7 normalization规范化

- 数据依赖：a->b (读作:a蕴涵b)
  - 意义：当任意两个元组在属性集a上相等时，则它们在属性集b上也相等。即同一个a（的值），必然对应同一个b（的值）
  - 若a不是关系的码，则该组数据依赖是不良的---------->分解
- 有没有坏的数据依赖？--->正确的分解成几个小的好关系----->重复
- “坏”的关系通过分解成为几个更小但是“好”的关系，以消除不良的函数依赖。但不正确的分解会带来信息丢失（所有小表合并之后相比于原表存在信息缺失）
- 无损分解（lossless decomposition）
  - Formally,

    $\Pi_{R_1}(r) \bowtie \Pi_{R_2}(r) = r$ （做自然连接【笛卡尔积】等于原来的表）
  - And, conversely a decomposition is lossy if

    $r \subset \Pi_{R_1}(r) \bowtie \Pi_{R_2}(r) = r$
- 函数依赖
  - 定义

    我们说关系R上存在以下函数依赖

    $\alpha \rightarrow \beta$ （读作：α蕴涵β ， 或β依赖α）

    的条件是当且仅当：任意两个元组t1, t2如果在属性集α上相等，它们在属性集β上必然相等

    （ 同一个α对应同一个β）

    $t_1[\alpha] = t_2[\alpha] \Rightarrow t_1[\beta] = t_2[\beta]$
  - 使用函数依赖
    - 测试关系以查看它们在给定的功能依赖集下是否合法。如果关系 r 在函数依赖集 F 下是合法的，我们说 r 满足 F。
    - 指定对法律关系集的约束。如果 R 上的所有法律关系都满足函数依赖集 F，我们说 F 对 R 成立。
  - 如果一个关系的所有实例都满足一个函数依赖，那么它是平凡的。
    - 如果b是a的子集，那么a->b是平凡的
- 闭包：F逻辑上隐含的所有函数依赖项的集合是F的闭包。
  - 测试是否是超码？a的闭包是否包含模式的所有属性？
  - 测试函数依赖，a->b？b是否在a的闭包里？
  - 无关的：删掉这个依赖也能推出闭包

    - 正则覆盖：闭包 F_c，可以推出F的闭包，左侧都是唯一的，没有无关项
- 码
    - 超码：K是关系R的超码当且仅当K->R
    - 候选码：若关系中的一个属性或属性组的值能够唯一地标识一个元组，且他的真子集不能唯一的标识一个元组，则称这个属性或属性组做候选码。K是关系R的候选码：

      $K \rightarrow R$, and

      for no $\alpha \subset K$, $\alpha \rightarrow R$
- 正确的分解方案：
    - 依赖关系保存：如果分解之后各个模式的闭包的并集等于原模式的闭包，那么依赖关系保存了----------指数级运算时间
    - 无损分解
- 范式normal form：满足一定要求的所有关系的全集
    - 概念
        - 高级范式比低级范式更好：第一范式、第二范式、第三范式、BC范式、第四范式。高级范式是低级范式的子集
        - 不同范式之间的联系4NF 包含于 BCNF 包含于 3NF 包含于 2NF 包含于 1NF
        - 范式级别(从低级到高级)第一范式(1NF)、第二范式 (2NF)、第三范式 (3NF)、BC范式(BCNF)第四范式* (4NF)……范式越高级，代表的关系就越“好”，要满足的要求也就越高
        - 高级范式是低级范式的子集。范式越高，满足的要求也就越高；满足高要求的关系肯定能够满足低要求，所以高级范式中的关系肯定也在低级范式中
    - 第一范式：关系的每个属性都是原子的。【非原子，属性可以再分复合属性、多值属性】
        - 一些术语和解释
            - 码属性（主属性）：一个属性，which出现在**候选码**中
            - 非码属性（非主属性）：一个属性，不出现在任何**候选码**中
            - （候选码）码的一部分：候选码的真子集
            - 超码
            - 非超码：超码以外，不具有唯一性的属性集
    - 第二范式：第一范式，**每个非码属性**都完全函数依赖于候选码
        - 判断是否属于第二范式：检查每个非码属性，是否可能依赖于候选码的一部分？有，则不属于【首先：列出所有函数依赖，之后看看非码属性能否有候选码的一部分推出。再通俗一点：候选码的子集能不能推出不是候选码的属性，能推出则不属于】
    - 第三范式：第一范式，**每个非码属性**都非传递依赖与候选码
        - 判断是否属于第三范式：检查每个非码属性，看是否只依赖于超码。如果有依赖于非超码的则不属于

    - 对关系的闭包中所有a->b都有以下至少一项成立
        - a->b是一个平凡的函数依赖
        - a是超码
        - b-a的每个属性都包含与R的一个候选码中
    - 若一个关系没有非码属性，那么它一定可以是第三范式
    - 可能需要空值
    - 有信息重复问题
- BCN范式
    - 对F的闭包中**所有**形如a->b的函数依赖，有至少一项成立：
        - a->b是一个平凡的函数依赖，也即b是a的子集
        - a是超码
- 总结：
    - 先列出关系的闭包，即所有的函数依赖；检查每一个函数依赖a->b
    - 如若a是非码---------->不是BCNF
    - a是非码，b有非码属性-------->不是第三范式
    - a是候选码的真子集，b有非码属性------->不是第二范式
- 证明候选码时，既要证明他们的闭包是全部，也要证明他们的任意真子集的闭包不是全部

    $R = (A, B, C, G, H, I)$

    $F = \{A \rightarrow B$
    $A \rightarrow C$
    $CG \rightarrow H$
    $CG \rightarrow I$
    $B \rightarrow H\}$

    $(AG)^+$

    1. $result = AG$
    2. $result = ABCG$ $(A \rightarrow C$ and $A \rightarrow B)$
    3. $result = ABCGH$ $(CG \rightarrow H$ and $CG \subseteq AGBC)$
    4. $result = ABCGHI$ $(CG \rightarrow I$ and $CG \subseteq AGBCH)$

    AG是候选码么？

    1. Is AG a super key?
        1. Does $AG \rightarrow R$? == Is $R \supseteq (AG)^+$
    2. Is any subset of AG a superkey?
        1. Does $A \rightarrow R$? == Is $R \supseteq (A)^+$
        2. Does $G \rightarrow R$? == Is $R \supseteq (G)^+$
        3. In general: check for each subset of size $n-1$
- 如何让关系达到更高的范式？
    - 分解：坏关系分解为属于高级范式的好关系
    - 无损分解+规范化
- 将模式分解为BCNF
    - 找到违反BCNF的那个函数依赖a->b（非平凡的函数依赖，a不是超码），将其分解为

        - $(\alpha \cup \beta)$
        - $(R-(\beta-\alpha))$
    - 交集是表的主键说明信息无损，依赖保留
- 函数依赖理论
    - 函数依赖集的闭包
        - 自反率：如果b是a的子集，那么a->b
        - 增广率：如果a->b，那么 ra -> rb
        - 传递率：如果a->b 并且 b-> r 那么 a->r
        - 合并规则：如果a->b，a->r，那么a->br
        - 分解规则：如果a->br，那么a->b a->r
        - 伪传递性规则：如果a->b rb->c，那么ar->c
    - 属性集的闭包

```
result := α;
    repeat
        for each 函数依赖 β→r in F do
            begin
                if β⊆ result then result := result ∪ γ;
            end
    until (result 不变)
```

图 8-8　计算 F 下 α 的闭包 α^+ 的算法

        - 先将要计算的属性放到result中
        - 对函数依赖集中的每个依赖b->r循环，若b属于result，则r也可以加入result
        - 用途：判断属性及是否为超码；计算依赖集的闭包；检查a->b的函数依赖是否成立
    - 正则覆盖
        - 无关属性：去除函数依赖中的一个属性，依赖集的闭包不会被改变
        - 正则覆盖：不含无关属性，左半部分唯一
    - 无损分解
        - R被分解为R_1和R_2，若R_1和R_2的交际是R_1或者R_2的超码，那该分解就是无损分解。
    - 保持依赖
        - F在R_i上的限定是F闭包中所有只包含R_i中属性的函数依赖的集合F_i
        - 判定保持依赖的算法，对F中的每个a->b计算上面的过程

```
result = α
repeat
    for each 分解后的 Ri
        t = (result ∩ Ri)+ ∩ Ri
        result = result ∪ t
    until (result 没有变化)
```

- BCNF分解
  - BCNF的判定
    - 对非平凡依赖a->b，计算a的闭包，看a是不是超码
    - 仅检查F中各项函数依赖是否违反了BCNF，但是若分解之后就就不行了
    - 为检查R分解之后的R_i是否满足BCNF，对于R_i中属性的每个子集a，确保a的闭包要么不包含$R_i - a$的任何属性，要么包含他的所有属性。
  - BCNF分解
    -

```
result := {R };
done := false;
计算 F +;
while (not done) do
  if (有模式 Ri in result that 不是 BCNF)
    then begin
          让 α → β 是一个Ri 上成立的非平凡函数依赖
          ，满足 α → Ri 不属于 F +, 并且 α ∩ β = ∅;
            result := (result – Ri) ∪ (Ri – β) ∪ (α, β);
          end
    else done := true;
```

以上内容整理于 幕布文档