO），（DEPT），（HEAD）

分解二：（SNO，DEPT），（SNO，HEAD）

分解三：（SNO，DEPT），（DEPT，HEAD）

模式分解中的问题

- 分解一所形成的三个关系无法通过自然连接恢复成原来的关系，实际上是它们三个的笛卡尔积，元组增加了，信息却丢失了。

SNO	DEPT	HEAD
S1	D1	张红
S2	D1	张红
S3	D2	李微
S4	D3	王力

SNO
S1
S2
S3
S4

DEPT
D1
D2
D3

HEAD
张红
李微
王力

模式分解中的问题

- 分解二能够 通过自然连接恢复到原来的关系，但仍然具有插入和删除等异常。原因在于丢失了函数依赖**DEPT**$\rightarrow$**HEAD**。

SNO	DEPT	HEAD
S1	D1	张红
S2	D1	张红
S3	D2	李微
S4	D3	王力

SNO	DEPT
S1	D1
S2	D1
S3	D2
S4	D3

SNO	HEAD
S1	张红
S2	张红
S3	李微
S4	王力

模式分解中的问题

- 分解三既可以通过自然连接恢复到原来的关系，同时也保持了原关系的函数依赖，消除了原关系的异常。

SNO	DEPT	HEAD
S1	D1	张红
S2	D1	张红
S3	D2	李微
S4	D3	王力

SNO	DEPT
S1	D1
S2	D1
S3	D2
S4	D3

DEPT	HEAD
D1	张红
D2	李微
D3	王力

关系模式的分解算法

- **分解的目标**
  - 无损连接分解
  - 保持函数依赖
  - 达到更高级范式

关系模式的分解算法

❖模式分解

- 函数依赖集合$F_i = \{X \to Y \mid X \to Y \in F^+ \wedge XY \subseteq U_i\}$称为F在$U_i$上的投影
- 关系模式R<U , F>的一个分解是指

  $\rho = \{R_1<U_1 , F_1> , R_2<U_2 , F_2>, \ldots , R_n<U_n , F_n>\}$

  其中$U = \bigcup_{i=1}^{n} U_i$，并且没有$U_i \subseteq U_j$，$1 \le i, j \le n$，$F_i$是F在$U_i$上的投影。
- 设$\rho = \{R_1<U_1 , F_1> , R_2<U_2 , F_2>, \ldots , R_n<U_n, F_n>\}$是R(U，F)的一个分解，r是R(U，F)的一个关系，定义$m_\rho(r) = \bowtie_{i=1}^{k} \Pi_{R_i}(r)$，即$m_\rho(r)$是r在ρ中各关系模式投影上的连接。这里$\Pi_{R_i}(r)=\{t.U_i \mid t \in r\}$

关系模式的分解算法

- 一般定义

  关系模式R<U , F>， $U = \bigcup_{i=1}^{n} U_i$,
  ρ = {R_1<U_1 , F_1> , R_2<U_2 , F_2>, … , R_n<U_n , F_n>}是R<U , F>的一个分解，r是R<U , F>的一个关系。定义$m_\rho(r) = \bowtie_{i=1}^{n} \prod_{R_i}(r)$,若对于R<U , F>的任一个关系r，都有r = $m_\rho(r)$，则称ρ是R<U , F>的一个无损连接分解。

关系模式的分解算法

- 算法：（判别一个分解的无损连接性）

  $U=\{A_1, A_2, \ldots, A_n\}$，$F=\{FD_1, FD_2, , FD_\rho\}$不妨设F是一个最小依赖集，记$FD_i$为$X_i \to A_j$

  $\rho = \{R_1<U_1, F_1>, R_2<U_2, F_2>, \ldots, R_k<U_k, F_k>\}$

  1.建立一个n列k行的矩阵。每一列对应一个属性，每一行对应于分解中的一个关系模式。若属性A_j属于U_i，则在j列i行交叉处填上a_j，否则填上b_{ij}。

  $$\{C_{ij} \mid 若A_j \in U_i, C_{ij} = a_j, 否则C_{ij} = b_{ij}\}$$

关系模式的分解算法

2.对每一个FD_i做下列操作：找到X_i所对应的列中具有相同符号的那些行。考察这些行中j列的元素，若其中有a_j，则全部改为a_j；否则全部改为b_{mj}；m是这些行的行号最小值。

如果在某次更改之后，有一行成为$a_1, a_2, \ldots, a_n$。则算法终止，ρ为无损分解，否则为有损分解。

关系模式的分解算法

3. 比较扫描前后，表有无变化。如有变化，则返回第二步，否则算法终止。

关系模式的分解算法

- 示例：U={A,B,C,D,E}, F={AB→C, C→D,D→E}

ρ ={(A, B, C), (C, D), (D, E)}

	A	B	C	D	E
ABC	a_1	a_2	a_3	b_{14}	b_{15}
CD	b_{21}	b_{22}	a_3	a_4	b_{25}
DE	b_{31}	b_{32}	b_{33}	a_4	a_5

AB→C

	A	B	C	D	E
ABC	a_1	a_2	a_3	b_{14}	b_{15}
CD	b_{21}	b_{22}	a_3	a_4	b_{25}
DE	b_{31}	b_{32}	b_{33}	a_4	a_5

C→D

	A	B	C	D	E
ABC	a_1	a_2	a_3	a_4	b_{15}
CD	b_{21}	b_{22}	a_3	a_4	b_{25}
DE	b_{31}	b_{32}	b_{33}	a_4	a_5

D→E

	A	B	C	D	E
ABC	a_1	a_2	a_3	a_4	a_5
CD	b_{21}	b_{22}	a_3	a_4	a_5
DE	b_{31}	b_{32}	b_{33}	a_4	a_5

关系模式的分解算法

- $U=\{A,B,C,D,E\}$,
- $F=\{A \rightarrow C,\ B \rightarrow C,\ C \rightarrow D,\ DE \rightarrow C,\ CE \rightarrow A\}$
- $\rho=\{R_1(A, D), R_2(A, B), R_3(B, E), R_4(C, D, E), R_5(A, E)\}$

关系模式的分解算法

	A	B	C	D	E
$R_1(A, D)$	a1	b12	b13	a4	b15
$R_2(A, B)$	a1	a2	b23	b24	b25
$R_3(B, E)$	b31	a2	b33	b34	a5
$R_4(C, D, E)$	b41	b42	a3	a4	a5
$R_5(A, E)$	a1	b52	b53	b54	a5

关系模式的分解算法

- **A→C**

	A	B	C	D	E
$R_1(A, D)$	a1	b12	b13	a4	b15
$R_2(A, B)$	a1	a2	b13	b24	b25
$R_3(B, E)$	b31	a2	b33	b34	a5
$R_4(C, D, E)$	b41	b42	a3	a4	a5
$R_5(A, E)$	a1	b52	b13	b54	a5

关系模式的分解算法

- $B \rightarrow C$

	A	B	C	D	E
$R_1(A, D)$	a1	b12	b13	a4	b15
$R_2(A, B)$	a1	a2	b13	b24	b25
$R_3(B, E)$	b31	a2	b13	b34	a5
$R_4(C, D, E)$	b41	b42	a3	a4	a5
$R_5(A, E)$	a1	b52	b13	b54	a5

关系模式的分解算法

- $C \rightarrow D$

	A	B	C	D	E
$R_1(A, D)$	a1	b12	b13	a4	b15
$R_2(A, B)$	a1	a2	b13	a4	b25
$R_3(B, E)$	b31	a2	b13	a4	a5
$R_4(C, D, E)$	b41	b42	a3	a4	a5
$R_5(A, E)$	a1	b52	b13	a4	a5

关系模式的分解算法

- **DE→C**

	A	B	C	D	E
$R_1(A, D)$	a1	b12	a3	a4	b15
$R_2(A, B)$	a1	a2	a3	a4	b25
$R_3(B, E)$	b31	a2	a3	a4	a5
$R_4(C, D, E)$	b41	b42	a3	a4	a5
$R_5(A, E)$	a1	b52	a3	a4	a5

关系模式的分解算法

- $CE \rightarrow A$

	A	B	C	D	E
$R_1(A, D)$	a1	b12	a3	a4	b15
$R_2(A, B)$	a1	a2	a3	a4	b25
$R_3(B, E)$	a1	a2	a3	a4	a5
$R_4(C, D, E)$	a1	b42	a3	a4	a5
$R_5(A, E)$	a1	b52	a3	a4	a5

关系模式的分解算法

- 定理

  若$U_1 \cap U_2 \to U_1$(或U_2)，则$r = \prod_{U_1}(r) \bowtie \prod_{U_2}(r)$。

关系模式的分解算法

保持函数依赖的分解

- 定义

若$F^+ = (\bigcup_{i=1}^{n} F_i)^+$，则称$R< U , F >$的分解$\rho = \{R_1<U_1 , F_1> , \ldots , R_n<U_n , F_n>\}$保持函数依赖。

如表（职工，级别，工资）的分解，

分解一：（职工，工资），（工资，级别）

丢失函数依赖，职工→级别

分解二：（职工，级别），（工资，级别）

保持函数依赖。

关系模式的分解算法

❖ **算法：（达到3NF且保持函数依赖的分解）**

1.对R<U，F> 中的函数依赖集F进行“极小化处理”（处理后得到的依赖集仍记为F）。

2.找出不在F中出现的属性，将它们构成一个关系模式，并从U中去掉它们(剩余属性仍记为U)。

3.若有$X \rightarrow A \in F$，且XA=U，

则ρ ={R}，算法终止。

关系模式的分解算法

4.否则，对F按具有相同左部的原则进行分组（设为k组），每一组函数依赖F_i'所涉及的全部属性为U_i。若$U_i \subseteq U_j$（i≠j），就去掉U_i。由于经过了步骤2，故$U = \bigcup_{i=1}^{n} U_i$，于是$\rho = \{R_1<U_1, F_1>, \ldots, R_k<U_k, F_k>\}$是$R<U, F>$的一个保持函数依赖的分解，并且每个$R_i<U_i, F_i> \in 3NF$。

关系模式的分解算法

- 示例

U={SNO，DEPT，HEAD，CNO，G}

F={SNO→DEPT，SNO→HEAD，DEPT→HEAD，(SNO,CNO)→G}

1.F_C={SNO→DEPT，DEPT→HEAD，(SNO,CNO)→G}

2.分组

{(SNO，DEPT)，SNO→DEPT}

{(DEPT，HEAD)，DEPT→HEAD}

{(SNO，CNO，G)，(SNO,CNO)→G}

关系模式的分解算法

- 示例：R（ABC；A→C，B→C）

  1.按保持无损连接分解

  码为AB，分解为{AC；A→C}，{AB}。

  丢失了函数依赖B→C。

  2.按保持函数依赖分解

  进行分组，{AC；A→C}，{BC；B→C}。

  分解是有损的。

A	B	C
1	1	1
2	1	1
2	2	1

A	C
1	1
2	1

B	C
1	1
2	1

A	B	C
1	1	1
1	2	1
2	1	1
2	2	1

关系模式的分解算法

- 算法：（达到3NF且同时保持无损连接与函数依赖的分解）

设X为R<U，F>的码，设$\rho = \{R_1<U_1, F_1>, \dots, R_k<U_k, F_k>\}$是R<U，F>的一个保持函数依赖的3NF分解。令$\tau = \rho \cup \{R^* <X, F_X>\}$

若有某个U_i，$X \subseteq U_i$，则将$R^* <X, F_X>$从τ去掉。

$\tau = \rho \cup R^* <X, F_X>$即为所求的解。

关系模式的分解算法

- 示例：

求R（ABC；A→C，B→C）的保持无损连接和函数依赖的3NF分解。

1.按保持函数依赖分解

进行分组， ρ={{AC；A→C}，{BC；B→C}}。

2.键为AB

τ = ρ∪{AB}

最后的分解为：

{{AC；A→C}，{BC；B→C}，{AB}}

关系模式的分解算法

❖ **算法：（达到BCNF无损连接分解算法）**

给定关系模式$R<U , F>$，

1.令$\rho = \{R<U , F>\}$

2.检查ρ中各关系模式是否属于BCNF，若是，则算法终止。

3.设ρ 中$R_i<U_i , F_i>$不属于BCNF，

则存在函数依赖$X \rightarrow A \in F_i^+$ ，且X不是R_i的码，

则XA是R_i的真子集，将R_i分解为$\sigma=\{S_1, S_1\}$，

其中$U_{S_1} = XA$，$U_{S_2} = U_i - \{A\}$

以σ代替R_i，返回到 2.

关系模式的分解算法

- 示例：U={SNO，DEPT，HEAD，CNO，G}

  Fc={SNO→DEPT，DEPT→HEAD，(SNO,CNO)→G}

1. U_1={DEPT，HEAD} , F_1={DEPT→HEAD}

   U_2={SNO, DEPT, CNO, G}, F_2={SNO→DEPT, (SNO,CNO)→G}

2. U_1 = {DEPT，HEAD}, F_1={DEPT→HEAD}

   U_2 = {SNO, DEPT}, F_2={SNO→DEPT}

   U_3 = {SNO, CNO, G}, F_3 = {(SNO,CNO)→G}

例子

- **设有关系R（U, F）**
  - U={A, B, C, D, E, F, G}
  - F={E→D, C→B, CE→AF, B→A, C→AB}
- **求：**
  - R的候选码
  - 判断R所属的范式
  - 如果R不属于第三范式，将R规范化到第三范式，并保持函数依赖和无损连接的分解

例子—求R的候选码

- **L: E, C**
- **R: A, D, F**
- **LR: B**
- **N: G**
- 首先查看 $(L \cup N)_F^+ = (CEG)_F^+ = \{ABCEFG\}$
- 因此，**CEG**是**R**唯一的候选码

例子一判断R属于第几范式

- **R的主属性：C, E, G**
- **R的非主属性：A, B, D, F**
- **非主属性与候选码的关系：**
  - C→D, 故CEG→D为部分函数依赖
- **故R属于第一范式**

例子一分解为第三范式

- 求最小函数依赖集
- 第一步：将函数依赖的右部分解为单属性
- Fm＝{E→D, C→B, CE→A, CE→F, B→A, C→A, C→B}
- 第二步：去掉多余的函数依赖
- Fm＝{E→D, C→B, CE→F, B→A}
- 第三步：检查是否存在部分函数依赖

例子一分解为第三范式

- 按相同左部分组，得到以下**4**个属性组：
  - $R_1(E, D)$，$F=\{E \rightarrow D\}$
  - $R_2(C, B)$，$F=\{C \rightarrow B\}$
  - $R_3(C, E, F)$，$F=\{CE \rightarrow F\}$
  - $R_4(B, A)$，$F=\{B \rightarrow A\}$
- 这些属性组之间没有相互包含的情况，因此，我们得到了保持函数依赖且达到第三范式的分解结果

例子一分解为第三范式

- 判断上述结果是否是无损连接分解
- 因为上述分解后的关系模式中没有哪一个包含了原关系模式**R**的候选码**CEG**，所以上述分解结果是有损的。
- 为此，在上述分解结果中加入一个全由候选码中属性构成的关系模式
  - $R_5(C, E, G)$，$F=\varphi$

例子一分解为第三范式

- 最后的分解结果为：
  - $R_1(E, D)$，$F=\{E \rightarrow D\}$
  - $R_2(C, B)$，$F=\{C \rightarrow B\}$
  - $R_3(C, E, F)$，$F=\{CE \rightarrow F\}$
  - $R_4(B, A)$，$F=\{B \rightarrow A\}$
  - $R_5(C, E, G)$，$F=\varphi$