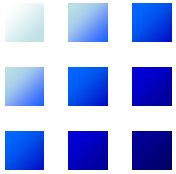

データベース
【7：リレーショナルデータベース設計論（2）】

石川　佳治

関数従属性

関数従属性(1):定義

- リレーションスキーマ $R(\ldots, X, \ldots, Y, \ldots)$
  - X, Y は属性集合
- R の任意のインスタンス中の任意の2タプルにおいて，X の値が等しいなら Y の値も必ず等しいという制約が成り立つとき，関数従属性(functional dependency) $X \rightarrow Y$ が成立
  - X は Y を関数的に決定する
  - Y は X に関数従属する
- 例:営業(商品番号，顧客番号，社員番号，販売価格)
  - 商品番号，顧客番号→販売価格
  - 顧客番号→社員番号，など

関数従属性(2):形式的な定義

- $X \to Y$ の形式的な定義
  - リレーションスキーマ RS と属性集合 $X, Y \subseteq RS$ が与えられたとき，R の任意のインスタンスにおいて以下が成立

$$(\forall t \in R)(\forall u \in R)(t[X] = u[X] \to t[Y] = u[Y])$$

- 関数従属性は整合性制約の一つ
- リレーショナルデータベース設計論で中心的な役割を果たす

関数従属性(3):注意

- 注意:ある一つのインスタンスから関数従属性を判断することは必ずしもできない

学生番号	氏名	専攻	住所	年齢
00001	山田一郎	情報工学	東京都	20
00002	鈴木明	情報工学	茨城県	20
00003	佐藤花子	知識工学	京都府	21

  - 年齢→住所が成立しないことは明らか
  - 住所→専攻はこのリレーションでは「たまたま」成立
- どのような関数従属性が成立するかは，対象世界をモデリングする設計者の背景知識と分析により決定

超キーと候補キー（3.2節の再定義）

- リレーションスキーマ RS において属性集合 X が超キー（superkey）であるとは，

  ① $X \rightarrow RS$

  が成立すること
- ①に加えて

  ② X のいかなる真部分集合 Y についても $Y \rightarrow RS$ は成立しない

  が成立するとき X は候補キー（candidate key）

論理的含意

- リレーションスキーマ $\{A, B, C\}$ において，関数従属性 $A \to B$, $B \to C$ が成立するなら，$A \to C$ も成立
- $\{A \to B, B \to C\}$ は $A \to C$ を論理的に含意（logically imply）するといい，

$$\{A \to B,\ B \to C\} \models A \to C$$

と記す

- 例：{学生番号→住所，住所→郵便番号} ⊨ 学生番号→郵便番号 が成立

関数従属性の閉包

- 関数従属性の集合 F が与えられたとき，
  - F の要素の関数従属性
  - F が論理的に含意する関数従属性

  をすべて集めた集合を F の閉包（closure）と呼び，F^+ で表す

$$F^+ = \{X \rightarrow Y \mid F \models X \rightarrow Y\}$$

- 与えられた関数従属性の集合 F に対し，F^+ をどう求めるか?
  - アームストロングの公理系

アームストロングの公理系(1):基本的な規則

- すべての関数従属性を列挙する規則
- 三つの規則(公理)
  - ① 反射律(reflexivity law):自明(trivial)な従属性

    $Y \subseteq X \Rightarrow X \rightarrow Y$

    - 例:氏名,住所→氏名
  - ② 増加律(augmentation law)

    $X \rightarrow Y \Rightarrow XZ \rightarrow YZ \ \ (X \cup Z \rightarrow Y \cup Z)$

    - 例:住所→郵便番号 ⇒ 住所,年齢→郵便番号,年齢
  - ③ 推移律(transitivity law)

    $X \rightarrow Y$ かつ $Y \rightarrow Z \Rightarrow X \rightarrow Z$

アームストロングの公理系(2):適用例

- 具体例:$R(A, B, C)$
- $F = \{A \to B, B \to C\}$

大量の関数従属性が導かれる!

  - 自明な関数従属性
    - $A \to A, B \to B, C \to C$
    - $AB \to AB, AB \to A, AB \to B, AC \to AC, AC \to A, AC \to C, BC \to BC, BC \to B, BC \to C$
    - $ABC \to ABC, ABC \to AB, ABC \to AC, ABC \to BC, ABC \to A, ABC \to B, ABC \to C$
  - F 中の関数従属性への増加律適用で得られるもの(自明なもの除く)
    - $A \to AB, AC \to BC, AC \to ABC, AB \to AC, B \to BC, AB \to ABC$
  - 推移律から得られるもの
    - $A \to C$
  - これらの組合せで得られるもの
    - $A \to AC, A \to BC, AB \to BC$

アームストロングの公理系(3):性質

- アームストロングの公理系は健全(sound)
  - 関数従属性集合 F が与えられたとき，公理系を用いて導出されるすべての関数従属性は F^+ の要素である(つまり F が論理的に含意するものである)
- アームストロングの公理系は完全(complete)
  - F^+ 中のすべての関数従属性が公理系の規則を適用することで導出可能である
- すなわち，F をもとに F^+ に属する関数従属性のみをすべて導出できる

アームストロングの公理系(4):他の規則

- アームストロングの公理系から以下の規則も導出可能
  - ① 合併律(union law)
    $X \to Y$ かつ $X \to Z$ のとき $X \to YZ$
    - 例:学籍番号→氏名,学籍番号→住所なら学籍番号→氏名,住所
  - ② 擬推移律(pseudotransitivity law)
    $X \to Y$ かつ $WY \to Z$ のとき $X\ W \to Z$
    - 例:品名→卸業者,地区名,卸業者→営業所のとき,品名,地区名→営業所
  - ③ 分解律(decomposition law)
    $X \to Y$ かつ $Z \subseteq Y \Rightarrow X \to Z$

F に関する X の閉包(1):定義・アルゴリズム

- 関数従属性集合 F のもとで，ある属性集合 X により関数的に決定される属性の全体

$$X^+ = \{A \mid A \in RS \wedge X \rightarrow A \in F^+\}$$

- X^+ を求めるアルゴリズム

Result := *X*
repeat
  for each $Y \rightarrow Z \in F$ **do**
    if $Y \subseteq$ *Result* **then**
      Result := *Result* $\cup$ *Z*
until *Result* が変化せず

F に関する X の閉包(2):例

- 例:$R(A, B, C, D, E, G)$
  - $F = \{A \rightarrow B, A \rightarrow G, B \rightarrow C, C \rightarrow D, BC \rightarrow E, CG \rightarrow A\}$
- $\{B\}^+$ を求める例
  - 初期状態 $Result := \{B\}$
  - 繰り返し
    - $B \rightarrow C \in F$ について:$\{B\} \subseteq Result$ なので $Result := Result \cup \{C\} = \{B, C\}$
    - $C \rightarrow D \in F$ について: $\{C\} \subseteq Result$ なので $Result := Result \cup \{D\} = \{B, C , D\}$
    - $BC \rightarrow E \in F$ について:$\{B, C\} \subseteq Result$ なので $Result := Result \cup \{E\} = \{B, C, D, E\}$
- 他の例:$\{B, G\}^+ = \{A, B, C, D, E, G\}$

F に関する X の閉包(3):計算量

- X^+ の導出処理
  - RS および F の要素数に関する多項式オーダの時間で計算可能
- 比較:F^+ の導出処理
  - 膨大な計算量:F^+ の要素数は RS の要素数の指数関数のオーダの計算量
- X^+ の計算で対応可能である処理については,F^+ を求めず X^+ を求めることで対応することで効率的処理を実現

関数従属性集合の等価性(1):定義・判定法

- 関数従属性集合 F と G が**等価**(equivalent)
  - $F^+ = G^+$ のときをいう: F と G は本質的には同じ整合性制約を表現
  - $F = G$ でなくともよい
- 等価性の判定方法
  - 直接的な方法:$F^+ \subseteq G^+$ かつ $F^+ \supseteq G^+$ が成立するかを判定
    - F^+, G^+を求める計算量大
  - 効率的な方法: $F^+ \subseteq G^+$を判定する場合($F^+ \supseteq G^+$も同様)
    - F 中の各関数従属性 $X \rightarrow Y$ について, G に関する X の閉包 X^+ を求め, $Y \subseteq X^+$ が成り立つかを見る

関数従属性集合の等価性(2):判定の例

- 例(p. 64):$R(A, B, C)$, $F = \{A \rightarrow C, B \rightarrow C, C \rightarrow B\}$, $G = \{A \rightarrow B, B \rightarrow C, C \rightarrow B\}$
- $F^+ \subseteq G^+$ の判定
  - $A \rightarrow C \in F$ について:G に関する A の閉包は $A^+ = \{A, B, C\}$ で, $C \in A^+$ が成立
  - $B \rightarrow C \in F$ について:G に関する B の閉包は $B^+ = \{B, C\}$ で, $C \in B^+$ が成立
  - $C \rightarrow B \in F$ について:G に関する C の閉包は $C^+ = \{B, C\}$ で, $B \in C^+$ が成立
  - 以上より $F^+ \subseteq G^+$ が成立
- $F^+ \supseteq G^+$ も成立, よって $F^+ = G^+$

極小被覆(1):定義

- 極小被覆(minimal cover):ある従属性集合に等価な従属性集合のうちで最も簡潔なもの
- 定義:従属性集合 F に対する極小被覆 M
  - ① M 中のすべての関数従属性の右辺は単一の属性
  - ② M 中のいかなる関数従属性 $X \rightarrow A$ に対しても, $M - \{X \rightarrow A\}$ と M は等価でない($X \rightarrow A$ は必須)
  - ③ M 中のいかなる関数従属性 $X \rightarrow A$ および $Z \subset X$ に対しても, $M - \{X \rightarrow A\} \cup \{Z \rightarrow A\}$ と M は等価でない
- 注意:極小被覆は複数個存在しうる

極小被覆(2):導出アルゴリズム

極小被覆を求めるアルゴリズム

Result := *F*

// 条件①を満たすように変形
for each $X \rightarrow \{A_1, \ldots, A_n\}$ $(n \geq 2) \in$ *Result* **do**
  Result := (*Result* – $\{X \rightarrow \{A_1, \ldots, A_n\}\}$) $\cup$ $\{X \rightarrow A_1, \ldots, X \rightarrow A_n\}$

// 条件②, ③への対応
while 条件②または③に違反する $X \rightarrow A$ が *Result* 中にあり **do**
  if 条件②に違反 **then**
    Result := *Result* – $\{X \rightarrow A\}$
  else if 条件③に違反 **then**
    Result := (*Result* – $\{X \rightarrow A\}$) $\cup$ $\{Z \rightarrow A\}$

極小被覆(3):具体例

- リレーションスキーマ (A, B, C)
- $F = \{A \to BC, B \to C, C \to B\}$
- 導出例1
  1. 条件①を満たすように修正:$\{A \to B, A \to C, B \to C, C \to B\}$
  2. $A \to B$ は他から導出でき必須でないので削除:極小被覆 $\{A \to C, B \to C, C \to B\}$ を得る
- 導出例2
  1. ステップ1は同上
  2. $A \to C$ は他から導出でき必須でないので削除:極小被覆 $\{A \to B, B \to C, C \to B\}$ を得る