Datenbanken Lernzettel

1 Algorithmen

1.1 Basis

Bestimmt die Basis einer fd-Menge F.

1. mache F rechts-minimal.
2. mache F links-minimal.
3. entferne redundante fds aus F.

1.2 Synthese

Erzeugt eine **verlustfreie** und **unabhängige** 3NF-Zerlegung $\mathcal{D}$ eines Relationenschemas $R = (U, F)$.

1. Sei $F^* := \textsc{Basis}(F)$.
2. $\mathcal{D} := \{\Pi_{XA}(R) | X \to A \in F^*\}$
3. Füge zu $\mathcal{D}$ eine Relation mit den Schlüsselattributen hinzu, falls keine Relation ein Oberschlüssel von R ist.

1.3 Synthese'

Erzeugt eine **verlustfreie** und **unabhängige** 3NF-Zerlegung $\mathcal{D}$ eines Relationenschemas $R = (U, F)$ mit möglichst wenig Relationenschemata.

1. Sei $F^* := \textsc{Basis}(F)$, $\mathcal{D} := \varnothing$.
2. Für jede linke Seite L einer fd aus F^* definiere $Y := \{A \in U | L \to A \in F^*\}$, also Y als Menge aller Attribute, die durch L eindeutig definiert wird. Ergänze dann $\mathcal{D}$ um die Relation mit den Attributen LY.
3. Ergänze (wie bei Synthese) die Zerlegung um eine Relation mit Schlüsselattributen, falls keine Relation alle Attribute eines Oberschlüssels von R enthält.

1.4 Tableau (LJP-Algorithmus)

Überprüft eine Zerlegung auf Verlustfreiheit.

1. Erzeuge das Tableau wie folgt: Tabelle mit einer Spalte für jedes Attribut und einer Zeilte für jedes Relationenschema in der Zerlegung. Fülle die Zellen der Tabelle mit a, falls das Attribut in der jeweiligen Relation enthalten ist, sonst mit b_i (i ist die Zeilennummer)
2. Für jede fd, für jede Zeile: Haben zwei Zeilen ein a bei **allen** Attributen der linken Seite der fd, dürfen in einer Zeile die a der anderen Zeile „übernommen“ werden.
3. Wiederholen, bis keine Veränderung mehr möglich.

Existiert eine Zeile die nur aus a besteht, ist die Zerlegung verlustfrei, andernfalls nicht.

1.5 Dekomposition

Erzeugt eine verlustfreie BCNF-Zerlegung von R.

1. Setze $\mathcal{D} := R$.
2. Solange $\mathcal{D}$ nicht in BCNF ist, wähle eine Relation (Attribute X) mit einer BCNF-verletzende fd und spalte diese anhand der fd $L \to R$ auf in zwei Relationen mit den Attributen LR und $X \setminus R$.

Die Zerlegung enthält möglichst wenige Relationenschemata.

1.6 BCNF-Test

R ist in BCNF, falls R keine BCNF-verletzende fd enthält.

2 Definitionen

2.1 Unabhängigkeit

Eine Zerlegung heißt unabhängig, falls $\bigcup_{i=1}^{k} F_i$ zu F äquivalent ist.

2.2 BCNF-verletzende fd

Eine fd $L \to R$ heißt BCNF-verletzend in R, falls L kein Oberschlüssel von R ist.

2.3 Verlustfreiheit

Jede erlaubte Instanz r zu R kann als Join der Projektionen $r_i = \Pi_{U_i}(r)$ rekonstruiert werden.

2.4 Abschluss von s

Der Abschluss eines Schedules s ist $\overline{s} = st_0c_0$.

2.5 freigegebene Projektion (commited projection)

Die freigegebene Projektion ist $CP(s) = \Pi_{commit(s)}(s)$.

2.6 Bereinigter Schedule

Der bereinigte Schedule ist $s^* = \Pi_{commit(s) \cup active(s)}(s)$.

2.7 View-serialisierbarkeit

Äquivalenz. Zwei Schedules s, s' heißen **view-äquivalent**, in Zeichen $s \approx_v s'$, falls $op(s) = op(s')$ und $RF^*(s) = RF^*(s')$ gilt.

Erklärung. Zwei Schedules s, s' heißen **view-äquivalent**, falls sie die selben Operationen ausführen (op) und im bereinigten Schedule die gleichen reads-from-Relationen gelten.

Serialisierbarkeit. Ein vollständiger Schedule heißt **view-serialisierbar**, falls er view-äquivalent zu einem seriellen Schedule ist.

2.8 Konfliktrelation

Konflikt. Zwei Operationen verschiedener Transaktionen im selben Schedule stehen in **Konflikt**, falls sie auf dem selben Datenobjekt operieren und mindestens eine von ihnen schreibt.

Konfliktrelation.

$$C(s) := \{p \xrightarrow{X} q | p(x) <_s q(X) \text{ und } p(x) \text{ und } q(x) \text{ stehen in Konflikt } \}.$$

Äquivalenz. Zwei Schedules heißen **konflikt-äquivalent**, in Zeichen $s \approx_s s'$, falls $op(s) = op(s')$ und $C^*(s) = C^*(s')$ gilt.

Serialisierbarkeit. Ein vollständiger Schedule heißt **konflikt-serialisierbar**, falls er konflikt-äquivalent zu einem seriellen Schedule ist.

2.9 Klassen

CSR ist die Klasse der konflikt-serialisierbaren Schedules, VSR die Klasse der view-serialisierbaren Schedules. Es gilt: $CSR \subsetneq VSR$.

2.10 Entropie

a) $entropy_A(T) = \varepsilon(p_1, \ldots, p_n) = -\sum_{i=1}^{n} p_i \log_2 p_i$.

b)

$$\begin{aligned} entropy_Z^A(T) &= entropy_Z(T_1, \ldots, T_n) \\ &= \sum_{v \in T_A} p_{A=v}(T) \cdot entropy_Z(\sigma_{A=v}(T)). \end{aligned}$$