Grundlagen: Datenbanken WS 15/16

2. Zentralübung / Wiederholung / Fragestunde

Harald Lang

gdb@in.tum.de

Diese Folien finden Sie online.

Die Mitschrift erhalten Sie im Anschluss.

Termine

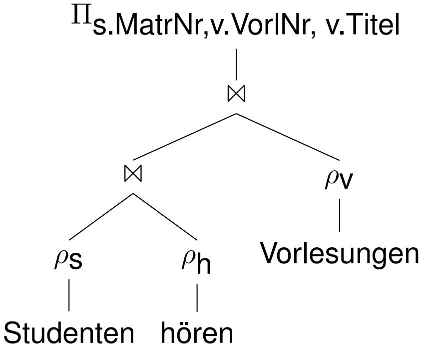
- Klausur
  - 24.02.2016, 10:30 - 12:30 Uhr
  - Anmeldung bis 15.01.2016
- Weihnachtsferien
  - 24.12.2015 - 6.1.2016
  - HINWEIS: **Ein** Übungblatt für die Wochen 21.-23.12 (Mo-Mi) und 7.-8.1. (Do, Fr).

Agenda

- Relationale Anfragesprachen
  - Anmerkungen zur relationalen Algebra
  - Anmerkung: Natural-Join vs. allgemeiner Theta-Join
  - Kalkül: Freie vs. gebundene Variablen
  - SQL
    - Anmerkung zu Unteranfragen
    - Gruppierung / Aggregation
    - All-Quantifizierung
- Relationale Entwurfstheorie
  - ...

Anmerkungen zur relationalen Algebra

- Geklammerter Ausdruck vs. Operatorbaum
- Umbenennung



Anmerkung: Natural-Join vs. allgemeiner Theta-Join

	Natural	Theta
Inner	$\bowtie$	$\bowtie_\theta$
Outer	$⟗$, $⟕$, $⟖$	$⟗_\theta$, $⟕_\theta$, $⟖_\theta$
Semi	$\ltimes$, $\rtimes$	$\ltimes_\theta$, $\rtimes_\theta$
Anti	$\rhd$, $\lhd$	$\rhd_\theta$, $\lhd_\theta$

- Natural
  - Implizite Gleichheitsbedingung auf gleichnamigen Attributen
  - Die gleichnamigen Attribute tauchen im Ergebnis nur einmal auf (inner und outer).
- Theta
  - Explizite (beliebige) Joinbedingung: θ .
  - Im Falle von Inner- und Outer-Join werden alle Attribute der beiden Eingaberelationen in das Ergebnis projiziert.

Kalkül: Freie vs. gebundene Variablen (1/2)

$R : \{[a, b, c]\}$, $S : \{[d, e, f]\}$

Relationale Algebra: $Q_1 := R \bowtie_{a=d} S$

Tupelkalkül: $Q_1 := \{\, r \circ s \mid \; r \in R \;\wedge\; s \in S \;\wedge\; r.a \;=\; s.d\}$

Domänenkalkül: $Q_1 := \{\, [a, b, c, a, e, f] \mid \; [a, b, c] \in R \;\wedge [a, e, f] \in S\}$

Kalkül: Freie vs. gebundene Variablen (2/2)

$R : \{[a, b, c]\}$, $S : \{[d, e, f]\}$

Relationale Algebra: $Q_2 := R \ltimes_{a=d} S$

Tupelkalkül: $Q_2 := \{\, r \mid \; r \in R \;\wedge \exists\, s \in S(r.a = s.d)\,\}$

Domänenkalkül: $Q_2 := \{\, [a, b, c] \mid [a, b, c] \in R \;\wedge \exists\, e, f([a, e, f] \in S)\,\}$

Alle Domänen- / Tupelvariablen tauchen entweder in der Projektion auf, oder wurden explizit quantifiziert.

Anmerkung zu Unteranfragen in SQL

```
select * from R
where R.x = (select y from S where ... )
```

Die Unteranfrage muss genau einen skalaren Wert ausgeben.

```
select * from R
where R.x in (select y from S where ... )
```

Die Unteranfrage muss eine Menge von skalaren Werten ausgeben.

Übung: Gruppierung/Aggregation in SQL

"Gute Studenten": Finde Studenten in höheren Semestern (sem > 5) mit einem Notenschnitt < 2.5. Geben Sie **MatrNr**, **Name** und **Notenschnitt** aus.

Übung: All-Quantifizierung in SQL

Finde Studenten, die ALLE Vorlesungen gehört haben.

Relationale Entwurftheorie

Funktionale Abhängigkeiten (kurz FDs, für functional dependencies):

- Seien α und β Attributmengen eines Schemas $\mathcal{R}$.
- Wenn auf $\mathcal{R}$ die FD $\alpha \rightarrow \beta$ definiert ist, dann sind nur solche Ausprägungen R zulässig, für die folgendes gilt:
  - Für alle Paare von Tupeln $r, t \in R$ mit $r.\alpha = t.\alpha$ muss auch gelten $r.\beta = t.\beta$.

Übung: Relationenausprägung vervollständigen

Gegen seien die folgende Relationenausprägung und die funktionalen Abhängigkeiten. Bestimmen Sie zunächst x und danach y, sodass die FDs gelten.

$$\begin{aligned} B &\rightarrow A \\ AC &\rightarrow D \end{aligned}$$

A	B	C	D
7	3	5	8
x	4	2	8
7	3	6	9
1	4	2	y

Funktionale Abhängigkeiten

Seien α,β,γ,$\delta \subseteq \mathcal{R}$

Axiome von Armstrong:

- *Reflexivität:*
  Falls $\beta \subseteq \alpha$, dann gilt immer $\alpha \rightarrow \beta$
- *Verstärkung*:
  Falls $\alpha \rightarrow \beta$ gilt, dann gilt auch $\alpha\gamma \rightarrow \beta\gamma$
- *Transitivität*:
  Falls $\alpha \rightarrow \beta$ und $\beta \rightarrow \gamma$ gelten, dann gilt auch $\alpha \rightarrow \gamma$

Mithilfe dieser Axiome können alle *geltenden* FDs hergeleitet werden.

Sei F eine FD-Menge. Dann ist F^+ die Menge aller geltenden FDs (Hülle von F)

Funktionale Abhängigkeiten

Nützliche und vereinfachende Regeln:

- *Vereinigungsregel*:
  Falls $\alpha \to \beta$ und $\alpha \to \gamma$ gelten, dann gilt auch $\alpha \to \beta\gamma$
- *Dekompositionsregel*:
  Falls $\alpha \to \beta\gamma$ gilt, dann gilt auch $\alpha \to \beta$ und $\alpha \to \gamma$
- *Pseudotransitivitätsregel*:
  Falls $\alpha \to \beta$ und $\gamma\beta \to \delta$ gelten, dann gilt auch $\gamma\alpha \to \delta$

Schlüssel

- Schlüssel identifizieren jedes Tupel einer Relation $\mathcal{R}$ eindeutig.
- Eine Attributmenge $\alpha \subseteq \mathcal{R}$ ist ein **Superschlüssel**, gdw. $\alpha \rightarrow \mathcal{R}$
- Ist α zudem noch *minimal*, ist es auch ein **Kandidatenschlüssel** (meist mit κ bezeichnet)
  - Es existiert also kein $\alpha' \subset \alpha$ für das gilt: $\alpha' \rightarrow \mathcal{R}$

- I.A. existieren mehrere Super- und Kandidatenschlüssel.
- Man muss sich bei der Realisierung für einen Kandidatenschlüssel entscheiden, dieser wird dann **Primärschlüssel** genannt.
- Der triviale Schlüssel $\alpha = \mathcal{R}$ existiert immer.

Übung: Schlüsseleigenschaft von Attributmengen ermitteln

- Ob ein gegebenes α ein Schlüssel ist, kann mithilfe der Armstrong Axiome ermittelt werden (i.A. zu aufwendig!)
- Besser: Die **Attributhülle** $AH(\alpha)$ bestimmen.

- Beispiel: $\mathcal{R} = \{\ A\ ,\ B\ ,\ C\ ,\ D\ \}$, mit $F_{\mathcal{R}} = \{AB \rightarrow CD, B \rightarrow C, D \rightarrow B\}$

$AH(\{D\})$:

$AH(\{A, D\})$:

$AH(\{A, B, D\})$:

Normalformen: 1NF $\supset$ 2NF $\supset$ 3NF $\supset$ BCNF $\supset$ 4NF

- **1. NF**: Attribute haben nur atomare Werte, sind also nicht mengenwertig.
- **2. NF**: Jedes Nichtschlüsselattribut (NSA) ist voll funktional abhängig von jedem Kandidatenschlüssel.
  - β hängt **voll funktional** von α ab ($\alpha \xrightarrow{\bullet} \beta$), gdw. $\alpha \to \beta$ und es existiert kein $\alpha' \subset \alpha$, so dass $\alpha' \to \beta$ gilt.
- **3. NF**: Frei von transitiven Abhängigkeiten (*in denen NSAe über andere NSAe vom Schlüssel abhängen*).
  - für alle geltenden nicht-trivialen FDs $\alpha \to \beta$ gilt entweder
    - α ist ein Superschlüssel, oder
    - jedes Attribut in β ist in einem Kandidatenschlüssel enthalten
- **BCNF**: Die linken Seiten (α) aller geltenden nicht-trivalen FDs sind Superschlüssel.
- **4. NF**: Die linken Seiten (α) aller geltenden nicht-trivalen MVDs sind Superschlüssel.

Übung: Höchste NF bestimmen

$$\mathcal{R} : \{[\ A\,,\ B\,,\ C\,,\ D\,,\ E\ ]\}$$

$$\begin{aligned} A &\rightarrow BCDE \\ AB &\rightarrow C \end{aligned}$$

- ○ 1. NF
- ○ 2. NF
- ○ 3. NF
- ○ 4. NF
- ○ BCNF
- ○ keine der angegebenen

Übung: Höchste NF bestimmen (2)

$$\mathcal{R} : \{[\ A\,,\, B\,,\, C\,,\, D\,,\, E\ ]\}$$

$$\begin{aligned} A &\rightarrow BCDE \\ B &\rightarrow C \end{aligned}$$

- ○ 1. NF
- ○ 2. NF
- ○ 3. NF
- ○ 4. NF
- ○ BCNF
- ○ keine der angegebenen

Schema in 3. NF überführen

Synthesealgorithmus

- Eingabe:
  - **Kanonische Überdeckung** $\mathcal{F}_c$
    - Linksreduktion
    - Rechtsreduktion
    - FDs der Form $\alpha \to \emptyset$ entfernen (sofern vorhanden)
    - FDs mit gleicher linke Seite zusammenfassen
- Algorithmus:
  1. Für jede FD $\alpha \to \beta$ in $\mathcal{F}_c$ forme ein Unterschema $\mathcal{R}_\alpha = \alpha \cup \beta$, ordne $\mathcal{R}_\alpha$ die FDs $\mathcal{F}_\alpha := \{\alpha' \to \beta' \in \mathcal{F}_c \mid \alpha' \cup \beta' \subseteq \mathcal{R}_\alpha\}$ zu
  2. Füge ein Schema $\mathcal{R}_\kappa$ mit einem Kandidatenschlüssel hinzu
  3. Eliminiere redundante Schemata, d.h. falls $\mathcal{R}_i \subseteq \mathcal{R}_j$, verwerfe $\mathcal{R}_i$
- Ausgabe:
  - Eine Zerlegung des unsprünglichen Schemas, wo alle Schemata in 3.NF sind.
  - Die Zerlegung ist **abhängigkeitsbewahrend** und **verlustfrei**.

Übung: Synthesealgorithmus (1)

$$\mathcal{R} : \{[\ A\,, B\,, C\,, D\,, E\,, F\ ]\}$$

$$\begin{aligned} B &\rightarrow ACDEF \\ EF &\rightarrow BC \\ A &\rightarrow D \end{aligned}$$

Übung: Synthesealgorithmus (2)

$$\mathcal{R} = \{\, A,\, B,\, C,\, D,\, E,\, F,\, G\},\, F_{\mathcal{R}} = \{A \rightarrow BCDE,\, E \rightarrow FB,\, F \rightarrow A\}$$

Schema in BCNF überführen

BCNF-Dekompositionsalgorithmus

- Starte mit $Z = \{\mathcal{R}\}$
- Solange es noch ein $\mathcal{R}_i \in Z$ gibt, das nicht in BCNF ist:
  - Finde eine FD $(\alpha \to \beta) \in F^+$ mit
    - $\alpha \cup \beta \subseteq \mathcal{R}_i$
    - $\alpha \cap \beta = \emptyset$
    - $\alpha \to \mathcal{R}_i \notin F^+$
  - Zerlege $\mathcal{R}_i$ in $\mathcal{R}_{i.1} := \alpha \cup \beta$ und $\mathcal{R}_{i.2} := \mathcal{R}_i - \beta$
  - Entferne $\mathcal{R}_i$ aus Z und füge $\mathcal{R}_{i.1}$ und $\mathcal{R}_{i.2}$ ein, also $Z := (Z - \{\mathcal{R}_i\}) \cup \{\mathcal{R}_{i.1}\} \cup \{\mathcal{R}_{i.2}\}$

BCNF-Dekompositionsalgorithmus

$$\mathcal{R} = \{\, A,\ B,\ C,\ D,\ E,\ F\},\ F_{\mathcal{R}} = \{B \to AD,\ DEF \to B,\ C \to AE\}$$