南京大学 NANJING UNIVERSITY

奋进行动 团结奋斗 争先进位 坐言起行 应势而动

# 数据管理基础

## 第6章 关系数据理论

（复习总结）

智能软件与工程学院

# 关系数据理论

- 在关系数据库系统的建立过程中，如何设计出一个合适的关系数据库模式?
  ① 如何评价关系模式设计的好坏?
  ② 如何设计性能良好的关系模式?

- 关系数据库的规范化理论，就是为了解决上述两个问题而提出的关系数据库设计理论

- 学习目的
  - 了解不好的关系模式设计所存在的问题
  - 理解数据依赖（包括函数依赖和多值依赖）及其相关概念
  - 理解关系规范化理论及其相关算法
  - 关系规范化理论的应用与实践

# 关系数据理论

- 将分以下六个主题来介绍本章的内容
  - 问题的提出
  - 函数依赖 与 码
  - 范式 与 规范化
  - Armstrong公理系统
  - 模式的分解
  - 多值依赖 与 4NF

南京大学
NANJING UNIVERSITY

奋进行动 团结奋斗 争先进位 坐言起行 应势而动

# 数据管理基础

## 第6章 关系数据理论

## （问题的提出）

智能软件与工程学院

# 关系模式 与 第一范式（1NF）

- 关系模式由五部分组成，是一个五元组：$R(U,D,DOM,F)$
  - **R**：关系名
  - **U**：组成该关系的属性名集合
  - **D**：**U** 中属性所来自的域
  - **DOM**：属性到域的映射集合（描述各个属性对应的域）
  - **F**：属性组**U**上的一组数据依赖（关系上的完整性约束条件）
  - 关系名**R**是符号化的元组语义

- 关系的性质
  ① 列是同质的
  - 每一列的分量来自同一个域

  ② 不同的列可出自同一个域
  ③ 列的无序性（属性的无序性）
  ④ 行的唯一性（元组的唯一性）
  ⑤ 行的无序性（元组的无序性）
  ⑥ 分量必须取原子值
  - 每个分量都是不可分的数据项

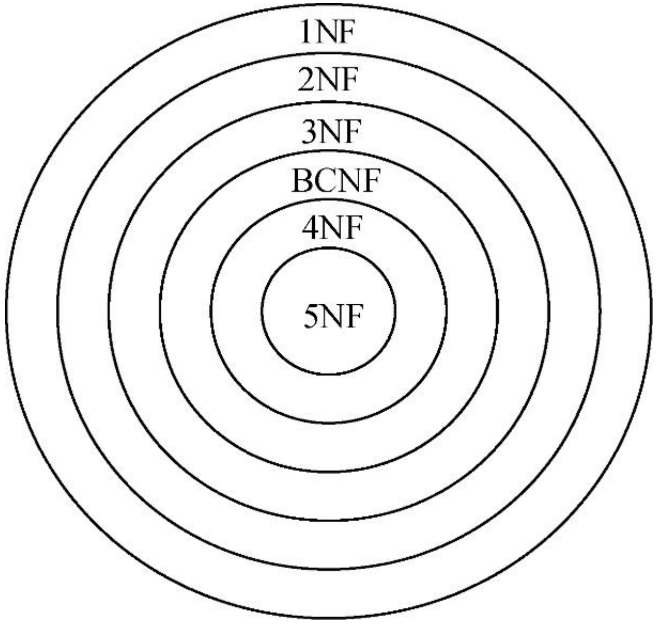
- 由于*D*、DOM与模式设计关系不大，因此可把关系模式看作一个三元组 $R<U,F>$
  - 作为关系的性质：每个分量都是不可分的数据项
  - 满足了上述六条性质的关系模式就属于**第一范式（1NF）**

# 关系规范化理论

- 关系模式设计
  - 同一个关系数据库系统可以有多种不同的关系模式设计方案
- 不同模式设计方案的比较
  - 不同的模式设计方案有好坏之分
  - 好的设计方案应该是：既具有合理的数据冗余度，又没有插入和删除等操作异常现象
- 在不同的设计结果之间产生区别的原因
  - 在一个关系中，属性与属性之间存在语义相关性（数据依赖）
  - 不同的设计方案，可能会产生不同的数据依赖
- 关系的规范化
  - 按照给定范式的要求来设计关系模式
  - 范式(Normal Form)：对一个关系中允许存在的函数依赖的要求

## 各种范式之间的联系

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




- 如果一个关系模式满足高级别范式对数据依赖的要求，那么它肯定也满足低级别范式对函数依赖的要求；
- 一个关系模式满足低级别范式对数据依赖的要求，但它不一定能满足高级别范式对函数依赖的要求。
- 如果某一关系模式**R**满足范式**n**的要求，则称关系模式**R**为第**n**范式，可简记为 $R \in nNF$
- 一个低一级范式的关系模式，通过模式分解（**schema decomposition**）可以转换为若干个高一级范式的关系模式的集合，这种过程就叫 规范化（**normalization**）。

南京大学
NANJING UNIVERSITY

奋进行动 团结奋斗 争先进位 坐言起行 应势而动

# 数 据 管 理 基 础

## 第6章 关系数据理论

## （函数依赖 与 码）

智能软件与工程学院

# 函数依赖 & 码

- 定义6.1 函数依赖
  - 函数依赖的定义
  - 非平凡函数依赖，平凡函数依赖
- 定义6.2 完全函数依赖，部分函数依赖
- 定义6.3 传递函数依赖（直接函数依赖）

- 定义6.4 候选码
  - 候选码、超码
  - 全码
  - 主属性，非主属性/非码属性

- 定义6.5 外码

# 常用的符号表示方法

| | |
|---|---|
| $A, B, C$ | 属性名 |
| $ABC$ | 代表由$A$、$B$、$C$三个属性组成的属性集合，即：$ABC = \{A, B, C\}$<br>有时候也用(...)代替其中的{...}，即表示为：$ABC\ =\ (A, B, C)$ |
| $X, Y, Z$ | 关系的属性子集(subset) |
| $XY$ | 表示$X$和$Y$的并集，即：$XY\ =\ X$ union $Y$ |
| $R, S, T$ | 关系名 (关系模式) |
| $R(U)$ | 关系模式：关系名$R$，关系中的属性集合$U$ |
| $R(U, F)$ | 关系模式：关系名$R$，属性集合$U$，函数依赖集$F$ |
| $r, s, t$ | 关系实例 (一个关系中的元组集合) |
| $r_1, r_2, r_3$ | 关系中的元组 |
| $r_1[A]$ | 元组 $r_1$ 在属性(集) $A$ 上的取值 |

# 函数依赖 1

- 定义6.1 设*R*(*U*)是一个属性集*U*上的关系模式，*X*和*Y*是*U*的子集。若对于*R*(*U*)的任意一个可能的关系*r*，*r* 中不可能存在两个元组在*X*上的属性值相等，而在*Y*上的属性值不等，则称“***X*函数确定*Y***”或“***Y*函数依赖于*X***”，记作$X \to Y$。

- 如果存在函数依赖 $X \to Y$
  - **X** 称为这个函数依赖的决定因素（**Determinant**）
  - **Y** 称为这个函数依赖的依赖因素

- 若X→Y，并且Y→X, 则记为 $X \longleftrightarrow Y$
- 若Y不函数依赖于X, 则记为 $X \nrightarrow Y$

# 平凡函数依赖 与 非平凡函数依赖

- 在关系模式***R*(*U*)**中，对于***U***的子集***X***和***Y***,
  - $X \to Y$ 但 $Y \not\subseteq X$，则称 $X \to Y$ 是'**非平凡函数依赖**'
  - $X \to Y$ 但 $Y \subseteq X$，则称 $X \to Y$ 是'平凡函数依赖'

- 对于任一关系模式，平凡函数依赖都是必然成立的，它不反映新的语义。因此若不特别声明，我们总是讨论非平凡函数依赖。

- 例：在关系**SC(Sno, Cno, Grade)**中,
  - 有非平凡函数依赖：**(Sno, Cno) → Grade**
  - 有平凡函数依赖：**(Sno, Cno) → Sno**
    **(Sno, Cno) → Cno**

# 完全函数依赖 与 部分函数依赖

- **定义6.2** 在$R(U)$中，如果$X \rightarrow Y$，并且对于$X$的任何一个真子集$X'$，都有 $X' \nrightarrow Y$，则称$Y$对$X$完全函数依赖，记作 $X \xrightarrow{F} Y$

- 若$X \rightarrow Y$，但$Y$不完全函数依赖于$X$，则称$Y$对$X$部分函数依赖，记作 $X \xrightarrow{P} Y$

- 由于：$Sno \nrightarrow Grade$，$Cno \nrightarrow Grade$，$Sno \rightarrow Sdept$

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

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

# 传递函数依赖 1

- **定义6.3** 在$R(U)$中，如果$X \rightarrow Y(Y \nsubseteq X)$，$Y \nrightarrow X$，$Y \rightarrow Z$，$Z \nsubseteq Y$, 则称$Z$对$X$**传递函数依赖**(**transitive functional dependency**)。记为：

$$X \xrightarrow{\text{传递}} Z$$

  - 在定义里加上条件$Y \nrightarrow X$, 是因为：如果 $X \rightarrow Y$且$Y \rightarrow X$ 即 $X \longleftrightarrow Y$, 则$Z$直接函数依赖于$X$，而不是传递函数依赖。
  - 在定义里加上条件$Y \nsubseteq X$ 和 $Z \nsubseteq Y$, 是因为：如果 $Y \subseteq X$ 或 $ZZ \subseteq Y$, 则 $X \rightarrow Z$ 就是一个直接函数依赖，并且是一个部分函数依赖或平凡函数依赖。
  - 一般情况下，传递函数依赖与直接函数依赖在表示上仍然使用函数依赖的表示方法，并没有区分。只有在规范化设计中需要时，我们才区分它们。

# 码 & 超码 1

- 定义6.4 设$K$为$R(U,F)$中的属性或属性组合。若 $K \xrightarrow{F} U$，则$K$称为$R$的一个候选码(Candidate Key)，简称‘码’。
  - 如果$K \rightarrow U$（可能是部分函数依赖，也可能是完全函数依赖）,则$K$称为$R$的一个超码 （Superkey）。
    - 候选码是最小的超码，候选码的真子集一定不是超码
    - 候选码的超集是超码
  - 如果关系$R$的所有属性$U$是$R$的码，称为全码（all-key）
  - 若关系模式***R***有多个候选码，则选定其中的一个做为主码(Primary key)。

# 主属性 & 非主属性

- 主属性 与 非主属性

  - 包含在任何一个候选码中的属性，称为**主属性**（Prime attribute）

  - 不包含在任何码中的属性称为**非主属性**（Nonprime attribute）或**非码属性**（Non-key attribute）

*

# 外码

- **定义6.5 关系模式*R*中属性或属性组*X*并非*R*的码，但*X*是另一个关系模式的码，则称 *X*是*R* 的外部码（Foreign key）也称外码。**
  - **SC(<u>Sno,Cno</u>,Grade)中，Sno不是码，Sno是 S(<u>Sno</u>,Sdept,Sage)的码，则Sno是SC的外码**
- **主码与外部码一起提供了表示关系间联系的手段**

*

# 函数依赖的发现

- 函数依赖的存在性判定
- 如何发现一个关系上的函数依赖?
- 函数依赖发现的例子

# 函数依赖

- 假设在关系模式R(U)上存在函数依赖：$X \rightarrow Y$
- r 是依据关系模式R(U)建立起来的任意一个关系，那么关系 r 必满足:
  - 从关系 r 中任取两个元组 $r_1$和 $r_2$，如果元组 $r_1$在X这组属性上的取值 $r_1[X]$ 等于元组 $r_2$在X这组属性上的取值 $r_2[X]$，即：$r_1[X]=r_2[X]$
  - 则它们在Y属性组上的取值也必定相等，即：$r_1[Y] = r_2[Y]$

$X \rightarrow Y$ <u>if and only if</u> For any rows $r_1$ and $r_2$ in any instance of relation R, if $r_1[X] = r_2[X]$ then $r_1[Y] = r_2[Y]$.

# 函数依赖的发现

- 如何发现一个关系中的函数依赖?
  - 对关系中各个属性的语义进行分析，寻找它们相互之间存在的语义联系，并据此发现在它们之间存在的函数依赖;
  - 除了上述方法外，也可以分析属性与属性之间在取值对应关系上的数量特征，进而判断它们之间是否存在某种函数依赖。
- 属性之间的取值对应关系的数量特征有三种:
  ① '一 对 一' (1:1)
  ② '一 对 多' (1:n) 或者称 '多 对 一' (n:1)
  ③ '多 对 多' (m:n)
- 说明:
  - '数量特征' 只有 '一' 和 '多' 这两种类型，一般用数字 1 代表 '一'，用一个英文字母代表 '多';
  - '一对多'要确定方向，标注清楚哪一个是'一'，哪一个是'多'。

# 函数依赖的发现

- 设X、Y是关系R的两个属性(集)

1) 一对一 (1:1)：如果“一个X值，最多只能有唯一一个Y值与之对应；一个Y值，也最多只能有唯一一个X值与之对应”，则称“X与Y是‘一对一’”。

2) 一对多 (1:n)：如果“一个X值，允许有多个Y值与之对应；一个Y值，却最多只能有唯一一个X值与之对应”，则称“从X到Y是‘一对多’”（或者称“从Y到X是 ‘多对一’”）。

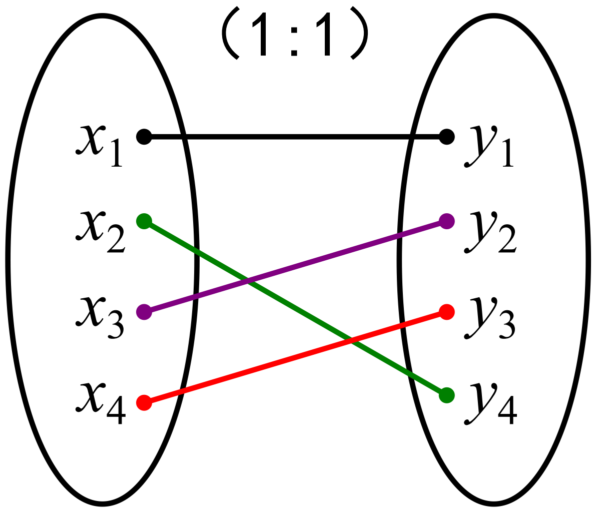
3) 多对多 (m:n)：如果“一个X值，允许有多个Y值与之对应；一个Y值，也允许有多个X值与之对应”，则称“X与Y是‘多对多’”。

- 说明：
  - 在‘多对多’中允许存在‘多对一’现象，例如：允许某个X值 $x_1$ 只对应到唯一的一个Y值 $y_1$，但 $y_1$ 可能对应着多个不同的X值(包括 $x_1$);
  - 同理，在‘多对一’中允许存在‘一对一’现象;
  - 不管关系R中的元组集合如何变化，X和Y在关系R中的取值对应关系的‘数量特征’是确定不变的!

[例8-2] 根据下列具体的关系实例，判断其中可能存在哪些函数依赖?

$T_1$

| X | Y |
|---|---|
| $x_1$ | $y_1$ |
| $x_2$ | $y_4$ |
| $x_3$ | $y_2$ |
| $x_4$ | $y_3$ |

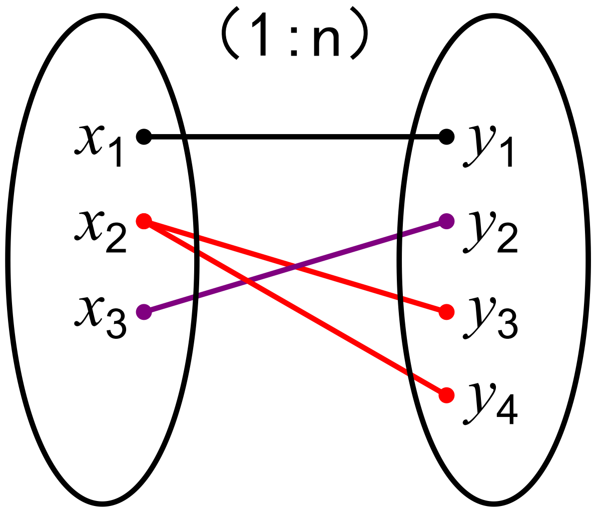



X → Y

Y → X

$T_2$

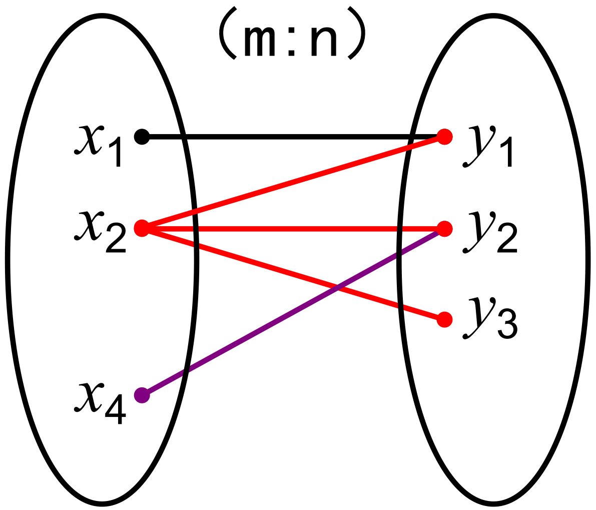
| X | Y |
|---|---|
| $x_1$ | $y_1$ |
| $x_2$ | $y_3$ |
| $x_2$ | $y_4$ |
| $x_3$ | $y_2$ |




Y → X

$T_3$

| X | Y |
|---|---|
| $x_1$ | $y_1$ |
| $x_2$ | $y_1$ |
| $x_2$ | $y_2$ |
| $x_2$ | $y_3$ |
| $x_4$ | $y_2$ |




在X与Y之间
不存在函数依赖

# 复习思考题（1）

1. 请举例说明，不好的关系模式设计会带来哪些方面的问题？

2. 请给出以下函数依赖的定义：
   ① 什么是函数依赖？
   ② 什么是平凡函数依赖？什么是非平凡函数依赖？
   ③ 什么是完全函数依赖？什么是部分函数依赖？
   ④ 什么是传递函数依赖？

3. 码
   ① 请用函数依赖给出关系的码(key)和超码(superkey)的定义；
   ② 在关系模式中，什么是主属性？什么是非主属性？请设计一个例子关系，并写出该关系的关系模式、函数依赖集、关键字、主属性集，非主属性集；
   ③ 请简要说明下列概念的相互关系：码，主码，主属性，非主属性。

# 复习思考题（2）

4. 设有一个图书借阅关系R(借书证号, 身份证号, 书号, 书名, 借阅日期, 归还日期)，其中的数据约束如下：

① 借书证号、身份证号分别是借书证、读者的标识属性；

② 一个读者只能办理一张借书证，一张借书证只能对应一个读者；

③ 每一本图书都有一个唯一的书号，不同的图书可能有相同的书名；

④ 一个读者可以同时借阅多本图书，也可以在不同时候借阅同一本图书；

⑤ 系统需要记录一本图书每一次被借阅的借阅日期和归还日期，并保存所有的借阅历史。

请写出在借阅关系R上的非平凡的完全函数依赖。

# 复习思考题（3）

5. 设有一个期末考试监考安排关系R，其中的属性有：课程的课程号(cno)和课程名(cname)，授课教师的工作证编号(tno)和姓名(tname)，监考老师的工作证编号(in_no)，每一场考试的开始时间(s_date)、结束时间(e_date)和考试教室(room)。其中：课程号和工作证编号分别是课程及教师的标识属性，开始时间和结束时间是date类型（含日期和时间）的字段，并且规定：
   ① 每一门课程至少有一位授课教师，也可能安排多位授课教师；
   ② 一位老师也可以担任多门课程的授课任务；
   ③ 每一门课的期末考试只安排一场，可分在多个教室中同时进行，除了授课教师外，在每一间考试教室中都必须安排一位或多位监考老师；
   ④ 同一时间段、同一间教室中只能安排一门课程的考试；
   ⑤ 一位老师可以担任多门课程的监考任务，但在同一时间段内，一位老师只能在指定的一间教室中监考一门课；
   ⑥ 授课教师必须参加自己承担授课任务的课程监考（不限定教室）。

**R(cno, cname, tno, tname, in_no, s_date, e_date, room)**

请找出该关系中的所有函数依赖（非平凡的完全函数依赖）。

# 数 据 管 理 基 础

## 第6章 关系数据理论

## （范式 与 规范化）

智能软件与工程学院

- 定义6.6 2NF
  - 不满足2NF可能产生的异常问题

- 定义6.7 3NF
  - 不满足3NF可能产生的异常问题

- 定义6.8 BCNF
  - 不满足BCNF可能产生的

# 1NF

- 如果一个关系模式R的所有属性都是不可分的基本数据项，则R∈1NF。
- 第一范式是对关系模式的最起码的要求。不满足第一范式的数据库模式不能称为关系数据库。
- 但是满足第一范式的关系模式并不一定是一个好的关系模式。

# 2NF（1）

- **定义6.6** 若关系模式R∈1NF，并且每一个非主属性都完全函数依赖于任何一个候选码，则R∈2NF

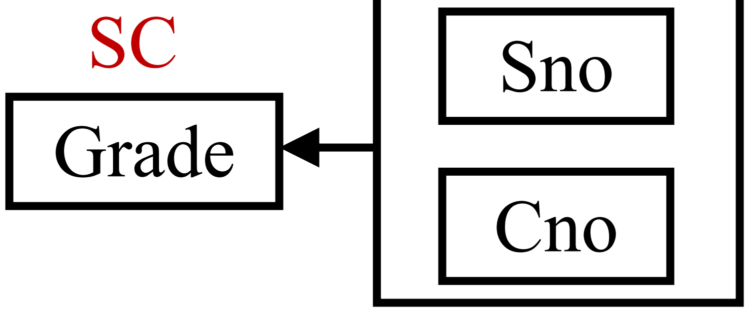
- 在一个满足2NF的关系R中，不允许存在如下的函数依赖：
  - A是关系R的非主属性，K是关系R的候选码，且A部分函数依赖于K，即 $K \xrightarrow{P} A$ .
  - 即：不允许存在“非主属性对候选码的部分函数依赖”

- 判断一个关系R是否满足2NF，需要按照以下步骤检查：
  ① 找到关系R的所有非主属性和所有的候选码；
  ② 检查每一个非主属性A和每一个候选码K之间的函数依赖，判断是否存在‘非主属性对于候选码的部分函数依赖’。

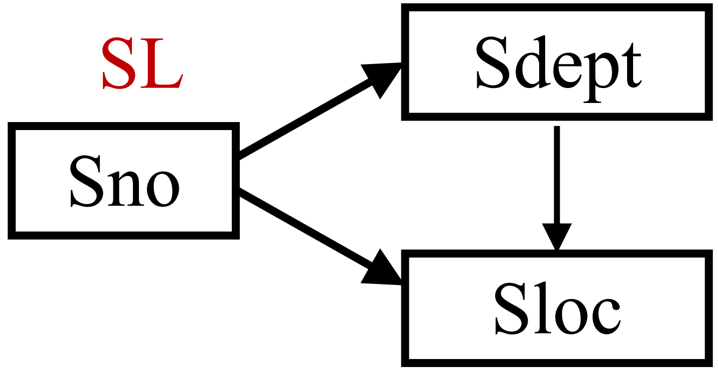
# 3NF（1）

- 定义6.7 设关系模式$R(U,F) \in 1NF$，若**R**中不存在这样的码**X**、属性组**Y**及非主属性$Z$（$Z \not\subseteq Y$），使得$X \to Y$，$Y \to Z$成立，$Y \nrightarrow X$，则称$R(U,F) \in 3NF$.

- 以关系**SC**为例：$SC \in 3NF$
  - **SC**中没有传递依赖




- 以关系**S-L**为例：***S-L*** $\notin 3NF$
  - **S-L**中有 $Sno \to Sdept,\ Sdept \nrightarrow Sno,\ Sdept \to Sloc$
  - 可得：$Sno \xrightarrow{\text{传递}} Sloc$
  - 解决的办法是将**S-L**分解成
    - ***S-D*** $(Sno, Sdept) \in 3NF$
    - ***D-L*** $(Sdept, Sloc) \in 3NF$



- **BCNF（Boyce Codd Normal Form）由Boyce和Codd提出，比3NF更进了一步。通常认为BCNF是修正的第三范式，有时也称为扩充的第三范式。**

- **定义6.8** 设关系模式$R(U,F) \in 1NF$，若 $X \to Y$ 且 $Y \not\subseteq X$ 时$X$必含有码，则$R(U,F) \in BCNF$.

- 换言之，在关系模式$R(U,F)$中，如果每一个决定属性集都包含候选码，则$R \in BCNF$.

# 复习思考题（4）

**6.** 范式定义

①请写出下列范式的定义：**1NF, 2NF, 3NF, BCNF**

②简要说明，上述各级范式之间的相互关系。

**7.** 在一个关系模式中，会因为存在下列函数依赖而产生不合理的数据冗余存储和操作异常。请利用函数依赖、候选码等概念的定义来解释其原因。

① 存在非主属性A对候选码K的部分函数依赖：$K \xrightarrow{p} A$

② 存在非主属性A对候选码K的传递函数依赖：$K \to X, X \not\subset K, X \nrightarrow K, X \to A$

③ 存在主属性B对候选码K的部分函数依赖：$K \xrightarrow{p} B$；

④ 存在主属性B对候选码K的传递函数依赖：$K \to X, X \not\subset K, X \nrightarrow K, X \to B$

# 数 据 管 理 基 础

## 第6章 关系数据理论

## （Armstrong公理系统）

智能软件与工程学院

# Armstrong公理系统

- 从已知的函数依赖中可以推导出另外一些函数依赖，这就需要定义一组推理规则。**Armstrong**公理系统就是由一组函数依赖推理规则构成的公理系统。
- **Armstrong**公理系统是模式分解算法的基础，其作用是：
  - 函数依赖的推导：从一组函数依赖求得蕴涵的函数依赖
  - 关系模式的码的计算

- 函数依赖的推理规则最早出现在**1974**年W.W.Armstrong 的论文里，这些规则常被称作 '**Armstrong** 公理'。

- 最基本的推理规则只有**3**条，由这**3**条基本规则可以定义出若干条扩充规则。

# 6.3 数据依赖的公理系统

- 定义6.11 逻辑蕴涵
- 定义6.12 函数依赖集的闭包
- Armstrong公理系统
  - 基本规则：
    - 自反律，增广律，传递律
    - 定理6.1 基本规则的正确性
  - 扩充规则
    - 合并规则，分解规则，伪传递规则
    - 引理6.1
  - Armstrong公理系统的有效性与完备性

- 定义6.13 属性集的闭包
  - 引理6.2
  - 算法6.1 属性集闭包的计算
  - 定理6.2 Armstrong公理系统是有效的、完备的。
- 定义6.14 函数依赖集的覆盖与等价
  - 引理6.3
- 定义6.15 极小函数依赖集、最小依赖集、最小覆盖
  - 定理6.3
  - 补充算法：极小函数依赖集的计算

# 6.3 数据依赖的公理系统（1）

- 定义6.11 逻辑蕴涵

- 定义6.12 函数依赖集的闭包

- Armstrong公理系统
  - 基本规则：自反律，增广律，传递律
    - 定理6.1 基本规则的正确性

  - 扩充规则：合并规则，分解规则，伪传递规则
    - 引理6.1

  - Armstrong公理系统的有效性与完备性

# 函数依赖的逻辑蕴涵

**[定义6.11] 逻辑蕴涵**

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$X \rightarrow Y$都成立（即$r$中任意两元组$r_1$、$r_2$，若$r_1[X] = r_2[X]$，则$r_1[Y] = r_2[Y]$），则称“$F$逻辑蕴涵$X \rightarrow Y$”。

- 在关系模式中，“$F$逻辑蕴涵$X \rightarrow Y$”也可以被描述为：“从$F$出发根据Armstrong公理推导得到$X \rightarrow Y$”
- “$F$ 逻辑蕴涵 $X \rightarrow Y$”可记为：$\boldsymbol{F \vDash X \rightarrow Y}$

# Armstrong公理系统 - 基本规则

- 设有关系模式$R(U,F)$，$U$是关系模式$R$的属性集总体，$F$是$U$上的一组函数依赖。
- 对$R(U,F)$来说有以下的推理规则：
  - **A1 自反律 (reflexivity rule)**：若$Y \subseteq X \subseteq U$，则$X \rightarrow Y$为$F$所蕴涵。
  - **A2 增广律 (augmentation rule)**：若$X \rightarrow Y$为$F$所蕴涵，且$Z \subseteq U$，则$XZ \rightarrow YZ$为$F$所蕴涵。
  - **A3 传递律 (transitivity rule)**：若$X \rightarrow Y$及$Y \rightarrow Z$为$F$所蕴涵，则$X \rightarrow Z$为$F$所蕴涵。

- 注意：使用自反律推导得到的函数依赖均是平凡函数依赖，自反律的使用并不依赖于$F$。

# Armstrong公理系统 - 基本规则

- 定理6.1 Armstrong推理规则是正确的。

  - 自反律、增广律、传递律是Armstrong公理中的三条基本规则
  - 只能根据函数依赖的定义来证明其正确性

# Armstrong公理系统 - 扩充规则

- 根据A1自反律、A2增广律、A3传递律这三条推理规则可以得到下面三条推理规则（扩充规则）：
  - **合并规则 (union rule):**
    若 $X \to Y$ 及 $X \to Z$，则 $X \to YZ$.
  - **分解规则 (decomposition rule):**
    若 $X \to Y$ 及 $Z \subseteq Y$，则 $X \to Z$.
  - **伪传递规则 (pseudo transitivity rule):**
    若 $X \to Y$ 及 $WY \to Z$，则 $XW \to Z$.

- 可以使用前面的三条基本规则来证明扩充规则。
- 也可以根据函数依赖的定义来证明扩充规则（证明规则推导结果函数依赖的成立）

# 引理 6.1

## 引理6.1

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

- 可以根据Armstrong公理中的合并规则和分解规则来证明引理6.1（证明过程略）

## [课后思考]

不使用Armstrong公理中的基本推导规则，根据函数依赖的定义来给出合并规则、分解规则、伪传递规则的证明。

# 函数依赖集的闭包

- [定义6.12] 函数依赖集$F$的闭包（$F^+$）

在关系模式$R(U,F)$中，为$F$所逻辑蕴涵的函数依赖的全体叫作$F$的闭包，记为$F^+$。

- 函数依赖集$F$闭包的定义，可表示为：

$$F^+ = \{ X \to Y \mid F \vDash X \to Y \}$$

# Armstrong公理系统的 有效性 与 完备性

- 人们把自反律、增广律和传递律合称为Armstrong公理系统。
- Armstrong公理系统是有效的、完备的。
  - 有效性：由$F$出发根据Armstrong公理推导出来的每一个函数依赖一定在闭包$F^+$中；
  - 完备性：闭包$F^+$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。

- 根据函数依赖集闭包的定义，可以说明Armstrong公理的有效性；但要证明Armstrong公理的'完备性'，首先需要解决：如何判定一个函数依赖$X \rightarrow Y$是否属于"由$F$根据Armstrong公理推导出来的函数依赖的集合"，即判断 $X \rightarrow Y \in \boldsymbol{F}^+$ 是否成立？
  - 方法一：首先计算$F^+$，然后判断 $X \rightarrow Y \in \boldsymbol{F}^+$ 是否成立？
  - 方法二：引入属性集闭包$\boldsymbol{X_F}^+$的概念并计算之，然后判断$\boldsymbol{Y} \subseteq \boldsymbol{X_F}^+$是否成立？

# 6.3 数据依赖的公理系统（2）

- 函数依赖集闭包$F^+$的计算是一个NP问题！
  - 函数依赖集闭包的计算示例

- 定义6.13 属性集的闭包
  - 引理6.2
  - 算法6.1 属性集闭包的计算
  - 定理6.2 Armstrong公理系统是有效的、完备的

# 属性集闭包（1）

- 定义6.13 设$F$为属性集$U$上的一组函数依赖，$X \subseteq U$，

$$X_F^+ = \{ A \mid X \to A \text{ 能由}F\text{根据}Armstrong\text{公理导出}, A \in U \}$$

$X_F^+$ 称为**属性集$X$关于函数依赖集$F$的闭包**。

  - “$X \to A$ 能由$F$根据$Armstrong$公理导出” 即 “$F$逻辑蕴涵$X \to A$”，可表示为：$F \vDash X \to A$

  - 属性集$X$关于函数依赖集$F$的闭包定义可简化表示如下：

$$X_F^+ = \{ A \mid A \in U \text{ 且 } F \vDash X \to A \}$$

- 这样，判定一个函数依赖 $X \to Y$ 是否属于“由$F$根据Armstrong公理推导出来的函数依赖的集合”，就可以被转化成判断$Y \subseteq X_F^+$是否成立。

  - 如果 $Y \subseteq X_F^+$ 则 $F \vDash X \to Y$ 成立

  - 如果 $Y \not\subseteq X_F^+$ 则 $F \vDash X \to Y$ 不成立

# 属性集闭包（2）

- **[引理6.2]** 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$Y \subseteq X_F^+$ .

- 引理6.2的用途：
  - 判定$X \to Y$是否能由$F$根据Armstrong公理导出的问题
  - 就转化为：求出 $X_F^+$，然后判定$Y$是否为 $X_F^+$ 的子集的问题
    - 即：判断$Y \subseteq X_F^+$是否成立
  - 描述性证明如下：
    - 假设$Y$是由$A_1, A_2, ..., A_k$等$k$个属性组成的属性集
    - 由引理6.1可知："$X \to A_1A_2...A_k$成立"的充分必要条件是"$X \to A_i$成立$(i = 1,2,...,k)$"
    - "$X \to A_i$ 成立$(i = 1,2,...,k)$" 的充分必要条件是 "$A_i \in X_F^+$成立 $(i = 1,2,...,k)$"，即：$A_1A_2...A_k \subseteq X_F^+$

# 算法6.1：求属性集 $X\ (X \subseteq U)$ 关于$U$上的函数依赖集$F$的闭包$X_F^+$

- 输入：函数依赖集$F$，属性集$X$
- 输出：属性集闭包 $X_F^+$

① 令$X^{(0)} = X$，$i = 0$

② 求$Y$，这里$Y = \{ A \mid (\exists V)(\exists W)(V \rightarrow W \in F \wedge V \subseteq X^{(i)} \wedge A \in W)\}$.

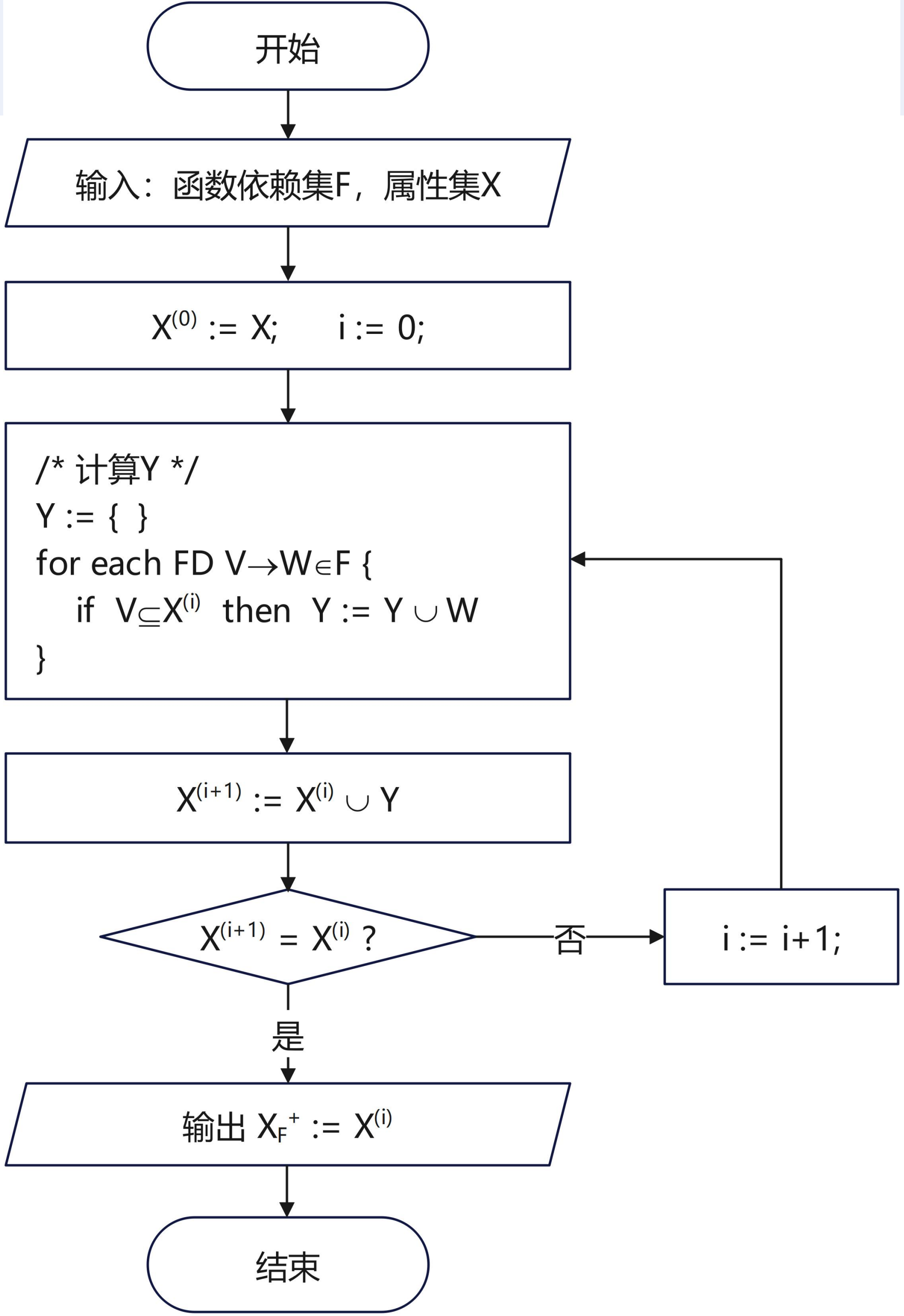
③ $X^{(i+1)} = Y \cup X^{(i)}$.

④ 判断 $X^{(i+1)} = X^{(i)}$ 是否成立?

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

⑥ 若否，则 $i = i + 1$，返回第②步。

# 算法6.1流程图


开始
输入：函数依赖集F，属性集X
X(0) := X;  i := 0;
/* 计算Y */
Y := { }
for each FD V→W∈F {
if V⊆X(i) then Y := Y ∪ W
}
X(i+1) := X(i) ∪ Y
X(i+1) = X(i) ?
否
i := i+1;
是
输出 XF+ := X(i)
结束

# 属性集闭包的计算示例

❑ **[例6.11]** 已知关系模式$R(U,F)$，其中：$U=\{A,B,C,D,E\}$，
$F=\{AB\to C, B\to D, C\to E, EC\to B, AC\to B\}$. 求$(AB)_F^+$ .

解：

- $X^{(0)}=AB$
- 计算$X^{(1)}$：
  - 逐一扫描集合F中的各个函数依赖，寻找左部为$X^{(0)}$的子集的函数依赖；
  - 得到两个函数依赖：$AB\to C$，$B\to D$. 于是：$Y=\{C,D\}$
  - 于是：$X^{(1)}=Y\cup X^{(0)}=\{A,B,C,D\}$
- 因为$X^{(1)}\neq X^{(0)}$，所以计算$X^{(2)}$：
  - 逐一扫描集合F中的各个函数依赖，寻找左部为$X^{(1)}$的子集的函数依赖；
  - 得到两个函数依赖：$C\to E$，$AC\to B$. 于是：$Y=\{B,E\}$
  - 于是：$X^{(2)}=Y\cup X^{(1)}=\{A,B,C,D,E\}$
- 因为$X^{(2)}$已等于全部属性集合$U$，所以 $(AB)_F^+=X^{(2)}=\{A,B,C,D,E\}$ .

# Armstrong公理系统的有效性与完备性

- 有效性与完备性的含义
  - 有效性：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。

- [定理6.2] Armstrong公理系统是有效的、完备的。

- Armstrong公理的完备性及有效性说明:
  - “导出”与“蕴涵”是两个完全等价的概念
  - $F^+$：为$F$所逻辑蕴涵的函数依赖的全体（定义6.12）
  - 可以等价地说成：由$F$出发借助Armstrong公理导出的函数依赖的集合

- **定义6.14 函数依赖集的覆盖 & 等价**
  - **引理6.3**

- **定义6.15 极小函数依赖集、最小依赖集、最小覆盖**
  - **定理6.3**

# 函数依赖集的覆盖与等价

- **函数依赖集的覆盖**
  - 设一个关系模式R，F和G是R上的两个函数依赖集，如果F逻辑蕴涵G中的所有函数依赖，则称：F覆盖G（或称 F是G的覆盖）
  - 如果F覆盖G，则 $G \subseteq F^+$

- [定义6.14] **函数依赖集的等价**

  如果两个函数依赖集的闭包相等（即 $F^+ = G^+$），则称F与G等价。

- [引理6.3] $F^+ = G^+$ 的充分必要条件是 $F \subseteq G^+$ 和 $G \subseteq F^+$.
  - 引理6.3表明，两个函数依赖集相互等价的充分必要条件是它们相互覆盖。即：

    “F与G等价” 的充分必要条件是 “F覆盖G 且 G覆盖F”

# 极小函数依赖集（最小依赖集、最小覆盖）

- [定义6.15] 如果函数依赖集F满足下列条件，则称F为一个**极小函数依赖集**，亦称为**最小依赖集**或**最小覆盖**。
  ① F中任一函数依赖的右部仅含有一个属性；
  ② F中不存在这样的函数依赖$X \to A$，使得$F$与$F-\{X \to A\}$等价；
  ③ F中不存在这样的函数依赖$X \to A$，$X$有真子集$Z$使得
  $F-\{X \to A\} \cup \{Z \to A\}$与$F$等价。

- 其中：
  - 条件①并不是必须的，只是为了方便对条件②和③的检查；
  - 条件②表明，在极小函数依赖集F中，不允许存在冗余的函数依赖（可以由F中的其他函数依赖导出）；
  - 条件③表明，在极小函数依赖集F中，不允许存在部分函数依赖。

# 定理6.3

- [定理6.3] 每一个函数依赖集F均等价于一个极小函数依赖集$F_m$。此$F_m$称为F的最小依赖集。
- 证：构造性证明，分三步对F进行“极小化处理”，找出F的一个最小依赖集。

① 逐一检查F中各函数依赖$FD_i$：$X \rightarrow Y$，若$Y = A_1A_2...A_k$，$k \geq 2$，则用$\{X \rightarrow A_j \mid j = 1,2,...,k\}$来取代$X \rightarrow Y$。

  - 引理6.1保证了F变换前后的等价性

② 逐一检查F中各函数依赖$FD_i$：$X \rightarrow A$，令$G = F - \{X \rightarrow A\}$，若$A \in \boldsymbol{X}_G^+$，则从F中去掉此函数依赖。

  - 由于F与G 等价的充要条件是$A \in \boldsymbol{X}_G^+$，因此F变换前后是等价的。

③ 逐一取出F中各函数依赖$FD_i$：$X \rightarrow A$，设$X = B_1B_2...B_m$，$m \geq 2$，逐一考查$B_i$ $(i = 1,2,...,m)$，若 $A \in (X - B_i)_F^+$ ，则以 $(X - B_i) \rightarrow \boldsymbol{A}$ 取代 $X \rightarrow \boldsymbol{A}$ 。

  - 由于$F$与$F - \{X \rightarrow A\} \cup \{Z \rightarrow A\}$等价的充要条件是$A \in \boldsymbol{Z}_F^+$，其中$Z = (X - B_i)$ 。

- 最后剩下的F就一定是极小依赖集。因为对F的每一次“改造”都保证了改造前后的两个函数依赖集等价，因此剩下的F与原来的F等价。（证毕）

# 【补充算法1】寻找与函数依赖集 F 等价的极小函数依赖集

- 输入：函数依赖集F
- 输出：与F等价的极小函数依赖集G
- 算法（核心计算任务）

> ① 消除F中的部分函数依赖（如果存在部分函数依赖，则需要将其简化为完全函数依赖）
>
> ② 消除冗余的函数依赖（如果存在一个函数依赖X→Y，能够被从F中的其他函数依赖中推导得到，那么需要从F剔除X→Y）

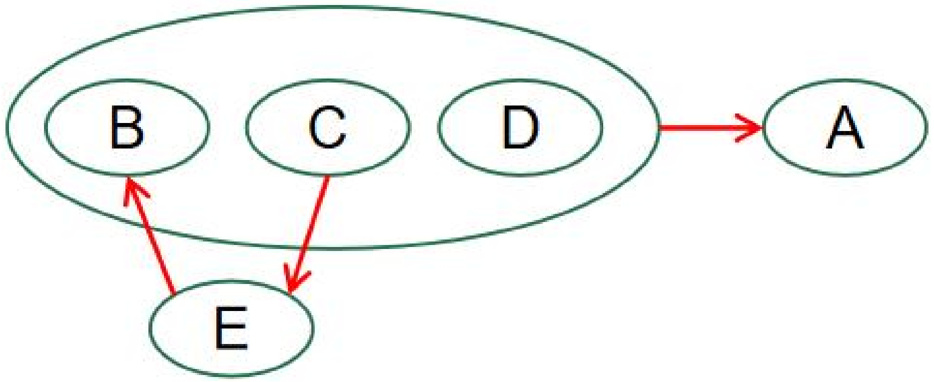
- 其中，步骤①和步骤②的检查顺序可以对调，但必须保证最终的计算结果中没有冗余的函数依赖！
- 具体计算过程如下(next slide)

# 【补充算法1】寻找与函数依赖集 F 等价的极小函数依赖集 G

1. 令 G = F
   - 将 G 中每一个形如 $X \to (A_1, A_2, ..., A_n)$ 的函数依赖替换为如下一组依赖因素为单个属性的函数依赖：$X \to A_1$, $X \to A_2$, …, $X \to A_n$
2. 对 G 中的每一个函数依赖 $X \to A$ 作如下的处理：
   - 对决定因素 X 中的每一个属性 B 作如下处理：
     1) 计算属性集的闭包 $(X - B)_G{}^+$;
     2) 如果 $A \in (X - B)_G{}^+$, 则用新的函数依赖 $(X - B) \to A$ 替换原来的函数依赖 $X \to A$；
3. 对 G 中的每一个函数依赖 $X \to A$ 作如下处理：
   1) 令 $N = G - \{ X \to A \}$；
   2) 计算属性集的闭包 $X_N{}^+$；
   3) 如果 $A \in X_N{}^+$, 那么从G中删去函数依赖 $X \to A$ ；
4. 将 G 中每一组形如 $X \to A_1$, $X \to A_2$, …, $X \to A_n$(决定因素相同)的函数依赖合并为一个函数依赖：$X \to (A_1, A_2, ..., A_n)$

# 【补充算法1】之步骤2示例（消除部分函数依赖）

- 设有如下图所示的函数依赖集 $M = \{C \to E, E \to B, BCD \to A\}$



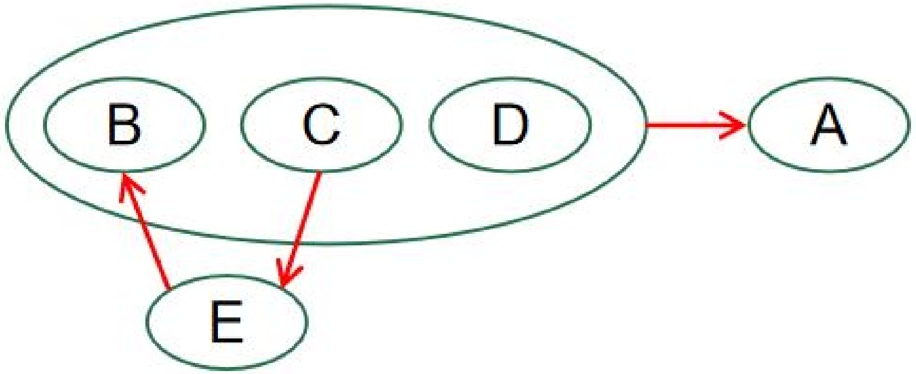

其中：$BCD \to A$ 是一个部分函数依赖！(决定因素中的属性B是多余的，可以将其简化为 $CD \to A$ )

- 要得到上述结论，我们需要做以下的证明：

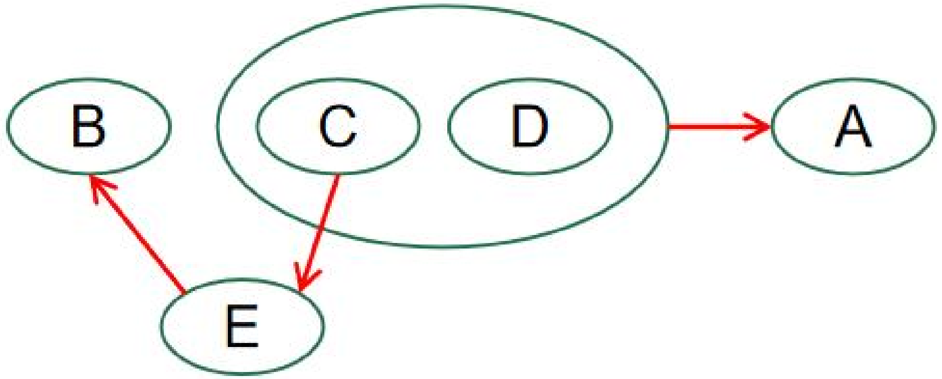
① 用 $CD \to A$ 替换 $BCD \to A$，从而得到一个新的函数依赖集

$$N = \{ C \to E, E \to B, CD \to A \}$$

② 证明：$\boldsymbol{M^+ = N^+}$ **(next slide)**




$$M = \{ C \to E,\ E \to B,\ BCD \to A \}$$



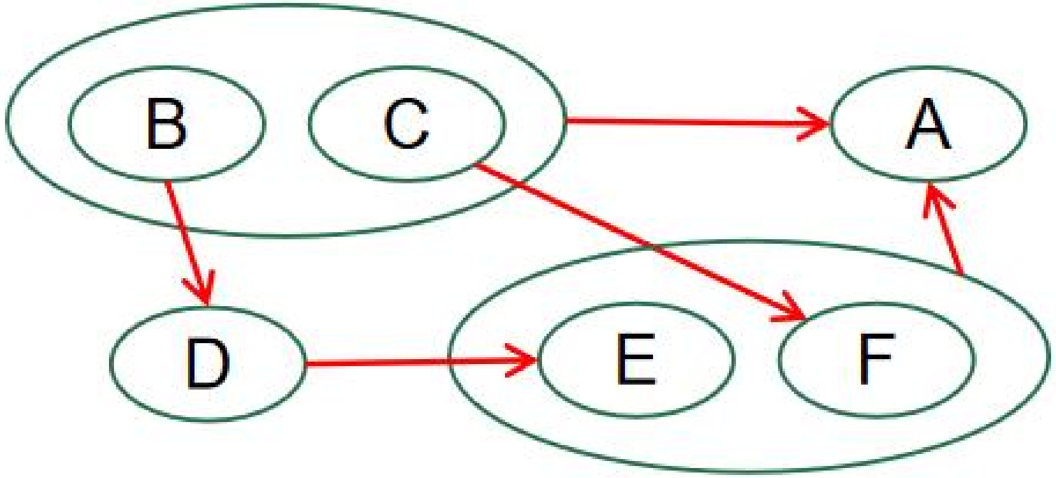

$$N = \{ C \to E,\ E \to B,\ CD \to A \}$$

要证明 $\boldsymbol{M}^+ = \boldsymbol{N}^+$，只需要证明“M逻辑蕴涵 $CD \to A$ 且 N逻辑蕴涵 $BCD \to A$”，具体思路如下：

$$\boldsymbol{M}^+ = \boldsymbol{N}^+ ? \Rightarrow \begin{cases} M \vDash CD \to A ? & \Rightarrow \boldsymbol{A} \in \{\boldsymbol{C}, \boldsymbol{D}\}_{\boldsymbol{M}}^+ ? \\ N \vDash BCD \to A ? & \Rightarrow \text{EASY} \end{cases}$$

- 设有如右图所示的函数依赖集

$$M = \{ B \to D,\; D \to E,\; C \to F,\; BC \to A,\; EF \to A \}$$



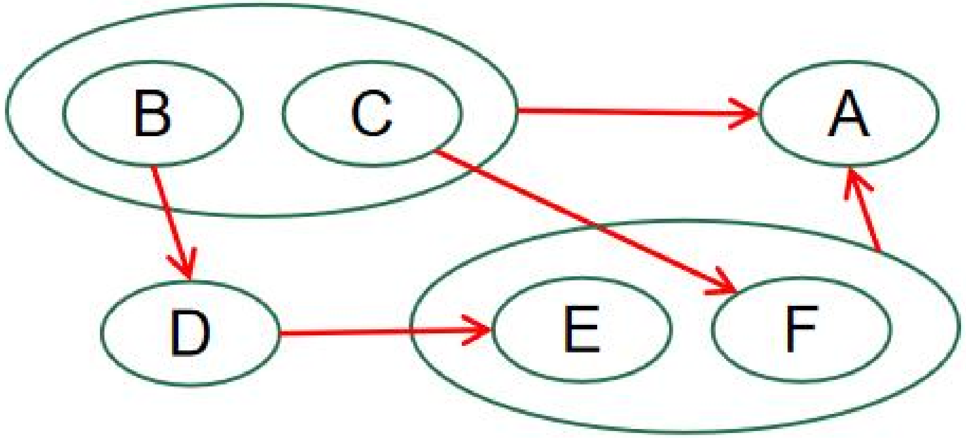

其中：$BC \to A$ 是一个冗余的函数依赖！

- 要得到上述结论，我们需要做以下的证明：

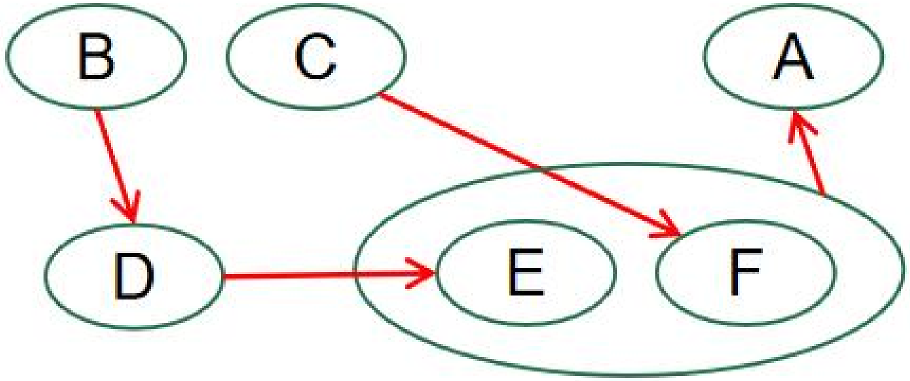
① 从M中删除 $BC \to A$，从而得到一个新的函数依赖集

$$N = \{ B \to D,\; D \to E,\; C \to F,\; EF \to A \}$$

② 证明：$M^+ = N^+$ (next slide)




$M = \{\ B \to D,$
$D \to E,$
$C \to F,$
$BC \to A,$
$EF \to A\ \}$




$N = \{\ B \to D,$
$D \to E,$
$C \to F,$
$EF \to A\ \}$

因为N是M的一个子集，要证明 $\boldsymbol{M^+ = N^+}$，只需要证明“N逻辑蕴涵 $BC \to A$”，具体思路如下：

$\boldsymbol{M^+ = N^+}$ ? $\Rightarrow$ $N \vDash BC \to A$? $\Rightarrow$ $\boldsymbol{A \in \{B, C\}_N^+}$ ?

# 复习思考题（5）

**8.  Armstrong公理系统**

①请写出三条基本规则：自反律，增广律，传递律

②请写出以下两条扩充规则并证明：分解规则，合并规则

③请举例说明：利用Armstrong公理系统中的传递规则推导得到的函数依赖不一定是传递函数依赖。

**9. ‘函数依赖集闭包’与‘属性集闭包’**

①什么是函数依赖的逻辑蕴涵?

②什么是函数依赖集闭包?

③什么是属性集闭包？请写出属性集闭包的计算算法。

**10.‘函数依赖集等价’与‘极小函数依赖集’**

①请写出下列概念的定义：函数依赖集等价，极小函数依赖集

②极小函数依赖集的判定方法是什么？请写出极小函数依赖集的计算算法。

# 复习思考题（6）

11. 请利用Armstrong公理系统证明下面的推导过程是否成立？如果不成立，请给出具体的例子关系。

1. $\{ W \rightarrow Y, X \rightarrow Z \} \Rightarrow \{ WX \rightarrow Y \}$
2. $\{ X \rightarrow Y \}$ and $Z \subseteq Y \Rightarrow \{X \rightarrow Z \}$
3. $\{ X \rightarrow Y, X \rightarrow W, WY \rightarrow Z \} \Rightarrow \{ X \rightarrow Z \}$
4. $\{ XY \rightarrow Z, Y \rightarrow W \} \Rightarrow \{ XW \rightarrow Z \}$
5. $\{ X \rightarrow Z, Y \rightarrow Z \} \Rightarrow \{ X \rightarrow Y \}$
6. $\{ X \rightarrow Y, XY \rightarrow Z \} \Rightarrow \{ X \rightarrow Z \}$
7. $\{ X \rightarrow Y, Z \rightarrow W \} \Rightarrow \{ XZ \rightarrow YW \}$
8. $\{ XY \rightarrow Z, Z \rightarrow X \} \Rightarrow \{ Z \rightarrow Y \}$
9. $\{ X \rightarrow Y, Y \rightarrow Z \} \Rightarrow \{ X \rightarrow YZ \}$
10. $\{ XY \rightarrow Z, Z \rightarrow W \} \Rightarrow \{ X \rightarrow W \}$

# 复习思考题（7）

**12.** 设 $F=\{A\to C, AC\to D, E\to AD, E\to H\}$，**请给出 F 的极小函数依赖集。**

**13.** $\boldsymbol{M}=\{ABD\to AC, C\to BE, AD\to BF, B\to E\}$，**请计算M的极小函数依赖集。**

南京大学 NANJING UNIVERSITY

奋进行动 团结奋斗 争先进位 坐言起行 应势而动

# 数 据 管 理 基 础

## 第6章 关系数据理论

## （模式分解）

智能软件与工程学院

# 6.4 模式的分解

- 定义6.16 关系模式的分解
- 定义6.17 函数依赖集在属性组上的投影
- 模式分解的三个定义
  - 准确说：两条性质，三种分解准则
- 引理6.4
- 无损连接
  - 定义6.18，定理6.5
  - （自学）算法6.2 & 定理6.4
- 保持函数依赖
  - 定义6.19
- 模式分解算法：算法6.3、6.4、6.5
  - 引理6.5、6.6
  - 定理6.6，算法6.6
- 补充算法

- **[定义6.16]** 关系模式$R(U,F)$的一个分解是指

$$\rho=\{R_1(U_1,F_1),R_2(U_2,F_2),\cdots,R_n(U_n,F_n)\}$$

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$F$在属性集$U_i$上的投影，也被称为‘$F_i$是$F$在关系$R_i$上的投影’

- **[定义6.17]** 与函数依赖集合$\{X\to Y\mid X\to Y\in F^{+}\ \wedge\ XY\subseteq U_i\}$等价的$F_i$叫作$F$在属性集$U_i$上的投影。
  - 在一个关系模式$R(U,F)$中，可以用与$F$等价的依赖集$G$来取代$F$。
  - 原因：两个关系模式$R_1(U,F)$和$R_2(U,G)$，如果$F$与$G$等价，那么$R_1$的关系一定是$R_2$的关系；反过来，$R_2$的关系也一定是$R_1$的关系。

# 6.4 模式的分解

- 6.4.1 模式分解的三个定义
- 6.4.2 模式分解的无损连接性和保持函数依赖性
- 6.4.3 模式分解的算法

# 6.4.1 模式分解的三个定义

- 对于一个模式的分解是多种多样的，但是分解后产生的模式应与原模式等价。
- 从不同的角度去观察，对‘等价’的概念形成了三种不同的定义：
  - 无损连接 (lossless join)
  - 保持函数依赖 (preserve functional dependency)
  - 既有‘无损连接’，又要‘保持函数依赖’
- 其中：
  - ‘无损连接’是进行模式分解必须满足的要求；
  - 单纯考虑一个分解$\rho$满足‘保持函数依赖’很容易实现，但这样的分解不一定满足‘无损连接性’；
  - 模式分解的目标：既有‘无损连接’，又要‘保持函数依赖’（但不一定能满足）

# 6.4.2 模式分解的无损连接性和保持函数依赖性

- **符号定义**
  - 设 $\rho=\{R_1(U_1,F_1),R_2(U_2,F_2),\cdots,R_k(U_k,F_k)\}$是$R(U,F)$的一个分解，$r$是$R(U,F)$的一个关系
  - $r_i=\pi_{R_i}(r)=\{t.U_i\mid t\in r\}$是关系$r$在关系模式$R_i$上的投影
  - 定义$m_\rho(r)=\pi_{R_1}(r)\bowtie\pi_{R_2}(r)\bowtie\cdots\bowtie\pi_{R_k}(r)$，即$m_\rho(r)$是$r$在$\rho$中各关系模式上投影的连接。

- **[引理6.4]** 设$R(U,F)$是一个关系模式，$\rho=\{R_1(U_1,F_1),R_2(U_2,F_2),\cdots,R_k(U_k,F_k)\}$是$R$的一个分解，$r$是$R$的一个关系，$r_i=\pi_{R_i}(r)$，则

  ① $r\subseteq m_\rho(r)$

  ② 若 $s=m_\rho(r)$，则 $\pi_{R_i}(s)=r_i$，即 $\pi_{R_i}(m_\rho(r))=r_i$

  ③ $m_\rho(m_\rho(r))=m_\rho(r)$

说明：①可以对$k$用归纳法来证明；②如果关系$R$和$S$不存在同名属性，那么$R\bowtie S=R\times S$.

# 模式分解的无损连接性

- **[定义6.18]**

  设 $\rho=\{R_1(U_1,F_1),R_2(U_2,F_2),\cdots,R_k(U_k,F_k)\}$ 是 $R(U,F)$ 的一个分解，若对$R(U,F)$的任何一个关系$r$均有$r=m_\rho(r)$成立，则称分解$\rho$具有**无损连接性**。简称$\rho$为**无损分解**。

- 对于不具备‘无损连接性’的分解来说，关系 $r$ 与分解后的子关系满足：

  $$r \subset \boldsymbol{\pi}_{R_1}(r) \bowtie \boldsymbol{\pi}_{R_2}(r) \bowtie ... \bowtie \boldsymbol{\pi}_{R_k}(r)$$

# 模式分解无损连接性的判定

- **[定理6.5]** 对于$R(U,F)$的一个分解$\rho=\{R_1(U_1,F_1),R_2(U_2,F_2)\}$，如果 $(\boldsymbol{U_1}\cap\boldsymbol{U_2})\rightarrow(\boldsymbol{U_1}-\boldsymbol{U_2})\in\boldsymbol{F}^+$ 或 $(\boldsymbol{U_1}\cap\boldsymbol{U_2})\rightarrow(\boldsymbol{U_2}-\boldsymbol{U_1})\in\boldsymbol{F}^+$ 成立，则分解$\rho$具有无损连接性。

- 说明：
  - 对$\boldsymbol{k}$用归纳法，可以判断$\boldsymbol{\rho}=\{\boldsymbol{R_1}(\boldsymbol{U_1},\boldsymbol{F_1}),\boldsymbol{R_2}(\boldsymbol{U_2},\boldsymbol{F_2}),\cdots,\boldsymbol{R_k}(\boldsymbol{U_k},\boldsymbol{F_k})\}$是不是$\boldsymbol{R}(\boldsymbol{U},\boldsymbol{F})$的无损分解；
  - 下面给出定理6.5的证明。

# 定理6.5证明（1）

- [思路] 对$R(U,F)$的任何一个关系$r$，关系$r$在$R_1(U_1,F_1)$和$R_2(U_2,F_2)$上的投影是$r_1$ 和 $r_2$，证明：$\boldsymbol{r \subseteq r_1 \bowtie r_2}$ ***and*** $\boldsymbol{r_1 \bowtie r_2 \subseteq r}$

- 设：$U_1 \cap U_2 = A$，$U_1 - U_2 = B$，$U_2 - U_1 = C$，则有：
  - $U_1 = AB,\ U_2 = AC,\ U = ABC$
  - $(U_1 \cap U_2) \rightarrow (U_1 - U_2)$ 就是 $A \rightarrow B$
  - $(U_1 \cap U_2) \rightarrow (U_2 - U_1)$ 就是 $A \rightarrow C$

下面证明：当 $A \rightarrow B$ 成立时，对$R(U,F)$的任何一个关系$r$，经分解投影得到两个子关系 $r_1 = r[A,B]$ 和 $r_2 = r[A,C]$，必有 $r = (r_1 \bowtie r_2)$ 成立，即下述两个结论成立：

结论1：**if** $t \in r$ **then** $t \in (r_1 \bowtie r_2)$

结论2：**if** $t \in (r_1 \bowtie r_2)$ **then** $t \in r$

# 定理6.5证明（2）

**结论1：if $t \in r$ then $t \in (r_1 \bowtie r_2)$**

- 对关系$r$中的任意一个元组$t$，它在两个子关系中的投影结果分别是关系$r_1$中的元组$t_1$和关系$r_2$中的元组$t_2$，并满足：

  $t_1[A] = t[A] = t_2[A], t_1[B] = t[B], t_2[C] = t[C]$ ……………………………①

- 因为有 $t_1 \in r_1, t_2 \in r_2$ 且 $t_1[A] = t_2[A]$，所以由$t_1$和$t_2$组合得到的元组

  $(t_1[A], t_1[B], t_2[C]) \in (r_1 \bowtie r_2)$ ……………………………………………②

- 依据式①中的等价关系，可将式②变换为

  $(t[A], t[B], t[C]) \in (r_1 \bowtie r_2)$ 即： $t \in (r_1 \bowtie r_2)$

- 所以，结论1 成立。

# 定理6.5证明（3）

**结论2：if $t \in (r_1 \bowtie r_2)$ then $t \in r$**

- 对$(r_1 \bowtie r_2)$中的任意一个元组$t$，必存在关系$r_1$中的一个元组$t_1$和关系$r_2$中的一个元组$t_2$，并满足：

  $t[A] = t_1[A] = t_2[A], t[B] = t_1[B], t[C] = t_2[C]$ ……③

- 因为 $t_1 \in r_1$, 所以在关系$r$中必存在一个元组$s_1$，$t_1$是$s_1$在关系$R_1$上的投影，即：存在元组 $s_1 \in r$ 且满足

  $s_1[A] = t_1[A], s_1[B] = t_1[B]$ ……④

- 同理，有：存在元组 $s_2 \in r$ 且满足

  $s_2[A] = t_2[A], s_2[C] = t_2[C]$ ……⑤

- 由式③④⑤可得

  $s_1[A] = t_1[A] = t_2[A] = s_2[A]$，即：$s_1[A] = s_2[A]$ ……⑥

- 在关系$r$中,由$s_1 \in r, s_2 \in r, s_1[A] = s_2[A]$, 及$A \to B$可知：

  $s_1[B] = s_2[B]$ ……⑦

# 定理6.5证明（4）

$t[A]=t_1[A]=t_2[A], t[B]=t_1[B], t[C]=t_2[C]$ ……………………③

$s_1[A]=t_1[A], s_1[B]=t_1[B]$ ……………………④

$s_2[A]=t_2[A], s_2[C]=t_2[C]$ ……………………⑤

$s_1[A]=s_2[A]$ ……………………⑥

$s_1[B]=s_2[B]$ ……………………⑦

- 由式③④⑤⑥⑦可知：$s_2[A]=t_2[A]=t[A]$
  $s_2[B]=s_1[B]=t_1[B]=t[B]$
  $s_2[C]=t_2[C]=t[C]$

  即 $s_2$ 和 $t$ 是同一个元组
- 既然 $s_2$ 是关系 $r$ 中的元组，那么 $t$ 也是 $r$ 中的元组，即： $t\in r$
- 所以，结论 2 也成立。

综上所述，结论1 和 结论2 都成立，所以有：$r=(r_1\bowtie r_2)$ （证毕）

# 模式分解的保持函数依赖

- **[定义6.19]** 若 $F^+ = (F_1 \cup F_2 \cup \cdots \cup F_k)^+$，则$R(U,F)$的分解 $\rho = \{R_1(U_1,F_1), R_2(U_2,F_2), \cdots, R_k(U_k,F_k)\}$ 保持函数依赖。

- 可以根据引理6.3，找到判断分解$\rho$是否保持函数依赖的方法，即判断$F$和$(F_1 \cup F_2 \cup \cdots \cup F_k)$是否等价的方法。
  - 显然，$(F_1 \cup F_2 \cup \cdots \cup F_k) \subseteq F^+$ 成立
  - 只需要判断：$F \subseteq (F_1 \cup F_2 \cup \cdots \cup F_k)^+$ 是否成立？
    - 如果$F \subseteq (F_1 \cup F_2 \cup \cdots \cup F_k)^+$ 成立，则分解$\rho$保持函数依赖；
    - 否则，分解$\rho$不具有保持函数依赖性。

# 6.4.3 模式分解的算法

- 已经介绍的算法
  - 算法6.1：求属性集 $X$ $(X \subseteq U)$ 关于 $U$ 上的函数依赖集 $F$ 的闭包 $X_F^+$
  - [补充算法1] 极小函数依赖集的计算
    - 寻找与函数依赖集 F 等价的极小函数依赖集 G

- 接下来
  - [补充算法2] 候选码的计算
  - 算法6.4：转换为3NF既有无损连接性又保持函数依赖的分解。
  - 引理6.5 & 引理6.6

# [补充算法2] 候选码的计算（1）

- 如何寻找一个关系模式$R(U,F)$的候选码？

**方法一：运用Armstrong公理系统做如下推导：**$F \vDash K \xrightarrow{f} U$

【困难】依赖于个人经验和对推理规则的熟练使用。

**方法二：运用属性集闭包的概念，寻找满足条件$(K_F^+ == U)$的最小属性集合K**

【优点】有算法支持（属性集闭包计算 & 候选码计算）
【缺点】计算工作量大

**方法三：运用极小函数依赖集来优化方法二中的候选码计算。**

# [补充算法2] 候选码的计算（2）

- 设有关系模式$R(U,F)$，$U$是关系$R$的属性集合，$F$是关系上的极小函数依赖集。根据$F$中的函数依赖，可以将属性集合$U$划分为以下的三个子集：
  1. 只在函数依赖的左边出现过的属性的集合 $U_L$（包括没有出现在任何函数依赖中的属性）
  2. 只在函数依赖的右边出现过的属性的集合 $U_R$
  3. 在两边都出现过的属性的集合 $U_A$

- 其中：
  - $U_L$ 中的属性是每一个候选码的组成部分；
  - $U_R$ 中的属性不可能出现在任何一个候选码中；
  - 在候选码计算中，只需要对 $U_A$ 中的属性进行**FOR**循环检查。

# [补充算法2] 候选码的计算（3）

**输入：关系 $R$ 的属性集合 $U$，极小函数依赖集 $F$**

**只在函数依赖的右边出现过的属性集合 $U_R$**

**在函数依赖的左右两边都出现过的属性集合 $U_A$**

**输出：候选码 $K$**

**set** $K := U - U_R$ ;

**for each attribute** $A$ **in** $U_A$     */* 对$U_A$中属性的不同处理顺序，可能得到不同的计算结果K */*

{

    **compute** $(K - A)_F^+$ ;

    **if** $(K - A)_F^+$ **contains all the attributes in** $R$

    **then**   **set** $K := K - A$ ;

}

**return** $K$ ;

# 算法6.3&6.4：转换为3NF既有无损连接性又保持函数依赖的分解

- 输入：关系模式 $R(U, F)$
- 输出：满足3NF且既有无损连接性又保持函数依赖的分解 $\rho$

1) 计算F的极小函数依赖集，并用来代替F进行后续的模式分解;
2) $\rho = \emptyset$; // 初始化分解 $\rho$ 为空集
3) 对 F 中的每一个函数依赖 $X \rightarrow Y$ 做如下处理：
   - 如果在分解 $\rho$ 中找不到满足下述条件的关系模式 $Z(U_Z, F_Z)$：$XY \subseteq U_Z$
   - 则由X和Y合并构成一个新的子关系模式并加入到分解 $\rho$ 中;
4) 如果关系 R 的所有候选码都没有出现在分解 $\rho$ 的关系模式中，即：找不到一个原关系R的候选码K和一个关系模式 $Z(U_Z, F_Z)$ $(Z \in \rho)$，满足 $K \subseteq U_Z$

   那么，就从关系R中任选一个候选码K, 由K中的属性单独构成一个关系模式并加入到分解 $\rho$ 中去。

# 算法6.5：转换为BCNF的无损连接分解

- 输入：关系模式$R(U,F)$
- 输出：到BCNF且具有无损连接性的分解$\rho$
- 算法：

① 令 $\rho = \{ R(U,F) \}$

② 检查$\rho$中各关系模式是否均属于BCNF。若是，则算法终止。

③ 设$\rho$中$R_i(U_i,F_i) \notin BCNF$，那么必有 $X \rightarrow A \in F_i^+$ $(A \notin X)$，且$X$非$R_i$的候选码。对$R_i(U_i,F_i)$进行如下分解：

- $\sigma = \{S_1,S_2\}$，$U_{S_1} = XA$，$U_{S_2} = U_i - A$
- 以 $\sigma$ 代替 $\rho$ 中的 $R_i(U_i,F_i)$
- 返回步骤②。

- **[引理6.5]**
  - 若$\rho = \{R_1(U_1, F_1), R_2(U_2, F_2), \cdots, R_k(U_k, F_k)\}$是$R(U, F)$ 的一个无损连接分解
  - $\boldsymbol{\sigma} = \{\boldsymbol{S}_1, \boldsymbol{S}_2, \cdots, \boldsymbol{S}_{\boldsymbol{m}}\}$是$\rho$中$R_{\boldsymbol{i}}(U_i, F_i)$的一个无损连接分解
  - 那么$\rho' = \{R_1, \cdots, R_{\boldsymbol{i-1}}, \boldsymbol{S}_1, \boldsymbol{S}_2, \cdots, \boldsymbol{S}_{\boldsymbol{m}}, R_{\boldsymbol{i+1}}, \cdots, R_k\}$也是$R(U, F)$的无损连接分解。

- **[引理6.6]** $(\boldsymbol{R}_1 \bowtie \boldsymbol{R}_2) \bowtie \boldsymbol{R}_3 = \boldsymbol{R}_1 \bowtie (\boldsymbol{R}_2 \bowtie \boldsymbol{R}_3)$

# 关系规范化设计综合示例（1）

emps(emp_id, emp_name, emp_phone, dept_name, emp_cityst, emp_straddr, emp_zip)

- **职工emps关系中各属性的语义如下：**
  - **每一位员工只有一个姓名emp_name、电话emp_phone和通讯地址，通讯地址由城市emp_cityst、街道地址emp_straddr、邮政编码emp_zip组成；**
  - **emp_cityst属性值具体到省(州)和城市名称，确保每一个emp_cityst值对应着唯一的一座城市，以解决不同地区的城市同名问题；**
  - **在不同城市中，可能存在同名的街道；同一条街道上的不同区域，可能对应不同的邮政编码；当城市和街道地址确定时，对应的邮政编码唯一确定；**
  - **在同一座城市中，不同的区域使用不同的邮政编码；每一个邮政编码，只能对应着唯一一座城市中的某个区域；**
  - **每一位员工只就职于一个部门dept_name。**

# 关系规范化设计综合示例（2）

emps(emp_id, emp_name, emp_phone, dept_name, emp_cityst, emp_straddr, emp_zip)

- **根据对上述语义信息的分析，初步得到该关系上的函数依赖集如下：**
  ① emp_id→{emp_name, emp_phone, dept_name, emp_cityst, emp_straddr, emp_zip}
  ② {emp_cityst, emp_straddr}→emp_zip
  ③ emp_zip→emp_cityst

- **按照最小覆盖的要求进行检查。在函数依赖①中，emp_cityst 和 emp_zip 只要保留一个即可。**

- **保留emp_cityst，计算得到的函数依赖集最小覆盖如下：**
  ① emp_id→{emp_name, emp_phone, dept_name, emp_cityst, emp_straddr}
  ② {emp_cityst, emp_straddr}→emp_zip
  ③ emp_zip→emp_cityst

# 关系规范化设计综合示例（3）

emps(emp_id, emp_name, emp_phone, dept_name, emp_cityst, emp_straddr, emp_zip)

- **保留emp_cityst，计算得到的函数依赖集最小覆盖如下：**
  ① emp_id→{emp_name, emp_phone, dept_name, emp_cityst, emp_straddr}
  ② {emp_cityst, emp_straddr}→emp_zip
  ③ emp_zip→emp_cityst

- **对emps关系进行规范化设计检查，发现** emps∉**3**NF **and** emps∉BCNF**.**
- **理由如下：**
  - **候选码：emp_id**
  - **函数依赖⑤和⑥既不符合3NF定义，也不符合BCNF定义。**

# 关系规范化设计综合示例（4）

emps(emp_id, emp_name, emp_phone, dept_name, emp_cityst, emp_straddr, emp_zip)

- **首先将关系emps分解到满足3NF，且分解满足‘无损联接’和‘依赖保持’：**
  - emps(emp_id, emp_name, emp_phone, dept_name, emp_cityst, emp_straddr)

    ① emp_id→{emp_name, emp_phone, dept_name, emp_cityst, emp_straddr}

  - empadds(emp_cityst, emp_straddr, emp_zip)

    ② {emp_cityst, emp_straddr}→emp_zip

    ③ emp_zip→emp_cityst

# 关系规范化设计综合示例（5）

- emps(emp_id, emp_name, emp_phone, dept_name, emp_cityst, emp_straddr)
  ① emp_id→{emp_name, emp_phone, dept_name, emp_cityst, emp_straddr}
- empadds(emp_cityst, emp_straddr, emp_zip)
  ② {emp_cityst, emp_straddr}→emp_zip
  ③ emp_zip→emp_cityst

- **对分解后的两个子关系，分别用3NF和BCNF的定义进行检查，发现：**
  - emps ∈ 3NF **and** emps ∈ BCNF
  - empadds ∈ 3NF **and** empadds ∉ BCNF

# 关系规范化设计综合示例（6）

- empadds(emp_cityst, emp_straddr, emp_zip)
  ② {emp_cityst, emp_straddr}→emp_zip
  ③ emp_zip→emp_cityst

- 对关系 empadds 作出判断的理由如下：
  - 候选码：{emp_cityst, emp_straddr} 和 {emp_zip, emp_straddr}
  - 函数依赖符合 3NF 的定义（不存在 *非主属性* ）
  - 函数依赖 ③ 不符合BCNF定义。

- 可继续将empadds分解成为如下的两个子关系，结果关系满足BCNF，分解满足‘无损联接’，但不满足‘保持函数依赖’：
  - zipstr(emp_zip, emp_straddr)
  - zipcit(emp_zip, emp_cityst)
    ③ emp_zip→emp_cityst

# 复习思考题（8）

14. 给定关系模式R(A,B,C,D,E,F) 及其上的函数依赖集：

$S = \{ A \to E,\ B \to ADE,\ DF \to AC,\ ADF \to B \}$

① 请直接写出与S等价的最小函数依赖集；

② 请直接写出关系模式R的所有候选码、主属性集、非主属性集；

③ 请使用3NF模式分解算法对关系R进行分解，并满足无损联接性和依赖保持性；

④ 上题分解结果是否满足BCNF？如果不满足，请将其继续分解到满足BCNF并说明理由。

15. 给定关系R(A,B,C,D,E,F,G)及其上的函数依赖集：(不需要写计算过程)

$M = \{ ABC \to DEF,\ AC \to BG,\ D \to F,\ E \to GC \}$

① 请直接写出与M等价的最小函数依赖集；

② 请直接写出关系R的所有候选码、主属性集、非主属性集；

③ 请使用3NF模式分解算法对关系R进行分解，并满足无损联接性和依赖保持性；

④ 上述的分解结果是否满足BCNF？如满足BCNF，请简单说明理由；否则，请将其继续分解到满足BCNF。

# 复习思考题（9）

**16.** 设有一个机场跑道使用调度管理系统，其关系模式如下：

| | 机场编号 | 跑道编号 | 飞机编号 | 使用开始时间 | 使用结束时间 |
|---|---|---|---|---|---|
| R | ( ano, | lno, | pno, | s_time, | e_time ) |

其中：

- 机场编号ano和飞机编号pno分别是机场和飞机的标识属性；
- 在一个机场中，可能有多条用于飞机起降的跑道，每条跑道都有一个唯一的编号lno；分属于不同机场的跑道，可能有相同的跑道编号；
- 每一条跑道每次只能供一架飞机使用（供飞机起降用）；
- 使用开始时间和使用结束时间的数据类型是时间戳(timestamp)。

① 根据上述描述，请写出关系R上的最小函数依赖集。(不需要写计算过程)

② 关系R最高能够满足哪个范式的定义？请简单说明理由。

# 复习思考题（10）

**17.** 设有一个大学生创新项目管理关系P，其属性包括：项目的编号pno、执行年份pyear、验收等级ps，项目负责学生的学号mgrno，项目参与学生的学号sno。

其中：①项目编号和学号分别是项目和学生的标识属性；②每个项目都有唯一的一名负责的学生，以及可能的若干名参与的学生；③每个项目的执行周期只有一年；④一个学生可以负责或参与过若干个项目，但每一年最多只能负责或参与一个项目。

① 请写出该关系上的最小函数依赖集（直接写出结果）

② 该关系最高能够满足到第几范式？请简单说明理由；

③ 关系P是否满足3NF？如不满足，请用到3NF的模式分解算法直接对其进行模式分解；

④ 上述分解结果是否满足BCNF？如不满足，请将其进一步分解到满足BCNF。