Functional Dependency

Functional Dependencies

- *Constraint* pada himpunan *legal relation*
- Mensyaratkan nilai untuk himpunan atribut tertentu menentukan nilai himpunan atribut lainnya secara unik
- Merupakan generalisasi konsep *key*

Functional Dependencies (Cont.)

- Misalkan R adalah skema relasi, di mana

  $\alpha \subseteq R \text{ and } \beta \subseteq R$

- Functional dependency $\alpha \rightarrow \beta$

  pada skema R jika dan hanya jika untuk legal relations r(R), kapanpun terdapat dua tuple t_1 dan t_2 pada r yang mempunyai nilai α sama, juga mempunyai nilai yang sama untuk β :

  $t_1[\alpha] = t_2[\alpha] \Rightarrow t_1[\beta] = t_2[\beta]$

- α secara fungsionalitas menentukan β, dan β secara fungsionalitas tergantung pada α

Functional Dependencies (Cont.)

- K adalah sebuah superkey untuk skema relasi R jika dan hanya jika K $\rightarrow$ R
- K adalah sebuah candidate key untuk R jika dan hanya jika
  - K $\rightarrow$ R, dan
  - Tidak ada $\alpha \subset K$ di mana $\alpha \rightarrow$ R
- Dengan functional dependency, dimungkinkan mengekspresikan constraint yang tidak dapat diekspresikan menggunakan superkey.

Kegunaan Functional Dependency

- Functional dependency digunakan untuk :
  - Menguji relasi apakah legal berdasarkan himpunan functional dependency tertentu
    - Jika sebuah relasi r legal berdasar himpunan functional dependency F, kondisi ini disebut sebagai r satisfies F
  - Menyatakan constraint pada himpunan legal relation
    - Kondisi di mana semua legal relation dari R satisfy himpunan functional dependency F disebut sebagai : F holds on R

Functional Dependency (lanj.)

- Sebuah functional dependency disebut trivial apabila FD tersebut dipenuhi oleh semua instan relasi
  - Contoh :
    - Customer-name, loan-number $\rightarrow$ customer-name
    - Customer-name $\rightarrow$ customer-name
  - Secara umum, $\alpha \rightarrow \beta$ trivial apabila $\beta \subseteq \alpha$

Closure Himpunan Functional Dependency

- Dalam sebuah himpunan functional dependency F, mungkin terdapat functional dependency lainnya yang didapat dari implikasi logis F
  - Contoh : jika $A \rightarrow B$ dan $B \rightarrow C$, kita dapat menyimpulkan bahwa $A \rightarrow C$
- Himpunan semua functional dependency yang diperoleh dari implikasi logis F disebut sebagai closure dari F
- Notasi closure F adalah F^+
- Kita dapat memperoleh semua anggota F^+ dengan menerapkan Armstrong's Axioms :
  - if $\beta \subseteq \alpha$, then $\alpha \rightarrow \beta$ **(reflexivity)**
  - if $\alpha \rightarrow \beta$, then $\gamma\alpha \rightarrow \gamma\beta$ **(augmentation)**
  - if $\alpha \rightarrow \beta$, and $\beta \rightarrow \gamma$, then $\alpha \rightarrow \gamma$ **(transitivity)**
- Rule tersebut :
  - Sound (men-generate hanya functional dependency yang *actually hold*), dan
  - Complete (men-generate semua functional dependency yang *hold*)

Contoh

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

  $F = \{A \rightarrow B,\ A \rightarrow C,\ CG \rightarrow H,\ CG \rightarrow I,\ B \rightarrow H\}$
- Beberapa anggota F^+ :
  - $A \rightarrow H$
    - Dengan menerapkan rule transitivity dari $A \rightarrow B$ dan $B \rightarrow H$
  - $AG \rightarrow I$
    - Dengan menerapkan rule augmentation pada $A \rightarrow C$ berupa penambahan G, sehingga didapat $AG \rightarrow CG$, kemudian menerapkan transitivity dengan $CG \rightarrow I$
  - $CG \rightarrow HI$
    - Didapat dari $CG \rightarrow H$ dan $CG \rightarrow I$ (union rule). Union rule diperoleh dari
      - Definisi functional dependency
      - Augmentation pada $CG \rightarrow I$ untuk mendapat $CG \rightarrow CGI$, augmentation $CG \rightarrow H$ untuk mendapat $CGI \rightarrow HI$, dan kemudian dilakukan transitivity

Closure Functional Dependency (lanj.)

- Terdapat beberapa rule tambahan untuk mempermudah penghitungan F^+ :

- If $\alpha \rightarrow \beta$ holds and $\alpha \rightarrow \gamma$ holds, then $\alpha \rightarrow \beta\gamma$ holds **(union)**
- If $\alpha \rightarrow \beta\gamma$ holds, then $\alpha \rightarrow \beta$ holds and $\alpha \rightarrow \gamma$ holds **(decomposition)**
- If $\alpha \rightarrow \beta$ holds and $\gamma\beta \rightarrow \delta$ holds, then $\alpha\gamma \rightarrow \delta$ holds **(pseudotransitivity)**

Rule di atas diperoleh dari Armstrong’s Axioms

Canonical Cover

- Himpunan functional dependency dapat mempunyai redundant dependencies yang dapat disimpulkan dari functional dependency lainnya
  - Contoh : $A \to C$ redundan dalam : $\{A \to B, B \to C, A \to C\}$
  - Bagian dari functional dependency mungkin redundan
    - Contoh pada RHS : $\{A \to B, B \to C, A \to CD\}$ dapat disederhanakan menjadi $\{A \to B, B \to C, A \to D\}$
    - Contoh pada LHS : $\{A \to B, B \to C, AC \to D\}$ dapat disederhanakan menjadi $\{A \to B, B \to C, A \to D\}$
- Secara intuitive, canonical cover dari F adalah himpunan "minimal" functional dependency yang equivalent dengan F, tidak mempunyai redundant dependencies atau bagian dependency yang redundan

Extranous Attributtes

- Misal ada sebuah himpunan *functional dependency* F dan sebuah *functional dependency* $\alpha \rightarrow \beta$ dalam F
  - Atribut A *extranous* dalam α jika A ε α dan F mengimplikasikan (F-{α->β})U{(α -A)-> β}
  - Atribut A *extranous* dalam β jika A ε β dan himpunan *functional dependency* (F-{α->β})U{α ->(β-A)} mengimplikasikan F
- Catatan : implikasi pada arah yang berlawanan bersifat trivial, karena functional dependency yang lebih kuat selalu mengimplikasikan yang lebih lemah
- Contoh ; terdapat F = {A->C, AB->C}
  - B extranous dalam AB->C karena {A->C, AB->C} mengimplikasikan A->C secara logis
- Contoh : terdapat F = {A->C, AB->CD}
  - C extranous dalam AB->CD karena AB->C dapat disimpulkan walaupun C dihapus

Testing Bila Sebuah Atribut *Extranous*

- Terdapat sebuah himpunan *functional dependency* F dan *functional dependency* $\alpha \text{->} \beta$ dalam F
- Untuk mentest apakah atribut A ε α *extranous* dalam α
  - Hitung $(\{\alpha\}\text{-}A)^+$ menggunakan *functional dependency* dalam F
  - Cek apakah $(\{\alpha\}\text{-}A)^+$ mengandung A, jika ya, maka A *extranous*
- Untuk mentest apakah atribut A ε β *extranous* dalam β
  - Hitung α+ menggunakan *dependency* dari
    $F' = (F\text{-}\{\alpha \text{->} \beta\})$ U $\{\alpha \text{->} (\beta\text{-}A)\}$
  - Cek apakah α^+ mengandung A, jika ya, maka A *extranous*

Canonical Cover

- Canonical cover untuk F adalah himpunan functional dependency Fc di mana :
  - F mengimplikasikan secara logis semua dependency dalam Fc, dan
  - Fc mengimplikasikan secara logis semua dependency dalam F, dan
  - Tidak ada functional dependency dalam Fc yang mengandung extranous attributte
  - Setiap sisi kiri functional dependency dalam Fc unik
- Untuk menghitung canonical cover F :

repeat

Use the union rule to replace any dependencies in *F*

$\alpha_1 \rightarrow \beta_1$ and $\alpha_1 \rightarrow \beta_1$ with $\alpha_1 \rightarrow \beta_1 \beta_2$

Find a functional dependency $\alpha \rightarrow \beta$ with an extraneous attribute either in α or in β

If an extraneous attribute is found, delete it from $\alpha \rightarrow \beta$

until *F* does not change

Contoh Penghitungan Canonical Cover

- R = (A,B,C)
  F = {A->BC, B->C, A->B, AB->C}
- Kombinasikan A->BC dan A->B menjadi A->BC
  - Himpunan F menjadi {A->BC, B->C, AB->C}
- A extranous dalam AB->C
  - Cek apakah dengan menghapus A dari AB->C akan mengahasilkan functional dependency yang diimplikasikan oleh functional dependency lainnya
    - Ya, karena terdapat B->C
  - Himpunan F menjadi {A->BC, B->C}
- C extranous dalam A->BC
  - Cek apakah A->C diimplikasikan oleh A->B dan functional dependency lainnya
    - Ya : dengan menerapkan transitivity pada A->B dan B->C
- Canonical cover-nya adalah : A->B, B->C