Normalisierung II

Ziele

- Verlustlosigkeit
- Abhängigkeitsbewahrung
- Algo: Dekomposition
- Algo: Basis
- Algo: Synthese

Normalisierung

Problem: Anomalien, wenn nicht “zusammenpassende” Informationen in einer Relation gebündelt sind.

Lösung: Zerlegung des Schemas in mehrere Relationenschemata.

$$\text{Aus } R \text{ wird } R_1, R_2, \ldots, R_n, \text{ mit } R_i \subseteq R,\ 1 \leq i \leq n$$

Korrektheitskriterien:

Verlustlosigkeit: Die ursprünglich in R enthaltenen Informationen müssen aus $R_1, \ldots, R_n$ rekonstruierbar sein

Abhängigkeitsbewahrend: Die für R geltenden funktionalen Abhängigkeiten müssen auf die Schemata $R_1, \ldots, R_n$ übertragbar sein.

Verlustlosigkeit

- In R enthaltene Information muss durch den natürlichen Verbund (Join) der zerlegten Relationen R_1 und R_2 rekonstruierbar sein.
- Keine existierenden Tupel dürfen verloren gehen
- Keine „neuen“ Tupel dürfen entstehen

Beispiel – Verlustlosigkeit?

Relation: Biertrinker

Kneipe	**Gast**	**Bier**
Zur runden Ecke	Alfons	Pils
Zur runden Ecke	Heinz	Hefeweizen
Brauhaus	Alfons	Hefeweizen

Zerlegung:

Relation: Besucht

Kneipe	**Gast**
Zur runden Ecke	Alfons
Zur runden Ecke	Heinz
Brauhaus	Alfons

Relation: Trinkt

Gast	**Bier**
Alfons	Pils
Heinz	Hefeweizen
Alfons	Hefeweizen

Beispiel – Keine Verlustlosigkeit.

Relation: Biertrinker

Kneipe	**Gast**	**Bier**
Zur runden Ecke	Alfons	Pils
Zur runden Ecke	Heinz	Hefeweizen
Brauhaus	Alfons	Hefeweizen
Zur runden Ecke	*Alfons*	*Hefeweizen*
Brauhaus	*Alfons*	*Pils*

Funktionale Abhängigkeiten:
FD1 = {Kneipe, Gast $\rightarrow$ Bier}
nicht:
FD2 = {Gast $\rightarrow$ Bier}
FD2 = {Gast $\rightarrow$ Kneipe}

Kreuzprodukt:

Relation: Besucht

Kneipe	**Gast**
Zur runden Ecke	Alfons
Zur runden Ecke	Heinz
Brauhaus	Alfons

$\bowtie$

Relation: Trinkt

Gast	**Bier**
Alfons	Pils
Heinz	Hefeweizen
Alfons	Hefeweizen

Abhängigkeitsbewahrung

- Prüfen aller funktionalen Abhängigkeiten lokal, auf den entstehenden Relationen R_1 bis R_n
  - Entstehende Relationen lokal konsistent
  - Parallele Einfügeoperationen in zerlegte Relationen können globale Konsistenzbedingen verletzen

Relation 1: PLZverzeichnis (<u>Strasse</u>, <u>Ort</u>, <u>BLand</u>, PLZ)

Funktionale Abhängigkeiten:
FD1 = {PLZ → Ort, BLand}
FD2 = {Straße, Ort, BLand → PLZ}

Relation 1.1: Strassen (<u>PLZ</u>, <u>Strasse</u>)

Relation 1.2: Orte (<u>PLZ</u>, Ort, BLand)

Algorithmen

Dekomposition: Verlustlose Zerlegung eines Relationenschmas R in Teilrelationen, die in BCNF sind. Die Abhängigkeitserhaltung ist nicht gewährleistet.

Synthese: Verlostlose und abhängigkeitserhaltende Zerlegung eines Relationenschmas R in Teilrelationen, die in 3. NF sind.

Basis: Beseitige Redundanzen in einer gegebenen Menge von FDs.

Dekomposition (Übersicht)

Sukzessive, verlustlose Zerlegung des Relationenschema R in (BCFN) Teilrelationen

Input: Menge Z = {R}   Output: Menge Z = $\{R_1 \dots R_n\}$

- Solange noch Relationenschema $R_i \in Z$ existiert, dass nicht in BCFN ist, folgendes ..

  ... es existiert also eine nicht-triviale FD ($\alpha \to \beta$) einer R_i mit:
  - $\alpha \cap \beta = \emptyset$
  - $\alpha \not\to R_i$
- Zerlege R_i in $R_j = \alpha \cup \beta$
  $R_k = R_i - \beta$
- Entferne R_i aus Z, füge R_j und R_k ein

  $Z := (Z - \{R_i\}) \cup \{R_j, R_k\}$

Dekomposition (Beispiel)

Relation 1:
Lieferanten (Name, Artikel, Anzahl, Ort, Entfernung)

FD1: Ort → Entfernung
FD2: Name → Ort
FD3: Name, Artikel → Anzahl

Z = {Lieferanten}

[Schritt 1 - FD1]
R2: Ort (<u>Ort</u>, Entfernung)
R3: Artikelmenge (<u>Name</u>, <u>Artikel</u>, Anzahl, Ort)

Z = {R2, R3}

[Schritt 2 - FD2]

R4: Kennzeichung (<u>Name</u>, Ort)

R5: Artikel (<u>Name</u>, <u>Artikel</u>, Anzahl)

[Schritt 3 – FD3] (entfällt)

Ergebnis:
Z = {R2, R4, R5}

Dekomposition (Algorithmus)

Algorithmus DEKOMPOSITION:
Input: Ein Universalschema $R = (U, F)$
Output: Verlustlose BCNF-Zerlegung $\mathbf{D} = (\mathbf{R}, .)$ von R
Methode:
begin
  $\mathbf{R} := \{R\}$;
  done := FALSE;
  while not *done* **do**
    if $(\exists R_i \in \mathbf{R})$ R_i nicht in BCNF,
      d.h. $(\exists Y \to Z \in F_i^+, Z \not\subseteq Y) Y \to X_i \notin F_i^+$
    then
      begin
        $X_{i1} := YZ$;
        $X_{i2} := X_i - Z$;
        $R_{i1} := (X_{i1}, \pi_{X_{i1}}(F_i^+))$;
        $R_{i2} := (X_{i2}, \pi_{X_{i2}}(F_i^+))$;
        $\mathbf{R} := (\mathbf{R} - \{R_i\}) \cup \{R_{i1}, R_{i2}\}$;
      end;
    else *done* := TRUE;
  end;

Basis (Übersicht)

Interesse: kleinstmögliche noch äquivalente Menge von FDs

Input: Menge F = $\{FD_1 .. FD_n\}$ Output: Eine Basis F_c von F

- $F_c \equiv F$, d.h. $F_c^+ = F^+$ (Hüllen identisch)
- F_c besitzt keine FDs ($\alpha \to \beta$), bei denen α oder β überflüssige Attribute besitzen
  - $\forall A \in \alpha : (F_c - (\alpha \to \beta) \cup ((\alpha - A) \to \beta)) \not\equiv F_c$
  - $\forall B \in \beta : (F_c - (\alpha \to \beta) \cup (\alpha \to (\beta - B)) \not\equiv F_c$
- Jede linke Seite einer FD in F_c ist einzigartig
  .. erzielt mit $(\alpha \to \gamma)$ und $(\alpha \to \beta)$ wird zu $(\alpha \to \gamma, \beta)$

Basis (Übersicht) [cont.]

Interesse: kleinstmögliche noch äquivalente Menge von FDs

Input: Menge F = $\{FD_1 .. FD_n\}$ Output: Eine Basis F_c von F

- „Linksreduktion" für jede FD $(\alpha \rightarrow \beta) \in F$
  - Prüfe $\forall A \in \alpha$, ob A *überflüssig* ist, d.h. $\beta \subseteq$ HÜLLE $(F, \alpha - A)$
  - Entferne A, d.h. $((\alpha - A) \rightarrow \beta)$
- „Rechtsreduktion" für jede FD $(\alpha \rightarrow \beta) \in F$
  - Prüfe $\forall B \in \beta$, ob B *überflüssig* ist, d.h. $B \subseteq$ HÜLLE $(F - (\alpha \rightarrow \beta) \cup (\alpha \rightarrow (\beta - B)), \alpha)$
  - Entferne B, d.h. $(\alpha \rightarrow (\beta - B))$
- Entferne $(\alpha \rightarrow \emptyset)$
- Zusammenfassen $(\alpha \rightarrow \beta)$ und $(\alpha \rightarrow \gamma)$ zu $(\alpha \rightarrow \beta, \gamma)$

Basis (Beispiel)

Relation 1: (A, B, C)

FD1: $A \rightarrow B$
FD2: $B \rightarrow C$
FD3: $A, B \rightarrow C$

Ergebnis:
$F_c = \{A \rightarrow B, B \rightarrow C\}$

[Schritt 1 - Linksreduktion]
FD3 (old): $A, B \rightarrow C$
FD3 (new): $A \rightarrow C$ // Grund: FD1, FD2

[Schritt 2 - Rechtsreduktion]
FD3 (old): $A \rightarrow C$
FD3 (new): $A \rightarrow \emptyset$ // Grund: FD1 , FD2

[Schritt 3 – Entfernen]
FD3 : $A \rightarrow \emptyset$ // Grund: überflüssig

Basis (Algorithmus)

Algorithmus BASIS:
Input: $F = \{f_1, ..., f_n\}$
Output: Eine Basis G von F.
Methode:
begin
/* mache F rechts-minimal */
  $G := \emptyset$;
  for $i := 1$ **to** n **do**
    if $f_i : X \rightarrow Y$ **and** $Y = A_1, ..., A_m$
      then $G := G \cup \{X \rightarrow A_1, ..., X \rightarrow A_m\}$;
    /* o.B.d.A. sei $G = \{f_1, ..., f_k\}$ */
/* mache G links-minimal */
  for $i := 1$ **to** k **do**
    if $f_i : X \rightarrow A$ **and** $X = B_1, ..., B_s$ **then**
    for $j := 1$ **to** s **do**
      if $FD\_MEMBERSHIP(G, X - B_j, A)$
        then $X := X - \{B_j\}$;
/* entferne redundante FDs */
  for $i := 1$ **to** k **do**
    if $FD\_MEMBERSHIP(G - \{f_i\}, L_{f_i}, R_{f_i})$
      then $G := G - \{f_i\}$;
end;

Synthese (Übersicht)

Verlustlose, abhängigkeitsbewahrende Zerlegung des Relationenschema R in (3.NF) Teilrelationen

Input: Menge Z = {R}   Output: Menge Z = $\{R_1 \ldots R_n\}$

- Bestimme die Basis F_c zu F
- Für jede funktionelle Abhängigkeit $(\alpha \rightarrow \beta) \in F_c$
  - Kreiere Relationenschema $R_a = \alpha \cup \beta$
  - Ordne R_a die FDs $FD_a := \{\alpha' \rightarrow \beta' \in F_c \mid \alpha' \cup \beta' \subseteq R_a\}$
- Wenn R bezgl. F_c einen Kandidatenschlüssel enthält – fertig, sonst Kandidatenschlüssel $k \subseteq R$ bestimmen
  - Neues Relationenschema $R_k := k$
  - $FD_k := \emptyset$
- Eliminiere Relationenschemata, die in anderen enthalten sind, d.h. $R_a \subseteq R_{a'}$

Synthese (Beispiel)

Relation 1: Universität
(Ort, BLand, Landesregierung, EW, Vorwahl,PLZ, Strasse)

FD1: Strasse, Ort, BLand → PLZ
FD2: PLZ → Ort, BLand
FD3: Ort, BLand → Vorwahl, EW
FD4: BLand → Landesregierung

[Schritt 1 - Erzeugung]

R2: PLZverzeichnis1 (<u>Strasse</u>, <u>Ort</u>, <u>BLand</u>, PLZ)
R3: PLZverzeichnis2 (Ort, BLand, <u>PLZ</u>)
R4: Städteverzeichnis (<u>Ort</u>, <u>BLand</u>, Vorwahl, EW)
R5: Regierungen (<u>BLand</u>, Landesregierung)

[Schritt 3 – Kandidaten entfällt] (entfällt)

[Schritt 2 - Eliminieren]
R3: PLZverzeichnis2 (Ort, BLand, <u>PLZ</u>)

Ergebnis:
Z = {R2, R4, R5}

Synthese (Algorithmus)

Algorithmus SYNTHESE:
Input: $R = (U, F)$
Output: Verlustlose, unabhängige 3-NF-Zerlegung **D** von R
Methode:
begin
    $G := BASIS(F)$;   /* Minimale Überdeckung */
    R$:= \emptyset$; $i := 0$;
    for each $Y \subseteq U$ mit $(\exists A \in U) Y \rightarrow A \in G$ **do**
        begin
            $i := i + 1$;
            $X_i := Y \cup \{A \in U \mid Y \rightarrow A \in G\}$
            $R_i := (X_i, \pi_{X_i}(G))$;
            R $:=$ **R** $\cup \{R_i\}$
        end;
    if $(\forall R_i \in \mathbf{R}) X_i \rightarrow U \notin G^+$ **then**
        begin
            $i := i + 1$;
            $X_i := KEY(U, G)$;
            $R_i := (X_i, \emptyset)$;
            R $:=$ **R** $\cup \{R_i\}$
        end;
    D $:= (\mathbf{R}, \emptyset)$
end;

Beispiele 9

- R (A, B, C, D, E)
  FD = {A,B,C $\rightarrow$ D,E; B,C,D $\rightarrow$ A,E; D $\rightarrow$ E}
- R (A, B, C, D, E, F)
  FD = {A,B $\rightarrow$ C,D,E,F; E $\rightarrow$ F}
- R (A, B, C, D, E)
  FD = {A,B,C $\rightarrow$ D,E; B,C,D $\rightarrow$ A,E}
- R (A, B, C, D, E)
  FD = {A,B,C $\rightarrow$ D,E; D $\rightarrow$ C; D $\rightarrow$ E}

Beispiele 10

- R (A, B, C, D, E, F)
  FD = {A,B $\rightarrow$ C,D,E,F; A $\rightarrow$ D; E $\rightarrow$ F}
- R (A, B, C, D, E, F)
  FD = {A,B $\rightarrow$ C,D,E,F; A $\rightarrow$ D; E $\rightarrow$ F; D $\rightarrow$ B}
- R (A, B, C, D, E)
  FD = {A,B,C $\rightarrow$ D,E; B,C,D $\rightarrow$ A,E}
- R (A, B, C, D, E)
  FD = {A,B,C $\rightarrow$ D,E; B,C,D $\rightarrow$ A,E; C $\rightarrow$ D}

Ziele

- Verlustlosigkeit
- Abhängigkeitsbewahrung
- Algo: Dekomposition
- Algo: Basis
- Algo: Synthese