Kapitel DB:V

Anfragen und Änderungen

Ausgangspunkt: Basisrelationen, die in der Datenbank gespeichert sind.

Ziel: abgeleitete Relationen, die aus Basisrelationen berechnet werden.

Ableitung von Relationen mit drei unterschiedlichen Mechanismen:

1. Anfrage

2. Sicht

3. Snapshot

Anfragen und Änderungen

Ausgangspunkt: Basisrelationen, die in der Datenbank gespeichert sind.

Ziel: abgeleitete Relationen, die aus Basisrelationen berechnet werden.

Ableitung von Relationen mit drei unterschiedlichen Mechanismen:

1. Anfrage

   Folge von Operationen, die aus Basisrelationen eine Ergebnisrelation berechnet. Die Ergebnisrelation kann angezeigt und interaktiv oder durch ein Programm weiterverarbeitet werden.

2. Sicht

   Folge von Operationen, die unter einem Sichtnamen *langfristig* gespeichert und unter diesem Namen wieder aufgerufen werden kann; ergibt eine Sichtrelation.

3. Snapshot

   Ergebnisrelation einer Anfrage, die unter einem Snapshot-Namen abgelegt wird, aber nie ein zweites Mal (mit geänderten Basisrelationen) berechnet wird. Beispiel: Zusammenstellung einer Jahresbilanz.

Bemerkungen:

- Bei der Ableitung von Relationen bleiben die Basisrelationen unverändert.
- Update- und Änderungsoperationen verändern die Basisrelationen.
- Die Einbettung der Anfragesprache in eine Programmiersprache ermöglicht eine integrierte Weiterverarbeitung.

Anfragen und Änderungen

Eigenschaften relationaler Anfragesprachen [Heuer/Scholl 1991]

- ad-hoc-formulierbar
- deklarativ
- mengenbasiert
- abgeschlossen
- orthogonal
- adäquat

Anfragen und Änderungen

Eigenschaften relationaler Anfragesprachen [Heuer/Scholl 1991]

- ad-hoc-formulierbar

  Man kann Anfragen formulieren, ohne ein Programm dafür zu schreiben.

- deklarativ

  Man formuliert im deklarativen Stil „Was will ich haben?“ und nicht prozedural „Wie erhalte ich das, was ich haben will?“

- mengenbasiert

  Die Operationen arbeiten auf Datenmengen – nicht navigierend auf einzelnen Elementen.

- abgeschlossen

  Das Ergebnis einer Anfrage ist wieder vom Typ eines Operands (= Relation) und direkt als Eingabe für weitere Anfragen verwendbar.

- orthogonal

  Die Operationen sind ohne Einschränkung kombinierbar.

- adäquat

  Die Charakteristika des unterliegenden Datenmodells werden unterstützt.

Anfragen und Änderungen

Eigenschaften relationaler Anfragesprachen (Fortsetzung)

- vollständig
- optimierbar
- effizient
- sicher
- spezialisiert

Anfragen und Änderungen

Eigenschaften relationaler Anfragesprachen (Fortsetzung)

- vollständig

  Die Anfragesprache bildet (mindestens) die Relationenalgebra oder den sicheren Relationenkalkül ab.

- optimierbar

  Die Anfragesprache umfasst wenige Operationen, für die es leistungsfähige Optimierungsregeln gibt.

- effizient

  Die Anfragen sind effizient ausführbar.

- sicher

  Keine syntaktisch korrekte Anfrage gerät in eine Endlosschleife oder liefert ein unendliches Ergebnis.

- spezialisiert

  Die Anfragesprache ist keine vollständige Programmiersprache. Diese Eigenschaft folgt aus Optimierbarkeit, Effizienz, Sicherheit.

Bemerkungen:

- Beispiel für Orthogonalität: an jeder Stelle, an der ein Basisrelationenname stehen kann, darf auch ein Anfrage stehen. In SQL-89 darf in der From-Klausel keine Anfrage stehen.
- Beispiel für Optimierbarkeit: Auswertung von Select-Operationen vor Join-Operationen.
- Beispiel für Effizienz: im Relationenmodell ist jede Operation in $O(n^2)$, mit n = Anzahl der Tupel einer Relation.

Relationale Algebra

Eine Algebra $\mathcal{A} = \langle M, \Omega \rangle$ besteht aus

1. einem Wertebereich M sowie
2. einer Menge von Operatoren Ω mit $\circ : M^n \rightarrow M$, $\circ \in \Omega$.

Bezogen auf die Relationenalgebra:

1. M ist die Menge aller Relationen über den Relationenschemata, die zu einer festen Menge von Attributen gebildet werden können.
2. Ω ist eine Menge von Operationen auf Relationen.

Relationale Algebra

Einstellige Operationen: Selektion

Syntax: $\sigma_{\text{<COND>}}(r)$

Semantik: Auswahl derjenigen Tupel in der Relation $r(\mathcal{R})$, die das Selektionsprädikat <COND> erfüllen:
$\{t \mid (t \in r) \wedge (\text{<COND>}(t) = \textit{TRUE})\}$

<COND> ist aus folgenden Elementen aufgebaut:

1. Operanden sind Attributnamen aus $\mathcal{R}$ oder Konstanten
2. arithmetische Vergleichsoperatoren $=, <, \leq, >, \geq, \neq$
3. logische Operatoren $\wedge, \vee, \neg$

Relationale Algebra

Einstellige Operationen: Selektion

Syntax: $\sigma_{<\text{COND}>}(r)$

Semantik: Auswahl derjenigen Tupel in der Relation $r(\mathcal{R})$, die das Selektionsprädikat $<\text{COND}>$ erfüllen:
$\{t \mid (t \in r) \wedge (<\text{COND}>(t) = \textit{TRUE})\}$

$<\text{COND}>$ ist aus folgenden Elementen aufgebaut:

1. Operanden sind Attributnamen aus $\mathcal{R}$ oder Konstanten
2. arithmetische Vergleichsoperatoren $=, <, \leq, >, \geq, \neq$
3. logische Operatoren $\wedge, \vee, \neg$

Beispiel:

Ausleihe	
InvNr	**Name**
4711	Meyer
1201	Schulz
0007	Müller
4712	Meyer

$\sigma_{\text{Name} \leq \text{'N'}}(\text{Ausleihe}) \quad \leadsto$

InvNr	Name
4711	Meyer
0007	Müller
4712	Meyer

Bemerkungen:

- Unmittelbar aufeinander folgende Selektionen lassen sich in ihrer Reihenfolge beliebig vertauschen. Die Hintereinanderausführung von Selektionen besitzt eine konjunktive Semantik: es werden nur diejenigen Tupel ausgewählt, die in der Schnittmenge aller durch die jeweilige Selektion spezifizierten Tupelmengen liegen. Die Schnittmengenbildung ist assoziativ.
- Kombination von Selektion und Projektion: bilden die Attribute in $<$COND$>$ eine Teilmenge der Attribute einer nachfolgenden Projektion, lassen sich Selektion und Projektion vertauschen. Falls nicht, muss die Selektion zuerst ausgeführt werden, um sicherzustellen, dass $<$COND$>$ definiert ist.
- Zur Bezeichnung einer Relation können r und $r(\mathcal{R})$ gleichermaßen verwendet werden – abhängig davon, ob auf das Relationenschema $\mathcal{R}$ im aktuellen Kontext Bezug genommen werden muss.

Relationale Algebra

Einstellige Operationen: Projektion

Syntax: $\pi_\alpha(r)$

Semantik: Projektion aller Tupel in r bzgl. der Attribute in α : $\{t(\alpha) \mid t \in r\}$

Relationale Algebra

Einstellige Operationen: Projektion

Syntax: $\pi_\alpha(r)$

Semantik: Projektion aller Tupel in r bzgl. der Attribute in α : $\{t(\alpha) \mid t \in r\}$

Beispiel:

Buch			
InvNr	Titel	ISBN	Autor
0007	Dr. No	3-125	James Bond
1201	Objektbanken	3-111	Heuer
4711	Datenbanken	3-765	Vossen
4712	Datenbanken	3-891	Ullman
4717	Pascal	3-999	Wirth

$\pi_{\text{InvNr,ISBN}}(\text{Buch}) \quad \leadsto$

InvNr	ISBN
0007	3-125
1201	3-111
4711	3-765
4712	3-891
4717	3-999

Bemerkungen:

- Unmittelbar aufeinander folgende Projektionen lassen sich in ihrer Reihenfolge vertauschen. Die Hintereinanderausführung von Projektionen besitzt eine konjunktive Semantik: es wird nur auf diejenigen Attribute projiziert, die in der Schnittmenge aller durch die jeweilige Projektion spezifizierten Attributmengen liegen. Die Schnittmengenbildung ist assoziativ.
- Gilt $\alpha \subseteq \beta \subseteq \gamma$, so ist die unmittelbare Hintereinanderausführung der Projektionen bzgl. dieser Attributmengen äquivalent zu der alleinigen Anwendung der Projektion bzgl. α : $\pi_\alpha \pi_\beta \pi_\gamma(r) \equiv \pi_\alpha(r)$
- α ist eine Menge; entsprechend müsste man z.B. bei $\alpha = \{\text{InvNr}, \text{ISBN}\}$ die Selektion π_α als $\pi_{\{\text{InvNr},\text{ISBN}\}}$ notieren. Die Notation von Attributnamen ohne Mengenklammern im Index ist formal unsauber, hat sich in der Datenbankliteratur aber durchgesetzt.

Relationale Algebra

Einstellige Operationen: Umbenennung

Syntax (a): $\rho_{\text{<Mapping>}}(r(\mathcal{R}))$, mit $\text{<Mapping>} = \{A_2 \leftarrow A_1 \mid A_1 \in \mathcal{R}, A_2 \notin \mathcal{R}\}$

Semantik: Umbenennung von Attribut A_1 zu A_2 in der Ergebnisrelation.

Relationale Algebra

Einstellige Operationen: Umbenennung

Syntax (a): $\rho_{\text{<Mapping>}}(r(\mathcal{R}))$, mit $\text{<Mapping>} = \{A_2 \leftarrow A_1 \mid A_1 \in \mathcal{R}, A_2 \notin \mathcal{R}\}$

Semantik: Umbenennung von Attribut A_1 zu A_2 in der Ergebnisrelation.

Syntax (b): $\rho_s(r)$

Semantik: Umbenennung der Relation r zu s

Relationale Algebra

Einstellige Operationen: Umbenennung

Syntax (a): $\rho_{\text{<Mapping>}}(r(\mathcal{R}))$, mit $\text{<Mapping>} = \{A_2 \leftarrow A_1 \mid A_1 \in \mathcal{R}, A_2 \notin \mathcal{R}\}$

Semantik: Umbenennung von Attribut A_1 zu A_2 in der Ergebnisrelation.

Syntax (b): $\rho_s(r)$

Semantik: Umbenennung der Relation r zu s

Beispiel:

zu (a) $\rho_{\text{Buchtitel}\leftarrow\text{Titel}}(\text{Buch})$

zu (b) $\rho_{\text{Dokument}}(\text{Buch})$

Bemerkungen:

- Mit der Umbenennung kann man die Voraussetzung zur Anwendung von Mengenoperation schaffen.
- Die Umbenennung ermöglicht natürliche Verbunde, wo ansonsten kartesische Produkte entstehen würden: unterschiedliche Attribute werden gleich benannt.
- Die Umbenennung ermöglicht kartesische Produkte, wo ansonsten natürliche Verbunde entstehen würden: gleiche Attribute werden verschieden benannt.
- Beispiel: $\mathcal{R} = \text{voraussetzen} = \{\text{Nachfolger}, \text{Vorgaenger}\}$. Bestimmung der Vorvorgänger von 4711:

$$
\begin{array}{l}
\pi_{\text{V2.Vorgaenger}}(\\
\quad \sigma_{\text{V1.Nachfolger}=4711 \,\wedge\, \text{V1.Vorgaenger}=\text{V2.Nachfolger}}(\\
\quad\quad \rho_{\text{V1}}(\text{voraussetzen}) \times \rho_{\text{V2}}(\text{voraussetzen}) \\
\quad) \\
)
\end{array}
$$

Relationale Algebra

Mengenoperationen: Vereinigung, Durchschnitt, Differenz

Syntax: $r_1 \cup r_2,\ \ r_1 \cap r_2,\ \ r_1 - r_2$

Semantik:

$$= \{t \mid t \in r_1 \vee t \in r_2\}$$
$$= \{t \mid t \in r_1 \wedge t \in r_2\}$$
$$= \{t \mid t \in r_1 \wedge t \notin r_2\}$$

Mengenoperationen sind nur für Relationen mit gleichem Schema – d. h., Schemata mit gleichen Attributnamen und Domänen – definiert.

Relationale Algebra

Mengenoperationen: Vereinigung, Durchschnitt, Differenz

Syntax: $r_1 \cup r_2,\ \ r_1 \cap r_2,\ \ r_1 - r_2$

Semantik:
$$\begin{aligned} r_1 \ \cup\ r_2 &= \{t \mid t \in r_1 \vee t \in r_2\} \\ &= \{t \mid t \in r_1 \wedge t \in r_2\} \\ &= \{t \mid t \in r_1 \wedge t \notin r_2\} \end{aligned}$$

Mengenoperationen sind nur für Relationen mit gleichem Schema – d. h., Schemata mit gleichen Attributnamen und Domänen – definiert.

Beispiel:

Buch
Autor
James Bond
Heuer
Vossen
Ullman
Wirth

Book
Author
Witt
Vossen
Silberschatz
Meier
Wirth

Buch $\cup\ \rho_{\text{Autor}\leftarrow\text{Author}}(\text{Book})$ $\leadsto$

Buch
Autor
James
Heuer
Vossen
Ullman
Wirth
Witt
Vossen
Silberschatz
Meier

Relationale Algebra

Mengenoperationen: Vereinigung, Durchschnitt, Differenz

Syntax: $r_1 \cup r_2,\ \ r_1 \cap r_2,\ \ r_1 - r_2$

Semantik:

$$r_1 \cup r_2 = \{t \mid t \in r_1 \lor t \in r_2\}$$
$$r_1 \cap r_2 = \{t \mid t \in r_1 \land t \in r_2\}$$
$$= \{t \mid t \in r_1 \land t \notin r_2\}$$

Mengenoperationen sind nur für Relationen mit gleichem Schema – d. h., Schemata mit gleichen Attributnamen und Domänen – definiert.

Beispiel:

Buch
Autor
James Bond
Heuer
Vossen
Ullman
Wirth

Book
Author
Witt
Vossen
Silberschatz
Meier
Wirth

$\text{Buch} \cap \rho_{\text{Autor} \leftarrow \text{Author}}(\text{Book}) \quad \rightsquigarrow$

Buch
Autor
Vossen
Wirth

Relationale Algebra

Mengenoperationen: Vereinigung, Durchschnitt, Differenz

Syntax: $r_1 \cup r_2,\ r_1 \cap r_2,\ r_1 - r_2$

Semantik:
$$r_1 \cup r_2 = \{t \mid t \in r_1 \vee t \in r_2\}$$
$$r_1 \cap r_2 = \{t \mid t \in r_1 \wedge t \in r_2\}$$
$$r_1 - r_2 = \{t \mid t \in r_1 \wedge t \notin r_2\}$$

Mengenoperationen sind nur für Relationen mit gleichem Schema – d. h., Schemata mit gleichen Attributnamen und Domänen – definiert.

Beispiel:

Buch
Autor
James Bond
Heuer
Vossen
Ullman
Wirth

Book
Author
Witt
Vossen
Silberschatz
Meier
Wirth

Buch $- \rho_{\text{Autor} \leftarrow \text{Author}}$(Book) $\rightsquigarrow$

Buch
Autor
James Bond
Heuer
Ullman

Bemerkungen:

- Die Syntax $A - B$ (anstelle von $A \setminus B$) zur Bezeichnung der Mengendifferenz ist in der Relationenalgebra üblich.

Relationale Algebra

Kartesisches Produkt

Syntax: $r_1(\mathcal{R}_1) \times r_2(\mathcal{R}_2)$

Semantik: Sei $\mathcal{R}_1 \cap \mathcal{R}_2 = \emptyset$. Bildung aller $|r_1| \cdot |r_2|$ Tupel über $\mathcal{R}_1 \cup \mathcal{R}_2$:
$\{t \mid t \in r(\mathcal{R}_1 \cup \mathcal{R}_2) \;\wedge\; t(\mathcal{R}_1) \in r_1 \;\wedge\; t(\mathcal{R}_2) \in r_2\}$ [natürlicher Verbund]

Relationale Algebra

Kartesisches Produkt

Syntax: $r_1(\mathcal{R}_1) \times r_2(\mathcal{R}_2)$

Semantik: Sei $\mathcal{R}_1 \cap \mathcal{R}_2 = \emptyset$. Bildung aller $|r_1| \cdot |r_2|$ Tupel über $\mathcal{R}_1 \cup \mathcal{R}_2$:

$\{t \mid t \in r(\mathcal{R}_1 \cup \mathcal{R}_2) \ \wedge \ t(\mathcal{R}_1) \in r_1 \ \wedge \ t(\mathcal{R}_2) \in r_2\}$ [natürlicher Verbund]

Beispiel:

Buch
Autor
James Bond
Heuer
Vossen
Ullman
Wirth

Book
Author
Witt
Vossen
Silberschatz
Meier
Wirth

Buch $\times$ Book $\leadsto$

Autor	Author
James Bond	Witt
James Bond	Vossen
James Bond	Silberschatz
James Bond	Meier
James Bond	Wirth
Heuer	Witt
Heuer	Vossen
...	...

Bemerkungen:

- Bei gleichen Attributnamen in den beteiligten Relationenschemata wird eine eindeutige Benennung dadurch erzwungen, dass ein qualifizierender Attributbezeichner $r.A$ aus dem Namen der Relation r und dem Attributnamen A konstruiert wird.
- Das kartesische Produkt ist eine Operation, die quadratischen Platz (und folglich auch mindestens quadratische Rechenzeit) benötigt.

Relationale Algebra

Konstruktion von Ausdrücken

1. Jede Basisrelation ist ein relationaler Algebra-Ausdruck.

2. Seien E_1 und E_2 relationale Algebra-Ausdrücke *(expressions)*, dann sind auch folgende Ausdrücke relationale Algebra-Ausdrücke:

   (a) $E_1 \cup E_2$ und $E_1 - E_2$,

   (b) $E_1 \times E_2$

   (c) $\sigma_{<\text{COND}>}(E_1)$,

   (d) $\pi_\alpha(E_1)$

   (e) $\rho_{A_2 \leftarrow A_1}(E_1)$ und $\rho_s(E_1)$,

Relationale Algebra

Konstruktion von Ausdrücken

1. Jede Basisrelation ist ein relationaler Algebra-Ausdruck.
2. Seien E_1 und E_2 relationale Algebra-Ausdrücke *(expressions)*, dann sind auch folgende Ausdrücke relationale Algebra-Ausdrücke:
   (a) $E_1 \cup E_2$ und $E_1 - E_2$, wobei E_1 und E_2 das gleiche Relationenschema besitzen müssen.
   (b) $E_1 \times E_2$
   (c) $\sigma_{\text{<COND>}}(E_1)$, wobei <COND> ein Prädikat über den Attributen des Relationenschemas von E_1 ist.
   (d) $\pi_\alpha(E_1)$ mit einer Attributliste α, deren Attribute in dem Relationenschema von E_1 vorkommen.
   (e) $\rho_{A_2 \leftarrow A_1}(E_1)$ und $\rho_s(E_1)$, wobei A_1 ein Attributname in dem Relationenschema von E_1 ist und A_2 dort nicht als Attributname vorkommt.

Relationale Algebra

Konstruktion von Ausdrücken (Fortsetzung)

Definition 1 (relational vollständig)

Eine Menge von Operationen Ω ist relational vollständig, wenn mit Ω jede relationenalgebraische Operation ausgedrückt („simuliert“) werden kann, die sich mit einer anderen Menge Ω' von Operationen ausdrücken lässt.

Relationale Algebra

Konstruktion von Ausdrücken (Fortsetzung)

Definition 1 (relational vollständig)

Eine Menge von Operationen Ω ist relational vollständig, wenn mit Ω jede relationenalgebraische Operation ausgedrückt („simuliert“) werden kann, die sich mit einer anderen Menge Ω' von Operationen ausdrücken lässt.

Satz 1

Die Menge der Operationen $\Omega = \{\cup, -, \times, \sigma, \pi\}$ ist relational vollständig.

Satz 2

Die Menge der Operationen in Satz 1 ist unabhängig: keine Operation kann weggelassen werden, ohne die Vollständigkeit zu verlieren.

Insbesondere lassen sich ausdrücken:

- $r_1 \cap r_2$ durch $r_1 - (r_1 - r_2)$
- $r_1 \div r_2$ durch $\pi_{(\mathcal{R}_1 - \mathcal{R}_2)}(r_1) - \pi_{(\mathcal{R}_1 - \mathcal{R}_2)}(\ (\pi_{(\mathcal{R}_1 - \mathcal{R}_2)}(r_1) \times r_2) - r_1\ )$
- $\bowtie$ durch die Kombination von π, σ und $\times$
- $\times$ durch die Kombination von $\bowtie$ und ρ

Bemerkungen:

- Die Beschränkung des Selektionsprädikates <COND> auf die einfache Attribut- und Konstantenselektion gefährdet nicht die Vollständigkeitseigenschaft von Ω. D.h., die booleschen Operatoren sind nicht notwendig.

Relationale Algebra

Konstruktion von Ausdrücken (Fortsetzung)

(a) Ausführung komplexer Operationen durch Schachtelung:

$$\pi_{\text{Nachname,Gehalt}}(\sigma_{\text{AbteilungsNr}=5}(\text{Angestellte}))$$

Relationale Algebra

Konstruktion von Ausdrücken (Fortsetzung)

(a) Ausführung komplexer Operationen durch Schachtelung:

$$\pi_{\text{Nachname,Gehalt}}(\sigma_{\text{AbteilungsNr}=5}(\text{Angestellte}))$$

Angestellte				
Vorname	Nachname	Abteilung	AbteilungsNr	Gehalt

Relationale Algebra

Konstruktion von Ausdrücken (Fortsetzung)

(a) Ausführung komplexer Operationen durch Schachtelung:

Angestellte

Angestellte				
Vorname	Nachname	Abteilung	AbteilungsNr	Gehalt
...	...	...	...	...
Derk	Smith	Research	5	6000
Peter	Sotelo	Research	5	5000
Pam	Brin	Accounting	3	5500
...	...	...	...	...

Konstruktion von Ausdrücken (Fortsetzung)

(a) Ausführung komplexer Operationen durch Schachtelung:

$$\sigma_{\text{AbteilungsNr}=5}(\text{Angestellte})$$

Angestellte				
Vorname	Nachname	Abteilung	AbteilungsNr	Gehalt
...	...	...	...	...
Derk	Smith	Research	5	6000
Peter	Sotelo	Research	5	5000
Pam	Brin	Accounting	3	5500
...	...	...	...	...

Relationale Algebra

Konstruktion von Ausdrücken (Fortsetzung)

(a) Ausführung komplexer Operationen durch Schachtelung:

$$\pi_{\text{Nachname,Gehalt}}(\sigma_{\text{AbteilungsNr}=5}(\text{Angestellte}))$$

Angestellte				
Vorname	Nachname	Abteilung	AbteilungsNr	Gehalt
...	...	...	...	...
Derk	Smith	Research	5	6000
Peter	Sotelo	Research	5	5000
Pam	Brin	Accounting	3	5500
...	...	...	...	...

Relationale Algebra

Konstruktion von Ausdrücken (Fortsetzung)

(b) Ausführung komplexer Operationen mit expliziten Zwischenergebnissen:

$$\text{Abt5\_Angestellte} \leftarrow \sigma_{\text{AbteilungsNr}=5}(\text{Angestellte})$$

$$\text{Ergebnisrelation}(\text{Name}, \text{Einkommen}) \leftarrow \pi_{\text{Nachname},\text{Gehalt}}(\text{Abt5\_Angestellte})$$

Relationale Algebra

Konstruktion von Ausdrücken (Fortsetzung)

(b) Ausführung komplexer Operationen mit expliziten Zwischenergebnissen:

$$\text{Abt5\_Angestellte} \leftarrow \sigma_{\text{AbteilungsNr}=5}(\text{Angestellte})$$

$$\text{Ergebnisrelation}(\text{Name}, \text{Einkommen}) \leftarrow \pi_{\text{Nachname},\text{Gehalt}}(\text{Abt5\_Angestellte})$$

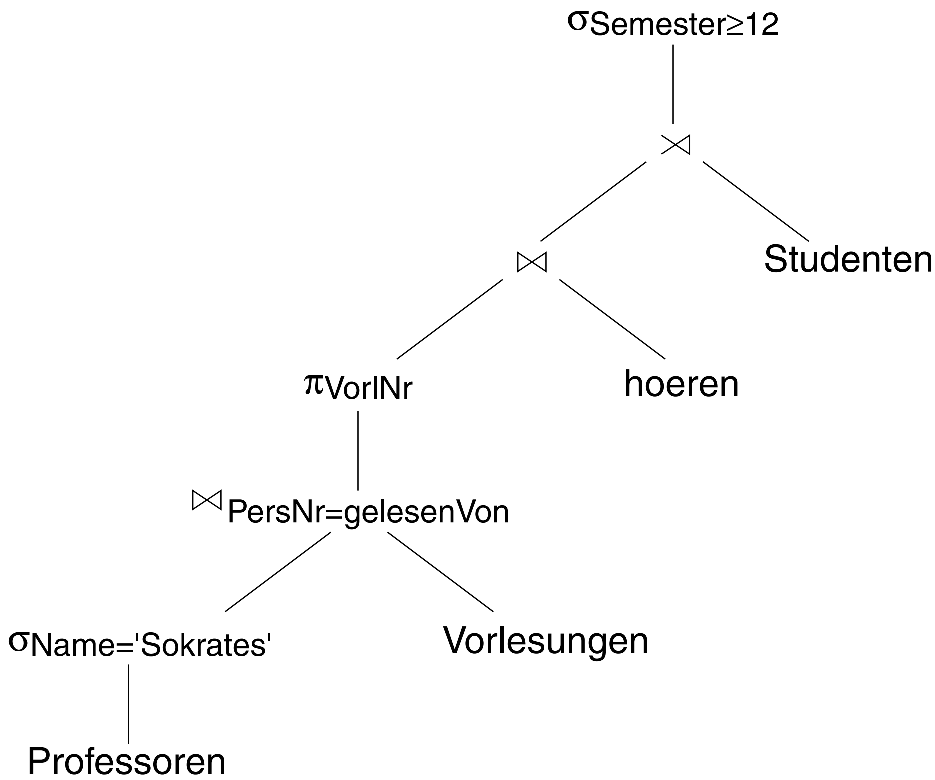
Abt5_Angestellte				
Vorname	Nachname	Abteilung	AbteilungsNr	Gehalt
Derk	Smith	Research	5	6000
Peter	Sotelo	Research	5	5000

Ergebnisrelation	
Name	Einkommen
Smith	6000
Sotelo	5000

Relationale Algebra

Konstruktion von Ausdrücken (Fortsetzung)

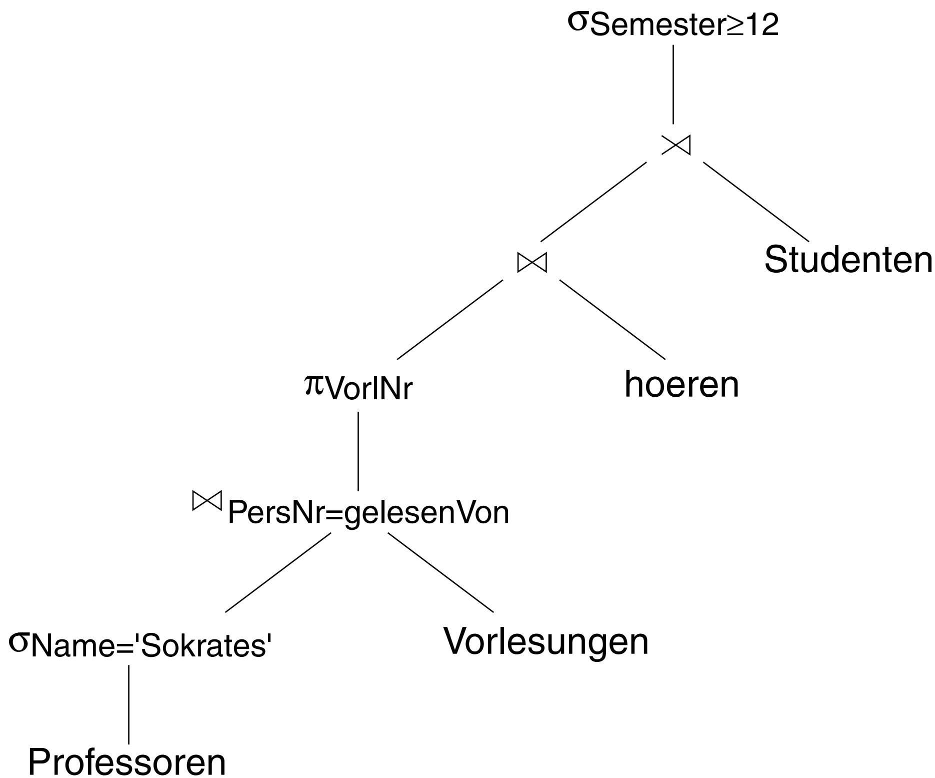
Baumdarstellung:



Relationale Algebra

Konstruktion von Ausdrücken (Fortsetzung)

Baumdarstellung:



Textuell:

$$\sigma_{\text{Semester}\geq 12}($$
$$((\pi_{\text{VorlNr}}(\sigma_{\text{Name}='\text{Sokrates}'}(\text{Professoren}) \bowtie_{\text{PersNr}=\text{gelesenVon}} \text{Vorlesungen})) \bowtie \text{hoeren})$$
$$\rtimes \text{Studenten})$$

Relationale Algebra

Weitere zweistellige Operationen: natürlicher Verbund *(Natural-Join)*

Syntax: $r_1(\mathcal{R}_1) \bowtie r_2(\mathcal{R}_2)$

Semantik: Alle Tupel, die bei Projektion auf die $\mathcal{R}_i$ ein Tupel aus r_i liefern:
$\{t \mid t \in r(\mathcal{R}_1 \cup \mathcal{R}_2) \ \wedge \ t(\mathcal{R}_1) \in r_1 \ \wedge \ t(\mathcal{R}_2) \in r_2\}$ [kartesisches Produkt]

Relationale Algebra

Weitere zweistellige Operationen: natürlicher Verbund *(Natural-Join)*

Syntax: $r_1(\mathcal{R}_1) \bowtie r_2(\mathcal{R}_2)$

Semantik: Alle Tupel, die bei Projektion auf die $\mathcal{R}_i$ ein Tupel aus r_i liefern:
$\{t \mid t \in r(\mathcal{R}_1 \cup \mathcal{R}_2) \ \wedge \ t(\mathcal{R}_1) \in r_1 \ \wedge \ t(\mathcal{R}_2) \in r_2\}$ [kartesisches Produkt]

Beispiel:

Ausleihe	
InvNr	Name
4711	Meyer
1201	Schulz
0007	Müller
4712	Meyer

Buch			
InvNr	Titel	ISBN	Autor
0007	Dr. No	3-125	James Bond
1201	Objektbanken	3-111	Heuer
4711	Datenbanken	3-765	Vossen
4712	Datenbanken	3-891	Ullman
4717	Pascal	3-999	Wirth

Ausleihe $\bowtie$ Buch $\leadsto$

Name	InvNr	Titel	ISBN	Autor
Müller	0007	Dr. No	3-125	James Bond
Schulz	1201	Objektbanken	3-111	Heuer
Meyer	4711	Datenbanken	3-765	Vossen
Meyer	4712	Datenbanken	3-891	Ullman

Relationale Algebra

Weitere zweistellige Operationen: natürlicher Verbund *(Natural-Join)*

Syntax: $r_1(\mathcal{R}_1) \bowtie r_2(\mathcal{R}_2)$

Semantik: Alle Tupel, die bei Projektion auf die $\mathcal{R}_i$ ein Tupel aus r_i liefern:
$\{t \mid t \in r(\mathcal{R}_1 \cup \mathcal{R}_2) \ \wedge \ t(\mathcal{R}_1) \in r_1 \ \wedge \ t(\mathcal{R}_2) \in r_2\}$ [kartesisches Produkt]

$$\overbrace{A_1, \ldots, A_n, B_1, \ldots, B_k}^{\mathcal{R}_1}$$

$$\underbrace{B_1, \ldots, B_k, C_1, \ldots, C_m}_{\mathcal{R}_2}$$

$$r_1(\mathcal{R}_1) \bowtie r_2(\mathcal{R}_2) = \underbrace{\pi_{A_1,\ldots,A_n,\ r_1.B_1,\ldots,r_1.B_k,\ C_1,\ldots,C_m}}_{\text{Projektion}} \big(\underbrace{\sigma_{r_1.B_1=r_2.B_1,\ldots,r_1.B_k=r_2.B_k}}_{\text{Selektion}} \underbrace{(r_1 \times r_2)}_{\text{kartesisches Produkt}}\big)$$

Relationale Algebra

Weitere zweistellige Operationen: natürlicher Verbund *(Natural-Join)*

Syntax: $r_1(\mathcal{R}_1) \bowtie r_2(\mathcal{R}_2)$

Semantik: Alle Tupel, die bei Projektion auf die $\mathcal{R}_i$ ein Tupel aus r_i liefern:

$\{t \mid t \in r(\mathcal{R}_1 \cup \mathcal{R}_2) \;\wedge\; t(\mathcal{R}_1) \in r_1 \;\wedge\; t(\mathcal{R}_2) \in r_2\}$ [kartesisches Produkt]

$$\overbrace{A_1, \ldots, A_n, B_1, \ldots, B_k}^{\mathcal{R}_1}$$

$$\underbrace{B_1, \ldots, B_k, C_1, \ldots, C_m}_{\mathcal{R}_2}$$

$$r_1(\mathcal{R}_1) \bowtie r_2(\mathcal{R}_2) = \underbrace{\pi_{A_1,\ldots,A_n,\; r_1.B_1,\ldots,r_1.B_k,\; C_1,\ldots,C_m}}_{\text{Projektion}} \big(\underbrace{\sigma_{r_1.B_1=r_2.B_1,\ldots,r_1.B_k=r_2.B_k}}_{\text{Selektion}} \underbrace{(r_1 \times r_2)}_{\text{kartesisches Produkt}}\big)$$

$r_1(\mathcal{R}_1) \bowtie r_2(\mathcal{R}_2)$		
$\mathcal{R}_1 - \mathcal{R}_2$	$\mathcal{R}_1 \cap \mathcal{R}_2$	$\mathcal{R}_2 - \mathcal{R}_1$

Relationale Algebra

Weitere zweistellige Operationen: natürlicher Verbund (Fortsetzung)

Eigenschaften des natürlichen Verbunds:

- Kommutativität: $r_1 \bowtie r_2 \ = \ r_2 \bowtie r_1$
- Assoziativität: $(r_1 \bowtie r_2) \bowtie r_3 \ = \ r_1 \bowtie (r_2 \bowtie r_3)$
  Somit ist folgende Schreibweise möglich: $r_1 \bowtie r_2 \bowtie \ldots \bowtie r_p \ \equiv \bowtie_{i=1}^{p} r_i$
- $\mathcal{R}_1 \cap \mathcal{R}_2 = \emptyset \quad \Rightarrow \quad r_1 \bowtie r_2 \ = \ r_1 \times r_2$

Relationale Algebra

Weitere zweistellige Operationen: natürlicher Verbund (Fortsetzung)

Eigenschaften des natürlichen Verbunds:

- Kommutativität: $r_1 \bowtie r_2 \;=\; r_2 \bowtie r_1$
- Assoziativität: $(r_1 \bowtie r_2) \bowtie r_3 \;=\; r_1 \bowtie (r_2 \bowtie r_3)$
  Somit ist folgende Schreibweise möglich: $r_1 \bowtie r_2 \bowtie \ldots \bowtie r_p \;\equiv \bowtie_{i=1}^{p} r_i$
- $\mathcal{R}_1 \cap \mathcal{R}_2 = \emptyset \quad \Rightarrow \quad r_1 \bowtie r_2 \;=\; r_1 \times r_2$

Beispiel „3-Wege-Join":

$$(\text{Ausleihe} \bowtie \text{Buch}) \bowtie \text{Verlag} \;=\; \text{Ausleihe} \bowtie (\text{Buch} \bowtie \text{Verlag})$$

Beispiel für Entartung zum kartesischen Produkt:

$$\pi_{\text{Autor}}(\text{Buch}) \bowtie \pi_{\text{InvNr}}(\text{Ausleihe})$$

Bemerkungen:

- Die gemeinsamen Attribute der bei einem Join beteiligten Relationen werden auch als „Join-Attribute“ bezeichnet.
- Die Umbennungsoperation ρ ermöglicht es, Relationen über zwei Attribute zu verbinden, welche die gleiche Bedeutung (Semantik), aber einen unterschiedlichen Namen haben.
- Tupel, die keinen Join-Partner finden, sogenannte „Dangling Tuples“, werden eliminiert. Folglich ist die Projektion im Allgemeinen nicht die inverse Operation zum natürlichen Verbund. Es gilt: $\pi_{\mathcal{R}_1}(r_1 \bowtie r_2) \subseteq r_1$
- Der natürliche Verbund ist im Allgemeinen nicht die inverse Operation zu zwei Projektionen. Sei r eine Relation über $\mathcal{R}$ und mit $\mathcal{R} = \mathcal{R}_1 \cup \mathcal{R}_2$. Dann gilt der folgende Zusammenhang nur bei *Verbundtreue*: $\pi_{\mathcal{R}_1}(r) \bowtie \pi_{\mathcal{R}_2}(r) = r$

Relationale Algebra

Weitere zweistellige Operationen: allgemeiner Verbund *(Theta-Join)*

Der allgemeine Join-Operator, $\bowtie_\theta$, erlaubt die Spezifikation eines beliebigen Join-Prädikates θ. Das Ergebnis des Theta-Joins enthält *alle* (bei Namensgleichheit qualifizierten) Attribute der beteiligten Relationen:

$$r_1 \bowtie_\theta r_2 \; = \; \sigma_\theta(r_1 \times r_2)$$

Beispiel:

$$r_1 \bowtie_{A_1 > A_2 \,\wedge\, A_3 = A_4 \,\wedge\, A_5 < A_6} r_2$$

Bemerkungen:

- Einen Theta-Join der Form $r_1 \bowtie_{A_{1_1}=A_{1_2},\ldots,A_{1_n}=A_{2_n}} r_2$ nennt man auch Equi-Join. Im Unterschied zum Natural-Join werden beim Equi-Join alle Attribute übernommen.

- Die bislang eingeführten Join-Operatoren werden auch *innere* Joins genannt. Für sie gilt, dass diejenigen Tupel der Argumentrelationen verloren gehen, die keinen Join-Partner gefunden haben.

- Mit den *äußeren* Join-Operatoren können auch partnerlose Tupel der Argumentrelationen in die Ergebnisrelation übernommen werden: bei Anwendung des Left-Outer-Join bleiben die Tupel der linken Argumentrelation immer erhalten, bei Anwendung des Right-Outer-Join die Tupel der rechten Argumentrelation. Die nicht gegebenen Attributwerte der partnerlosen Tupel werden mit Nullwerten, in Zeichen: $\perp$, aufgefüllt.

Relationale Algebra

Weitere zweistellige Operationen: äußerer Verbund *(Outer-Join)*

- Natural-Join:

r_1		
A	B	C
a_1	b_1	c_1
a_2	b_2	c_2

$\bowtie$

r_2		
C	D	E
c_1	d_1	e_1
c_3	d_2	e_2

$\rightsquigarrow$

$r_1 \bowtie r_2$				
A	B	C	D	E
a_1	b_1	c_1	d_1	e_1

- Left-Outer-Join:

r_1		
A	B	C
a_1	b_1	c_1
a_2	b_2	c_2

$⟕$

r_2		
C	D	E
c_1	d_1	e_1
c_3	d_2	e_2

$\rightsquigarrow$

$r_1 ⟕ r_2$				
A	B	C	D	E
a_1	b_1	c_1	d_1	e_1
a_2	b_2	c_2	$\perp$	$\perp$

Relationale Algebra

Weitere zweistellige Operationen: äußerer Verbund *(Outer-Join)*

- Natural-Join:

r_1		
A	B	C
a_1	b_1	c_1
a_2	b_2	c_2

$\bowtie$

r_2		
C	D	E
c_1	d_1	e_1
c_3	d_2	e_2

$\leadsto$

$r_1 \bowtie r_2$				
A	B	C	D	E
a_1	b_1	c_1	d_1	e_1

- Left-Outer-Join:

r_1		
A	B	C
a_1	b_1	c_1
a_2	b_2	c_2

$⟕$

r_2		
C	D	E
c_1	d_1	e_1
c_3	d_2	e_2

$\leadsto$

$r_1 ⟕ r_2$				
A	B	C	D	E
a_1	b_1	c_1	d_1	e_1
a_2	b_2	c_2	$\perp$	$\perp$

- Right-Outer-Join:

r_1		
A	B	C
a_1	b_1	c_1
a_2	b_2	c_2

$⟖$

r_2		
C	D	E
c_1	d_1	e_1
c_3	d_2	e_2

$\leadsto$

$r_1 ⟖ r_2$				
A	B	C	D	E
a_1	b_1	c_1	d_1	e_1
$\perp$	$\perp$	c_3	d_2	e_2

Relationale Algebra

Weitere zweistellige Operationen: äußerer Verbund *(Outer-Join)*

- Natural-Join:

r_1		
A	B	C
a_1	b_1	c_1
a_2	b_2	c_2

$\bowtie$

r_2		
C	D	E
c_1	d_1	e_1
c_3	d_2	e_2

$\rightsquigarrow$

$r_1 \bowtie r_2$				
A	B	C	D	E
a_1	b_1	c_1	d_1	e_1

- Left-Outer-Join:

r_1		
A	B	C
a_1	b_1	c_1
a_2	b_2	c_2

$⟕$

r_2		
C	D	E
c_1	d_1	e_1
c_3	d_2	e_2

$\rightsquigarrow$

$r_1 ⟕ r_2$				
A	B	C	D	E
a_1	b_1	c_1	d_1	e_1
a_2	b_2	c_2	$\perp$	$\perp$

- Right-Outer-Join:

r_1		
A	B	C
a_1	b_1	c_1
a_2	b_2	c_2

$⟖$

r_2		
C	D	E
c_1	d_1	e_1
c_3	d_2	e_2

$\rightsquigarrow$

$r_1 ⟖ r_2$				
A	B	C	D	E
a_1	b_1	c_1	d_1	e_1
$\perp$	$\perp$	c_3	d_2	e_2

- Outer-Join:

r_1		
A	B	C
a_1	b_1	c_1
a_2	b_2	c_2

$⟗$

r_2		
C	D	E
c_1	d_1	e_1
c_3	d_2	e_2

$\rightsquigarrow$

$r_1 ⟗ r_2$				
A	B	C	D	E
a_1	b_1	c_1	d_1	e_1
a_2	b_2	c_2	$\perp$	$\perp$
$\perp$	$\perp$	c_3	d_2	e_2

Relationale Algebra

Weitere zweistellige Operationen: Semi-Verbund *(Semi-Join)*

Die Semi-Verbundoperatoren projizieren die Tupel der Ergebnisrelation eines Natural-Join auf das Schema einer der Ausgangsrelationen:

$$r_1 \ltimes r_2 = \pi_{\mathcal{R}_1}(r_1 \bowtie r_2) \quad \text{bzw.} \quad r_1 \rtimes r_2 = \pi_{\mathcal{R}_2}(r_1 \bowtie r_2)$$

Relationale Algebra

Weitere zweistellige Operationen: Semi-Verbund *(Semi-Join)*

Die Semi-Verbundoperatoren projizieren die Tupel der Ergebnisrelation eines Natural-Join auf das Schema einer der Ausgangsrelationen:

$$r_1 \ltimes r_2 = \pi_{\mathcal{R}_1}(r_1 \bowtie r_2) \quad \text{bzw.} \quad r_1 \rtimes r_2 = \pi_{\mathcal{R}_2}(r_1 \bowtie r_2)$$

- Semi-Join von r_1 mit r_2:

r_1		
A	B	C
a_1	b_1	c_1
a_2	b_2	c_2

$\ltimes$

r_2		
C	D	E
c_1	d_1	e_1
c_3	d_2	e_2

$\leadsto$

$r_1 \ltimes r_2$		
A	B	C
a_1	b_1	c_1

- Semi-Join von r_2 mit r_1:

r_1		
A	B	C
a_1	b_1	c_1
a_2	b_2	c_2

$\rtimes$

r_2		
C	D	E
c_1	d_1	e_1
c_3	d_2	e_2

$\leadsto$

$r_1 \rtimes r_2$		
C	D	E
c_1	d_1	e_1

Es gilt folgende Äquivalenz: $r_1 \rtimes r_2 = r_2 \ltimes r_1$.

Relationale Algebra

Weitere zweistellige Operationen: relationale Division

Die bisher betrachteten Anfragebeispiele liefern diejenigen Tupel, die eine bestimmte Selektionsbedingung erfüllen.

Frage: Wie bestimmt man diejenigen Tupel, die *alle* Bedingungen – im Sinne von *gleichzeitig* – einer Menge von Selektionsbedingungen erfüllen?

Relationale Algebra

Weitere zweistellige Operationen: relationale Division

Die bisher betrachteten Anfragebeispiele liefern diejenigen Tupel, die eine bestimmte Selektionsbedingung erfüllen.

Frage: Wie bestimmt man diejenigen Tupel, die *alle* Bedingungen – im Sinne von *gleichzeitig* – einer Menge von Selektionsbedingungen erfüllen?

Beispiel:

Buecher	
Titel	Verlag
Harry Potter	Princeton
Heuristics	Addison
Glücksformel	dpunkt
Datenbanken	Springer

Buchhaendler		
Name	Stadt	PLZ
Lehmann	Berlin	99011
Meiersche	Aachen	42100
Amazon	Köln	52100

Angebote	
Titel	Haendler
Harry Potter	Lehmann
Harry Potter	Meiersche
Harry Potter	Amazon
Datenbanken	Amazon
Glücksformel	Amazon
Glücksformel	Lehmann

Anfragen:

1. „Welche Titel sind bei *allen* Buchhändlern im Angebot?“
2. Nicht zu verwechseln mit „Welche Titel befinden sich (alle) im Angebot?“

Relationale Algebra

Weitere zweistellige Operationen: relationale Division (Fortsetzung)

Syntax: $r_1(\mathcal{R}_1) \div r_2(\mathcal{R}_2)$

Semantik: Sei $\mathcal{R}_2 \subseteq \mathcal{R}_1$ und $\mathcal{R} = \mathcal{R}_1 - \mathcal{R}_2$. Dann ist $r_1 \div r_2$ definiert als:

$$\{t \mid \forall t_2 \in r_2 \;\; \exists t_1 \in r_1 \; : \; t_1(\mathcal{R}) = t \;\wedge\; t_1(\mathcal{R}_2) = t_2\}$$

Beispiel:

r_1	
A	B
a_1	b_1
a_1	b_2
a_1	b_3
a_2	b_2
a_2	b_3

$\div$

r_2
B
b_1
b_2

$=$

$r_1 \div r_2$
A
a_1

Relationale Algebra

Rekursiver Abschluss

Der rekursive Abschluss kann mit Mitteln der relationalen Algebra nicht ausgedrückt werden.

Beispiel: $\leadsto \mathcal{TAFEL}$

Relationale Algebra

Übersicht über Operationen

Operation	Argumente	Notation
SELECT	Relation r, Auswahlbedingung <COND>	$\sigma_{<\text{COND}>}(r)$
PROJECT	Relation r, Attributliste α	$\pi_\alpha(r)$
RENAME	Relation r, Attributzuordnungen <Mapping> Relation r, Relationenname s	$\rho_{<\text{Mapping}>}(r)$ $\rho_s(r)$
NATURAL JOIN	Relationen r_1, r_2	$r_1 \bowtie r_2$
THETA JOIN	Relationen r_1, r_2, Verbundbedingung θ	$r_1 \bowtie_\theta r_2$
EQUI JOIN	Relationen r_1, r_2, $=$-Verbundbedingung θ	$r_1 \bowtie_\theta r_2$
OUTER JOIN (LEFT/RIGHT/FULL)	Relationen r_1, r_2	$r_1 \leftouterjoin r_2$, $r_1 \rightouterjoin r_2$, $r_1 \fullouterjoin r_2$
SEMI JOIN (LEFT/RIGHT)	Relationen r_1, r_2	$r_1 \ltimes r_2$, $r_1 \rtimes r_2$
UNION	Relationen r_1, r_2	$r_1 \cup r_2$
INTERSECTION	Relationen r_1, r_2	$r_1 \cap r_2$
DIFFERENCE	Relationen r_1, r_2	$r_1 - r_2$
CARTESIAN PRODUCT	Relationen r_1, r_2	$r_1 \times r_2$
DIVISION	Relationen r_1, r_2	$r_1 \div r_2$

Relationale Algebra

Übersicht über Operationen

Operation	Argumente	Notation
SELECT	Relation r, Auswahlbedingung <COND>	$\sigma_{<\text{COND}>}(r)$
PROJECT	Relation r, Attributliste α	$\pi_\alpha(r)$
RENAME	Relation r, Attributzuordnungen <Mapping> Relation r, Relationenname s	$\rho_{<\text{Mapping}>}(r)$ $\rho_s(r)$
NATURAL JOIN	Relationen r_1, r_2	$r_1 \bowtie r_2$
THETA JOIN	Relationen r_1, r_2, Verbundbedingung θ	$r_1 \bowtie_\theta r_2$
EQUI JOIN	Relationen r_1, r_2, $=$-Verbundbedingung θ	$r_1 \bowtie_\theta r_2$
OUTER JOIN (LEFT/RIGHT/FULL)	Relationen r_1, r_2	$r_1 ⟕ r_2$, $r_1 ⟖ r_2$, $r_1 ⟗ r_2$
SEMI JOIN (LEFT/RIGHT)	Relationen r_1, r_2	$r_1 \ltimes r_2$, $r_1 \rtimes r_2$
UNION	Relationen r_1, r_2	$r_1 \cup r_2$
INTERSECTION	Relationen r_1, r_2	$r_1 \cap r_2$
DIFFERENCE	Relationen r_1, r_2	$r_1 - r_2$
CARTESIAN PRODUCT	Relationen r_1, r_2	$r_1 \times r_2$
DIVISION	Relationen r_1, r_2	$r_1 \div r_2$

Relationale Algebra

Übersicht über Operationen

Operation	Argumente	Notation
SELECT	Relation r, Auswahlbedingung <COND>	$\sigma_{<\text{COND}>}(r)$
PROJECT	Relation r, Attributliste α	$\pi_{\alpha}(r)$
RENAME	Relation r, Attributzuordnungen <Mapping> Relation r, Relationenname s	$\rho_{<\text{Mapping}>}(r)$ $\rho_s(r)$
NATURAL JOIN	Relationen r_1, r_2	$r_1 \bowtie r_2$
THETA JOIN	Relationen r_1, r_2, Verbundbedingung θ	$r_1 \bowtie_{\theta} r_2$
EQUI JOIN	Relationen r_1, r_2, $=$-Verbundbedingung θ	$r_1 \bowtie_{\theta} r_2$
OUTER JOIN (LEFT/RIGHT/FULL)	Relationen r_1, r_2	$r_1 ⟕ r_2$, $r_1 ⟖ r_2$, $r_1 ⟗ r_2$
SEMI JOIN (LEFT/RIGHT)	Relationen r_1, r_2	$r_1 \ltimes r_2$, $r_1 \rtimes r_2$
UNION	Relationen r_1, r_2	$r_1 \cup r_2$
INTERSECTION	Relationen r_1, r_2	$r_1 \cap r_2$
DIFFERENCE	Relationen r_1, r_2	$r_1 - r_2$
CARTESIAN PRODUCT	Relationen r_1, r_2	$r_1 \times r_2$
DIVISION	Relationen r_1, r_2	$r_1 \div r_2$

Kapitel DB:V (Fortsetzung)

Anfragekalküle

Paradigmen

- Anfragealgebren spiegeln das Konzept von abstrakten Datenstrukturen wider; der Datentyp ist die Relation mit entsprechenden Operationen hierauf.

  Ein relationaler Ausdruck ist eine *prozedurale* Beschreibung, also eine genau festgelegte Folge von Operationen zur Berechnung einer Anfrage.

- Anfragekalküle sind ein logikbasierter Ansatz zur Beschreibung der Ergebnismenge einer Anfrage.

  Sie können als *deklarative* bzw. *nicht-prozedurale* Sprache aufgefasst werden. Insbesondere enthält eine Formel des Kalküls keine Information darüber, *wie* sie auszuwerten ist.

  Für das relationale Modell betrachtet man folgende Kalküle:

  1. relationaler Tupelkalkül
  2. relationaler Domänenkalkül, auch Bereichskalkül genannt

Anfragekalküle

Aufbau einer Formel / Grammatik der Sprache / Syntax (Teil I)

Sei Σ eine Menge von Atomen aus einem Anfragekalkül, dann sind folgende Ausdrücke Formeln in diesem Kalkül:

1. Jedes Atom in Σ ist eine Formel.
2. Sind α und β Formeln, so sind es auch (α), $\neg\alpha$, $\alpha \wedge \beta$, $\alpha \vee \beta$ und $\alpha \rightarrow \beta$.

Anfragekalküle

Aufbau einer Formel / Grammatik der Sprache / Syntax (Teil I)

Sei Σ eine Menge von Atomen aus einem Anfragekalkül, dann sind folgende Ausdrücke Formeln in diesem Kalkül:

1. Jedes Atom in Σ ist eine Formel.
2. Sind α und β Formeln, so sind es auch (α), $\neg\alpha$, $\alpha \wedge \beta$, $\alpha \vee \beta$ und $\alpha \rightarrow \beta$.

Beispiele für Formeln:

- $\text{Mitarbeiter}(t_1)$ — Aussageform, Atom
- $\text{Mitarbeiter}(\text{Smith}, 1234, \text{Weimar}, 3334, 4)$ — Aussage, Atom
- $\exists t_1 : \text{Mitarbeiter}(t_1) \ \wedge \ t_1.\text{AbtNr} = 5$ — Aussage, keine atomare Formel
- $\text{Mitarbeiter}(x_1, x_2, \text{Weimar}, x_4, x_5)$ — Aussageform, Atom

Mitarbeiter				
Name	PersNr	Wohnort	ChefPersNr	AbtNr
Smith	1234	Weimar	3334	5
Wong	3334	Köln	8886	5
Zelaya	9998	Erfurt	9876	4

Anfragekalküle

Bewertung einer Formel / Interpretation der Sprache / Semantik (Teil I)

1. Auf Basis des Datenbankzustandes $d(\boldsymbol{\mathcal{R}}) = \{r_1, \ldots, r_p\}$ kann einem Atom in Σ ein Wahrheitswert zugewiesen werden.

2. Auf Basis der Wahrheitswerte der Atome lässt sich rekursiv gemäß der üblichen Semantik für $\neg, \wedge, \vee, \rightarrow$ einer Formel α ein Wahrheitswert $\mathcal{I}(\alpha)$ zuordnen.

Anfragekalküle

Bewertung einer Formel / Interpretation der Sprache / Semantik (Teil I)

1. Auf Basis des Datenbankzustandes $d(\boldsymbol{\mathcal{R}}) = \{r_1, \ldots, r_p\}$ kann einem Atom in Σ ein Wahrheitswert zugewiesen werden.

2. Auf Basis der Wahrheitswerte der Atome lässt sich rekursiv gemäß der üblichen Semantik für $\neg, \wedge, \vee, \rightarrow$ einer Formel α ein Wahrheitswert $\mathcal{I}(\alpha)$ zuordnen.

Beispiele:

- $\mathcal{I}(\underbrace{\text{Mitarbeiter}(\text{Smith}, 1234, \text{Weimar}, 3334, 4)}_{\alpha}) = 0$

- $\mathcal{I}(\underbrace{\exists t_1 : \text{Mitarbeiter}(t_1) \ \wedge \ t_1.\text{AbtNr} = 5}_{\alpha}) = 1$

Mitarbeiter				
Name	PersNr	Wohnort	ChefPersNr	AbtNr
Smith	1234	Weimar	3334	5
Wong	3334	Köln	8886	5
Zelaya	9998	Erfurt	9876	4

Bemerkungen:

- Ein Atom ist die einfachste Formel. Eine Formel stellt entweder eine Aussage oder eine Aussage*form* dar. Von einer Aussage lässt sich feststellen, ob sie wahr oder falsch ist; von einer Aussageform lässt sich nicht die Wahrheit bzw. Unwahrheit feststellen.

- In einem Anfragekalkül geschieht die Feststellung des Wahrheitswertes (= Interpretation $\mathcal{I}$, Semantik) einer atomaren Aussage auf Basis des Datenbankzustandes: das Atom entspricht dem Test, ob ein bestimmtes Wertetupel t oder ein bestimmter Attributwert x ein Element einer Relation r im Datenbankzustand $d(\boldsymbol{\mathcal{R}}) = \{r_1, \ldots, r_p\}$ ist.

- Der Wahrheitswert (= Interpretation $\mathcal{I}$, Semantik) einer komplexen Formel leitet sich in eindeutiger Weise von den Wahrheitswerten der Atome der Formel ab. Dabei ist die Verknüpfung von Wahrheitswerten mit den Junktoren (= Operatoren) $\neg, \wedge, \vee, \rightarrow$ wie folgt definiert:

$$
\begin{array}{cc|c}
 & \neg & \\
\hline
\alpha & 0 & 1 \\
 & 1 & 0
\end{array}
\qquad
\begin{array}{cc|ccc}
 & \wedge & \beta & 0 & 1 \\
\hline
\alpha & 0 & & 0 & 0 \\
 & 1 & & 0 & 1
\end{array}
\qquad
\begin{array}{cc|ccc}
 & \vee & \beta & 0 & 1 \\
\hline
\alpha & 0 & & 0 & 1 \\
 & 1 & & 1 & 1
\end{array}
\qquad
\begin{array}{cc|ccc}
 & \rightarrow & \beta & 0 & 1 \\
\hline
\alpha & 0 & & 1 & 1 \\
 & 1 & & 0 & 1
\end{array}
$$

- Die Interpretationsfunktion wird mit $\mathcal{I}$ bezeichnet und liefert für eine Formel α, die eine Aussage darstellt, ihren Wahrheitswert: $\alpha \mapsto \mathcal{I}(\alpha)$, $\mathcal{I}(\alpha) \in \{0, 1\}$.

- Mit Formeln werden die Bedingungen einer Datenbankanfrage nachgebildet.

Anfragekalküle

Freie und gebundene Variablen

Sei α eine Formel, die eine Variable x enthält. Dann sei vereinbart:

(a) Ist α ein Atom, so ist x eine *freie* Variable.

(b) Das Vorkommen von x in (α), $\neg\alpha$, $\alpha \wedge \beta$, $\alpha \vee \beta$ und $\alpha \rightarrow \beta$ ist *frei* oder *gebunden* – abhängig davon, ob es in α frei oder gebunden ist.

(c) Alle freien Vorkommen von x in α sind gebunden in $\exists x \alpha$ und $\forall x \alpha$.

(d) In keiner Formel darf eine Variable sowohl frei als auch gebunden auftreten.

Anfragekalküle

Freie und gebundene Variablen

Sei α eine Formel, die eine Variable x enthält. Dann sei vereinbart:

(a) Ist α ein Atom, so ist x eine *freie* Variable.

(b) Das Vorkommen von x in (α), $\neg\alpha$, $\alpha \wedge \beta$, $\alpha \vee \beta$ und $\alpha \rightarrow \beta$ ist *frei* oder *gebunden* – abhängig davon, ob es in α frei oder gebunden ist.

(c) Alle freien Vorkommen von x in α sind gebunden in $\exists x \alpha$ und $\forall x \alpha$.

(d) In keiner Formel darf eine Variable sowohl frei als auch gebunden auftreten.

Anfragekalküle

Freie und gebundene Variablen

Sei α eine Formel, die eine Variable x enthält. Dann sei vereinbart:

(a) Ist α ein Atom, so ist x eine *freie* Variable.

(b) Das Vorkommen von x in (α), $\neg\alpha$, $\alpha \wedge \beta$, $\alpha \vee \beta$ und $\alpha \rightarrow \beta$ ist *frei* oder *gebunden* – abhängig davon, ob es in α frei oder gebunden ist.

(c) Alle freien Vorkommen von x in α sind gebunden in $\exists x\alpha$ und $\forall x\alpha$.

(d) In keiner Formel darf eine Variable sowohl frei als auch gebunden auftreten.

Anfragekalküle

Freie und gebundene Variablen

Sei α eine Formel, die eine Variable x enthält. Dann sei vereinbart:

(a) Ist α ein Atom, so ist x eine *freie* Variable.

(b) Das Vorkommen von x in (α), $\neg\alpha$, $\alpha \wedge \beta$, $\alpha \vee \beta$ und $\alpha \rightarrow \beta$ ist *frei* oder *gebunden* – abhängig davon, ob es in α frei oder gebunden ist.

(c) Alle freien Vorkommen von x in α sind gebunden in $\exists x\alpha$ und $\forall x\alpha$.

(d) In keiner Formel darf eine Variable sowohl frei als auch gebunden auftreten.

Anfragekalküle

Freie und gebundene Variablen

Sei α eine Formel, die eine Variable x enthält. Dann sei vereinbart:

(a) Ist α ein Atom, so ist x eine *freie* Variable.

(b) Das Vorkommen von x in (α), $\neg\alpha$, $\alpha \wedge \beta$, $\alpha \vee \beta$ und $\alpha \rightarrow \beta$ ist *frei* oder *gebunden* – abhängig davon, ob es in α frei oder gebunden ist.

(c) Alle freien Vorkommen von x in α sind gebunden in $\exists x\alpha$ und $\forall x\alpha$.

(d) In keiner Formel darf eine Variable sowohl frei als auch gebunden auftreten.

Aufbau einer Formel / Grammatik der Sprache / Syntax (Teil II)

3. Ist α eine Formel, so sind es auch $\exists x\alpha$ und $\forall x\alpha$ – wobei x eine Variable ist, die in α frei vorkommt.

Bewertung einer Formel / Interpretation der Sprache / Semantik (Teil II)

3. Eine Formel $\exists x\alpha$ ist wahr, falls α bzgl. *einer* Instanziierung von x wahr wird. Eine Formel $\forall x\alpha$ ist wahr, falls α bzgl. *aller* Instanziierungen von x wahr wird. Das Universum definiert die Menge der möglichen Instanziierungen.

Bemerkungen:

- Eine Formel mit freien Variablen stellt eine Aussageform dar. Eine Formel, die keine oder nur gebundene (= instanziierte) Variablen enthält, stellt eine Aussage dar.

- Die für eine Instanziierung zur Verfügung stehenden Wertetupel t bzw. Attributwerte x stammen aus einem Grundbereich, auch *Universum* genannt. Das Universum enthält alle möglichen Tupel bzw. Attributwerte, die in einem Datenbankzustand vorliegen können. Im Allgemeinen enthält das Universum unendlich viele Elemente.

Anfragekalküle

Auswertung einer Anfrage

Gegeben:

- Anfrage $\{(\_) \mid \alpha\}$ mit freien Variablen $(\_)$ und Formel α
- Datenbankzustand $d(\boldsymbol{\mathcal{R}}) = \{r_1, \ldots, r_p\}$

Anfragekalküle

Auswertung einer Anfrage

Gegeben:

- Anfrage $\{(\_) \mid \alpha\}$ mit freien Variablen $(\_)$ und Formel α
- Datenbankzustand $d(\boldsymbol{\mathcal{R}}) = \{r_1, \ldots, r_p\}$

Konstruktion der Ergebnisrelation *res* für Anfrage $\{(\_) \mid \alpha\}$ unter $d(\boldsymbol{\mathcal{R}})$:

1. *res* $= \emptyset$
2. Die freien Variablen $(\_)$ werden hinsichtlich aller Tupel (im Tupelkalkül) bzw. aller Attributwerte (im Domänenkalkül) für die in der Datenbank befindlichen Relationen $\{r_1, \ldots, r_p\}$ instanziiert.

   Durch die Instanziierung wird die Aussageform α zu einer Aussage.
3. Für jede Instanziierung von $(\_)$ wird geprüft, ob die Formel α wahr (erfüllt) ist. Falls ja, setze *res* $=$ *res* $\cup$ {Instanziierung von $(\_)$}
4. *res* enthält keine weiteren Elemente.

Anfragekalküle

Auswertung einer Anfrage

Gegeben:

- Anfrage $\{(\_) \mid \alpha\}$ mit freien Variablen $(\_)$ und Formel α
- Datenbankzustand $d(\boldsymbol{\mathcal{R}}) = \{r_1, \ldots, r_p\}$

Konstruktion der Ergebnisrelation *res* für Anfrage $\{(\_) \mid \alpha\}$ unter $d(\boldsymbol{\mathcal{R}})$:

1. *res* $= \emptyset$
2. Die freien Variablen $(\_)$ werden hinsichtlich aller Tupel (im Tupelkalkül) bzw. aller Attributwerte (im Domänenkalkül) für die in der Datenbank befindlichen Relationen $\{r_1, \ldots, r_p\}$ instanziiert.

   Durch die Instanziierung wird die Aussageform α zu einer Aussage.
3. Für jede Instanziierung von $(\_)$ wird geprüft, ob die Formel α wahr (erfüllt) ist. Falls ja, setze *res* $=$ *res* $\cup$ $\{$Instanziierung von $(\_)\}$
4. *res* enthält keine weiteren Elemente.

Anfragekalküle

Sichere Anfragen

Unter (semantisch) sicheren Anfragen versteht man Formeln eines Anfragekalküls, die für jeden Datenbankzustand $d(\boldsymbol{\mathcal{R}}) = \{r_1, \ldots, r_p\}$ nur für eine endliche Menge von Variableninstanziierungen erfüllt sind.

Beispiel für eine nicht-sichere Anfrage:

$\neg\, r(x)$: „Alle Instanzen von x (im Universum), die nicht in $r \in d(\boldsymbol{\mathcal{R}})$ sind.“

Durch die Forderung bestimmter syntaktischer Einschränkungen kann man die semantische Sicherheit für eine Teilmenge der semantisch sicheren Anfragen auf einfache Art bestimmen. Stichwort: Domäne einer Formel

Relationaler Tupelkalkül

Konzepte [Domänenkalkül]

1. Tupelvariablen, die sich auf Relationen $r_i \in d(\boldsymbol{\mathcal{R}})$, $d(\boldsymbol{\mathcal{R}}) = \{r_1, \ldots, r_p\}$, beziehen und mit jedem Tupel aus r_i instanziiert werden können.

2. Formeln, mit denen sich auf Basis der Tupelvariablen Zusammenhänge zwischen Attributen formulieren lassen.

Relationaler Tupelkalkül

Anfragen [Domänenkalkül]

Anfrage im relationalen Tupelkalkül mit Variablen $t_1, t_2, \ldots, t_n, \; t_{n+1}, \ldots, t_{n+m}$:

$$\{t \mid \alpha\} \qquad \text{allgemein}: \quad \{(t_1.A_1, t_2.A_2, \ldots, t_n.A_n) \mid \alpha\}$$

- $t_1, \ldots, t_n$ sind freie, $t_{n+1}, \ldots, t_{n+m}$ sind gebundene Tupelvariablen.
- $A_1, \ldots, A_n$ sind Attribute der Relationen bzgl. derer die t_i instanziiert sind.
- α ist eine logische Formel, wobei die Menge der Atome Σ, aus denen α besteht, wie folgt definiert ist:

Relationaler Tupelkalkül

Anfragen [Domänenkalkül]

Anfrage im relationalen Tupelkalkül mit Variablen $t_1, t_2, \ldots, t_n,\ t_{n+1}, \ldots, t_{n+m}$:

$$\{t \mid \alpha\} \qquad \text{allgemein}: \quad \{(t_1.A_1, t_2.A_2, \ldots, t_n.A_n) \mid \alpha\}$$

- $t_1, \ldots, t_n$ sind freie, $t_{n+1}, \ldots, t_{n+m}$ sind gebundene Tupelvariablen.
- $A_1, \ldots, A_n$ sind Attribute der Relationen bzgl. derer die t_i instanziiert sind.
- α ist eine logische Formel, wobei die Menge der Atome Σ, aus denen α besteht, wie folgt definiert ist:

1. „$r(t)$“ alternativ: „$t \in r$“
   ist ein Atom, wobei t eine Tupelvariable und r eine Relation bezeichnet. $r(t)$ ist wahr für eine Instanziierung von t, falls diese Instanziierung ein Tupel in r ist.

Relationaler Tupelkalkül

Anfragen [Domänenkalkül]

Anfrage im relationalen Tupelkalkül mit Variablen $t_1, t_2, \ldots, t_n,\ t_{n+1}, \ldots, t_{n+m}$:

$$\{t \mid \alpha\} \qquad \text{allgemein}: \quad \{(t_1.A_1, t_2.A_2, \ldots, t_n.A_n) \mid \alpha\}$$

- $t_1, \ldots, t_n$ sind freie, $t_{n+1}, \ldots, t_{n+m}$ sind gebundene Tupelvariablen.
- $A_1, \ldots, A_n$ sind Attribute der Relationen bzgl. derer die t_i instanziiert sind.
- α ist eine logische Formel, wobei die Menge der Atome Σ, aus denen α besteht, wie folgt definiert ist:

1. „$r(t)$“ alternativ: „$t \in r$“
   ist ein Atom, wobei t eine Tupelvariable und r eine Relation bezeichnet. $r(t)$ ist wahr für eine Instanziierung von t, falls diese Instanziierung ein Tupel in r ist.
2. „$t_i.A_i$ *op* $t_j.A_j$“
   ist ein Atom mit *op* $\in \{=, <, \leq, >, \geq, \neq\}$. t_i, t_j bezeichnen Tupelvariablen und A_i, A_j bezeichnen Attribute aus den Relationen hinsichtlich derer t_i bzw. t_j instanziiert sind.
3. „$t_i.A_i$ *op* c“
   ist ein Atom mit *op* $\in \{=, <, \leq, >, \geq, \neq\}$. t_i bezeichnet eine Tupelvariable, A_i ein Attribut aus der Relation hinsichtlich der t_i instanziiert ist und $c \in$ *dom*(A_i) ist eine Konstante.

Relationaler Tupelkalkül

Beispiel 1

Mitarbeiter				
Name	PersNr	Wohnort	ChefPersNr	AbtNr
Smith	1234	Weimar	3334	5
Wong	3334	Köln	8886	5
Zelaya	9998	Erfurt	9876	4

Abteilung		
AbtName	Nr	Manager
Forschung	5	3334
Verwaltung	4	9876
Stab	1	8886

AbtStandort	
AbtNr	Ort
1	Berlin
4	Weimar
5	Hamburg
5	Köln

ArbeitetInProjekt	
PersNr	ProjektNr
1234	1
1234	2
6668	3
4534	1

Projekt			
Name	Nr	Ort	AbtNr
X	1	Köln	5
Y	2	Hamburg	5
Z	3	Weimar	4
New	8	Weimar	4

Anfrage

„Liefere Name und Wohnort der Mitarbeiter, die in der Forschung arbeiten.“

Relationaler Tupelkalkül

Beispiel 1

Mitarbeiter				
Name	PersNr	Wohnort	ChefPersNr	AbtNr
Smith	1234	Weimar	3334	5
Wong	3334	Köln	8886	5
Zelaya	9998	Erfurt	9876	4

Abteilung		
AbtName	Nr	Manager
Forschung	5	3334
Verwaltung	4	9876
Stab	1	8886

AbtStandort	
AbtNr	Ort
1	Berlin
4	Weimar
5	Hamburg
5	Köln

ArbeitetInProjekt	
PersNr	ProjektNr
1234	1
1234	2
6668	3
4534	1

Projekt			
Name	Nr	Ort	AbtNr
X	1	Köln	5
Y	2	Hamburg	5
Z	3	Weimar	4
New	8	Weimar	4

Anfrage

„Liefere Name und Wohnort der Mitarbeiter, die in der Forschung arbeiten.“

Relationenalgebra

$\rightsquigarrow \mathcal{TAFEL}$

Tupelkalkül

$\{(t_1.\text{Name}, t_1.\text{Wohnort}) \mid$
$\text{Mitarbeiter}(t_1) \;\wedge\; \exists t_2(\text{Abteilung}(t_2) \;\wedge\; t_2.\text{AbtName} =' \text{Forschung}' \;\wedge\; t_2.\text{Nr} = t_1.\text{AbtNr})\}$

Bemerkungen:

- Die Qualifizierung der freien Tupelvariablen vor dem Trennsymbol „|“ bei der Mengenbildung entspricht einer Projektion, π, in der relationalen Algebra.
  Beispiel: $\{(t_1.\text{Name}, \ldots) \mid \ldots\}$
- Eine Bedingung, die sich auf ein Attribut und eine Konstante bezieht, entspricht einer Selektion, σ, in der relationalen Algebra.
  Beispiel: $t_2.\text{AbtName} =' \text{Forschung}'$
- Eine Bedingung bzgl. zweier Attribute, die sich auf Tupel aus verschiedenen Relationen bezieht, entspricht einem Verbund (Join), $\bowtie$, in der relationalen Algebra.
  Beispiel: $t_2.\text{Nr} = t_1.\text{AbtNr}$

Relationaler Tupelkalkül

Beispiel 2

Mitarbeiter				
Name	PersNr	Wohnort	ChefPersNr	AbtNr
Smith	1234	Weimar	3334	5
Wong	3334	Köln	8886	5
Zelaya	9998	Erfurt	9876	4

Abteilung		
AbtName	Nr	Manager
Forschung	5	3334
Verwaltung	4	9876
Stab	1	8886

AbtStandort	
AbtNr	Ort
1	Berlin
4	Weimar
5	Hamburg
5	Köln

ArbeitetInProjekt	
PersNr	ProjektNr
1234	1
1234	2
6668	3
4534	1

Projekt			
Name	Nr	Ort	AbtNr
X	1	Köln	5
Y	2	Hamburg	5
Z	3	Weimar	4
New	8	Weimar	4

Anfrage

„Liefere für jedes Projekt in Weimar dessen Nummer, die Nummer der durchführenden Abteilung sowie Name und Wohnort des Abteilungsmanagers.“

Relationaler Tupelkalkül

Beispiel 2

Mitarbeiter				
Name	PersNr	Wohnort	ChefPersNr	AbtNr
Smith	1234	Weimar	3334	5
Wong	3334	Köln	8886	5
Zelaya	9998	Erfurt	9876	4

Abteilung		
AbtName	Nr	Manager
Forschung	5	3334
Verwaltung	4	9876
Stab	1	8886

AbtStandort	
AbtNr	Ort
1	Berlin
4	Weimar
5	Hamburg
5	Köln

ArbeitetInProjekt	
PersNr	ProjektNr
1234	1
1234	2
6668	3
4534	1

Projekt			
Name	Nr	Ort	AbtNr
X	1	Köln	5
Y	2	Hamburg	5
Z	3	Weimar	4
New	8	Weimar	4

Anfrage

„Liefere für jedes Projekt in Weimar dessen Nummer, die Nummer der durchführenden Abteilung sowie Name und Wohnort des Abteilungsmanagers.“

Relationenalgebra

$$\pi_{\substack{\text{PNr,AbtNr,}\\ \text{Name,Wohnort}}}\left(\left(\rho_{\substack{\text{PNr}\leftarrow\text{Nr,}\\ \text{PName}\leftarrow\text{Name}}}\left(\sigma_{\text{Ort}='\text{Weimar}'}(\text{Projekt})\right)\right) \bowtie_{\text{AbtNr}=\text{Nr}} \text{Abteilung} \bowtie_{\text{Manager}=\text{PersNr}} \text{Mitarbeiter}\right)$$

Tupelkalkül

$\leadsto$ *TAFEL*

Relationaler Tupelkalkül

Beispiel 3

Mitarbeiter				
Name	PersNr	Wohnort	ChefPersNr	AbtNr
Smith	1234	Weimar	3334	5
Wong	3334	Köln	8886	5
Zelaya	9998	Erfurt	9876	4

Abteilung		
AbtName	Nr	Manager
Forschung	5	3334
Verwaltung	4	9876
Stab	1	8886

AbtStandort	
AbtNr	Ort
1	Berlin
4	Weimar
5	Hamburg
5	Köln

ArbeitetInProjekt	
PersNr	ProjektNr
1234	1
1234	2
6668	3
4534	1

Projekt			
Name	Nr	Ort	AbtNr
X	1	Köln	5
Y	2	Hamburg	5
Z	3	Weimar	4
New	8	Weimar	4

Anfrage

„Liefere die Namen der Mitarbeiter, die in *allen* Projekten der Abteilung 5 arbeiten.“

Relationaler Tupelkalkül

Beispiel 3

Mitarbeiter				
Name	PersNr	Wohnort	ChefPersNr	AbtNr
Smith	1234	Weimar	3334	5
Wong	3334	Köln	8886	5
Zelaya	9998	Erfurt	9876	4

Abteilung		
AbtName	Nr	Manager
Forschung	5	3334
Verwaltung	4	9876
Stab	1	8886

AbtStandort	
AbtNr	Ort
1	Berlin
4	Weimar
5	Hamburg
5	Köln

ArbeitetInProjekt	
PersNr	ProjektNr
1234	1
1234	2
6668	3
4534	1

Projekt			
Name	Nr	Ort	AbtNr
X	1	Köln	5
Y	2	Hamburg	5
Z	3	Weimar	4
New	8	Weimar	4

Anfrage

„Liefere die Namen der Mitarbeiter, die in *allen* Projekten der Abteilung 5 arbeiten.“

Relationenalgebra

$\pi_{\text{Name}}\ ((\text{ArbeitetInProjekt} \div \rho_{\text{ProjektNr} \leftarrow \text{Nr}}(\pi_{\text{Nr}}(\sigma_{\text{AbtNr}='5'}(\text{Projekt})))) \bowtie \text{Mitarbeiter})$

Tupelkalkül

$\rightsquigarrow$ $\mathcal{TAFEL}$

Bemerkungen:

- Zur Semantik des $\forall$-Quantors in der Formel $\alpha = \text{Mitarbeiter}(t_1) \wedge \forall t_3 \beta$: „α ist erfüllt für diejenigen Mitarbeiter(tupel t_1), bei denen für *alle* Tupel t_3 die Teilformel β erfüllt ist.“ Beachte, dass t_3 an *alle* Tupel des Universums bzw. des Datenbankzustandes $d(\boldsymbol{\mathcal{R}})$ gebunden wird und bzgl. *aller* möglichen Instanziierungen die Formel β erfüllen muss.

- Die Quantoren können verschoben werden, solange sich keine Variablenbindungen ändern und die Ordnung zwischen den $\exists$- und $\forall$-Quantoren erhalten bleibt:

$$\begin{aligned}
&\{(t_1.\text{Name}) \mid \text{Mitarbeiter}(t_1) \wedge \\
&\quad \forall t_3 \exists t_4 (\neg(\text{Projekt}(t_3) \ \wedge \ (t_3.\text{AbtNr} = 5)) \vee \\
&\qquad (\text{ArbeitetInProjekt}(t_4) \ \wedge \ t_4.\text{ProjektNr} = t_3.\text{Nr} \ \wedge \ t_4.\text{PersNr} = t_1.\text{PersNr}))\}
\end{aligned}$$

Relationaler Tupelkalkül

Beispiel 4

Buecher	
Titel	**Verlag**
Harry Potter	Princeton
Heuristics	Addison
Glücksformel	dpunkt
Datenbanken	Springer

Buchhaendler		
Name	**Stadt**	**PLZ**
Lehmann	Berlin	99011
Meiersche	Aachen	42100
Amazon	Köln	52100

Angebote	
Titel	**Haendler**
Harry Potter	Lehmann
Harry Potter	Meiersche
Harry Potter	Amazon
Datenbanken	Amazon
Glücksformel	Amazon
Glücksformel	Lehmann

Anfrage

„Welche Titel sind bei *allen* Buchhändlern im Angebot?“

Relationaler Tupelkalkül

Beispiel 4

Buecher	
Titel	**Verlag**
Harry Potter	Princeton
Heuristics	Addison
Glücksformel	dpunkt
Datenbanken	Springer

Buchhaendler		
Name	**Stadt**	**PLZ**
Lehmann	Berlin	99011
Meiersche	Aachen	42100
Amazon	Köln	52100

Angebote	
Titel	**Haendler**
Harry Potter	Lehmann
Harry Potter	Meiersche
Harry Potter	Amazon
Datenbanken	Amazon
Glücksformel	Amazon
Glücksformel	Lehmann

Anfrage

„Welche Titel sind bei *allen* Buchhändlern im Angebot?“

Relationenalgebra

$\rho_{\text{Name}\leftarrow\text{Haendler}}(\text{Angebote}) \div \pi_{\text{Name}}(\text{Buchhaendler})$

Relationaler Tupelkalkül

Beispiel 4

Buecher	
Titel	**Verlag**
Harry Potter	Princeton
Heuristics	Addison
Glücksformel	dpunkt
Datenbanken	Springer

Buchhaendler		
Name	**Stadt**	**PLZ**
Lehmann	Berlin	99011
Meiersche	Aachen	42100
Amazon	Köln	52100

Angebote	
Titel	**Haendler**
Harry Potter	Lehmann
Harry Potter	Meiersche
Harry Potter	Amazon
Datenbanken	Amazon
Glücksformel	Amazon
Glücksformel	Lehmann

Anfrage

„Welche Titel sind bei *allen* Buchhändlern im Angebot?“

Relationenalgebra

$\rho_{\text{Name} \leftarrow \text{Haendler}}(\text{Angebote}) \div \pi_{\text{Name}}(\text{Buchhaendler})$

Tupelkalkül

$\{(t_1.\text{Titel}) \mid \text{Buecher}(t_1) \wedge$

$\quad \forall t_3(\neg \text{Angebote}(t_3) \vee$ oder: $\forall t_3(\text{Angebote}(t_3) \rightarrow$

$\quad\quad \exists t_2(\text{Buchhaendler}(t_2) \wedge t_3.\text{Haendler} = t_2.\text{Name} \wedge t_3.\text{Titel} = t_1.\text{Titel}))\}$

Relationaler Tupelkalkül

Beispiel 4

Buecher	
Titel	**Verlag**
Harry Potter	Princeton
Heuristics	Addison
Glücksformel	dpunkt
Datenbanken	Springer

Buchhaendler		
Name	**Stadt**	**PLZ**
Lehmann	Berlin	99011
Meiersche	Aachen	42100
Amazon	Köln	52100

Angebote	
Titel	**Haendler**
Harry Potter	Lehmann
Harry Potter	Meiersche
Harry Potter	Amazon
Datenbanken	Amazon
Glücksformel	Amazon
Glücksformel	Lehmann

Anfrage

„Welche Titel sind bei *allen* Buchhändlern im Angebot?“

Relationenalgebra

$\rho_{\text{Name}\leftarrow\text{Haendler}}(\text{Angebote}) \div \pi_{\text{Name}}(\text{Buchhaendler})$

Tupelkalkül

$$\{(t_1.\text{Titel}) \mid \text{Buecher}(t_1) \wedge$$
$$\forall t_3(\neg\text{Angebote}(t_3) \vee$$
$$\exists t_2(\text{Buchhaendler}(t_2) \;\wedge\; t_3.\text{Haendler} = t_2.\text{Name} \;\wedge\; t_3.\text{Titel} = t_1.\text{Titel}))\}$$

Bemerkungen:

- Bei (formalen, logischen, natürlichen) Sprachen unterscheidet man zwischen Sätzen aus der Sprache selbst und der Formulierung von Zusammenhängen *über* solche Sätze. Sätze aus der Sprache selbst dienen uns zur Kommunikation mittels dieser Sprache; die Symbole, die vewendet werden, um solche Sätze zu formulieren, gehören zur Objektsprache. Symbole, die verwendet werden, um *über* Sätze zu sprechen, die in der Objektsprache formuliert sind, gehören zur Metasprache.

- Die Formelbezeichner α, β, γ, die Prädikatsbezeichner P, Q, die Quantoren $\forall$, $\exists$, die Variablenbezeichner t, x, y, z, und die Junktoren $\neg$, $\wedge$, $\vee$, $\rightarrow$ gehören zur Objektsprache. Das $\approx$-Zeichen ist ein Zeichen der Metasprache und steht für „ist logisch äquivalent mit“. Es gelten u.a. folgende Zusammenhänge:

$$
\begin{aligned}
\forall x P(x) &\approx \nexists x(\neg P(x)) \\
\alpha \rightarrow \beta &\approx \neg\alpha \vee \beta \\
\neg(\alpha \wedge \beta) &\approx \neg\alpha \vee \neg\beta \\
(\alpha \wedge \beta) \rightarrow \gamma &\approx \neg\alpha \vee \neg\beta \vee \gamma
\end{aligned}
$$

Relationaler Tupelkalkül

Sichere Anfragen im Tupelkalkül [Sichere Anfragen]

Definition 2 (Domäne einer Formel α)

Der Bereich bzw. die Domäne einer Formel α ist die Menge aller Konstanten in α vereinigt mit der Menge aller Attributwerte der Relationen r, $r \in \alpha$.

Beispiel:

Mitarbeiter				
Name	PersNr	Wohnort	ChefPersNr	AbtNr
Smith	1234	Weimar	3334	5
Wong	3334	Köln	8886	5
Zelaya	9998	Erfurt	9876	4

Domäne der Formel „$\neg$ Mitarbeiter(t)“ :

{Smith, Wong, Zelaya, Weimar, Köln, Erfurt, 4, 5, 1234, 3334, 8886, 9876, 9998}

Relationaler Tupelkalkül

Sichere Anfragen im Tupelkalkül [Sichere Anfragen]

Folgende Anfrage liefert eine unendliche Zahl von Ergebnissen:

$$\{t \mid \neg \text{Mitarbeiter}(t)\}$$

Eine Anfrage $\{t \mid \alpha\}$ des Tupelkalküls ist sicher, falls folgende Bedingungen erfüllt sind:

1. Ist $t = (c_1, c_2, \ldots, c_n)$ im Anfrageergebnis enthalten, so muss $\{c_1, c_2, \ldots, c_n\}$ Teilmenge der Domäne von α sein.

2. Für jede Teilformel $\exists t \beta$ muss gelten, dass β höchstens für Elemente aus seiner Domäne erfüllbar sein kann.

   $\Rightarrow$ β ist für alle Elemente, die nicht in seiner Domäne sind, unerfüllbar.

3. Für jede Teilformel $\forall t \beta$ muss gelten, dass $\forall t \beta$ dann und nur dann erfüllt ist, wenn β für alle Elemente aus seiner Domäne erfüllt ist.

   $\Rightarrow$ β ist für alle Elemente, die nicht in seiner Domäne sind, immer erfüllt.

Bemerkungen:

- Semantische Sicherheit ist eine Eigenschaft, die im Einzelfall leicht zu zeigen sein kann, die aber in der Allgemeinheit nicht automatisch nachprüfbar ist. Die Ursache dafür liegt in der Unentscheidbarkeit der Prädikatenlogik erster Stufe mit Arithmetik.
- Mit Hilfe der Domäne wird eine syntaktische Einschränkung eingeführt, unter der sich semantische Sicherheit einfach überprüfen lässt.
- Die Bedingungen 2. und 3. verhindern, dass unendlich viele Variableninstanziierungen evaluiert werden müssen. Beachte: es sind auch Anfragen denkbar, die ein endliches Ergebnis liefern, aber die Evaluierung unendlich vieler Variableninstanziierungen erfordern. Bei solchen Anfragen liegt die „Unendlichkeit“ in der Zeit und nicht in der Größe der Ergebnismenge.

Relationaler Domänenkalkül

Konzepte [Tupelkalkül]

1. Domänenvariablen, die sich auf die Attribute, A, in den Relationen beziehen und mit jedem Wert aus dem Wertebereich *dom*(A) von A instanziiert werden können.

2. Formeln, mit denen sich auf Basis der Domänenvariablen Zusammenhänge zwischen Attributen formulieren lassen.

Bemerkungen:

- SQL *(Structured Query Language)* basiert auf dem relationalen Tupelkalkül und wurde von IBM-Research, San Jose, Kalifornien, entwickelt.
- QBE *(Query By Example)* basiert auf dem relationalen Domänenkalkül und wurde von IBM-Research, Yorktown Heights, New York, entwickelt. Diese Entwicklung fand fast zeitgleich mit der Entwicklung von SQL in San Jose statt.
- QBE war eine der ersten graphischen Anfragesprachen für Datenbanksysteme und ist bei IBM als Interface-Option für DB2 erhältlich.

Relationaler Domänenkalkül

Anfragen [Tupelkalkül]

Anfrage im relationalen Domänenkalkül mit Variablen $x_1, \ldots, x_n,\ x_{n+1}, \ldots, x_{n+m}$:

$$\{(x_1, x_2, \ldots, x_n) \mid \alpha\}$$

- $x_1, \ldots, x_n$ sind freie, $x_{n+1}, \ldots, x_{n+m}$ sind gebundene Domänenvariablen.
- α ist eine logische Formel, wobei die Menge der Atome Σ, aus denen α besteht, wie folgt definiert ist:

Relationaler Domänenkalkül

Anfragen [Tupelkalkül]

Anfrage im relationalen Domänenkalkül mit Variablen $x_1, \ldots, x_n,\ x_{n+1}, \ldots, x_{n+m}$:

$$\{(x_1, x_2, \ldots, x_n) \mid \alpha\}$$

- $x_1, \ldots, x_n$ sind freie, $x_{n+1}, \ldots, x_{n+m}$ sind gebundene Domänenvariablen.
- α ist eine logische Formel, wobei die Menge der Atome Σ, aus denen α besteht, wie folgt definiert ist:

1. „$r(x_{r_1}, x_{r_2}, \ldots, x_{r_k})$“ alternativ: „$(x_{r_1}, x_{r_2}, \ldots, x_{r_k}) \in r$“
   ist ein Atom, wobei die x_{r_i} Domänenvariablen und r eine Relation über k Attribute bezeichnet. $r(x_{r_1}, x_{r_2}, \ldots, x_{r_k})$ ist wahr für eine Instanziierung von $(x_{r_1}, x_{r_2}, \ldots, x_{r_k})$, falls diese Instanziierung ein Tupel in r ist.

Relationaler Domänenkalkül

Anfragen [Tupelkalkül]

Anfrage im relationalen Domänenkalkül mit Variablen $x_1, \ldots, x_n,\ x_{n+1}, \ldots, x_{n+m}$:

$$\{(x_1, x_2, \ldots, x_n) \mid \alpha\}$$

- $x_1, \ldots, x_n$ sind freie, $x_{n+1}, \ldots, x_{n+m}$ sind gebundene Domänenvariablen.
- α ist eine logische Formel, wobei die Menge der Atome Σ, aus denen α besteht, wie folgt definiert ist:

1. „$r(x_{r_1}, x_{r_2}, \ldots, x_{r_k})$“ alternativ: „$(x_{r_1}, x_{r_2}, \ldots, x_{r_k}) \in r$“
   ist ein Atom, wobei die x_{r_i} Domänenvariablen und r eine Relation über k Attribute bezeichnet. $r(x_{r_1}, x_{r_2}, \ldots, x_{r_k})$ ist wahr für eine Instanziierung von $(x_{r_1}, x_{r_2}, \ldots, x_{r_k})$, falls diese Instanziierung ein Tupel in r ist.

2. „x_i *op* x_j“
   ist ein Atom mit *op* $\in \{=, <, \leq, >, \geq, \neq\}$. x_i, x_j bezeichnen Domänenvariablen, die über den Wertebereichen der zugeordneten Attribute instanziiert sind.

3. „x_i *op* c“
   ist ein Atom mit *op* $\in \{=, <, \leq, >, \geq, \neq\}$. x_i bezeichnet eine Domänenvariable, die über dem Wertebereich des zugeordneten Attributes instanziiert ist, und $c \in$ *dom*(A_i) ist eine Konstante aus dem gleichen Wertebereich.

Relationaler Domänenkalkül

Beispiel 1

Mitarbeiter				
Name	PersNr	Wohnort	ChefPersNr	AbtNr
Smith	1234	Weimar	3334	5
Wong	3334	Köln	8886	5
Zelaya	9998	Erfurt	9876	4

Abteilung		
AbtName	Nr	Manager
Forschung	5	3334
Verwaltung	4	9876
Stab	1	8886

AbtStandort	
AbtNr	Ort
1	Berlin
4	Weimar
5	Hamburg
5	Köln

ArbeitetInProjekt	
PersNr	ProjektNr
1234	1
1234	2
6668	3
4534	1

Projekt			
Name	Nr	Ort	AbtNr
X	1	Köln	5
Y	2	Hamburg	5
Z	3	Weimar	4
New	8	Weimar	4

Anfrage

„Liefere Name und Wohnort der Mitarbeiter, die in der Forschung arbeiten.“

Relationaler Domänenkalkül

Beispiel 1

Mitarbeiter				
Name	PersNr	Wohnort	ChefPersNr	AbtNr
Smith	1234	Weimar	3334	5
Wong	3334	Köln	8886	5
Zelaya	9998	Erfurt	9876	4

Abteilung		
AbtName	Nr	Manager
Forschung	5	3334
Verwaltung	4	9876
Stab	1	8886

AbtStandort	
AbtNr	Ort
1	Berlin
4	Weimar
5	Hamburg
5	Köln

ArbeitetInProjekt	
PersNr	ProjektNr
1234	1
1234	2
6668	3
4534	1

Projekt			
Name	Nr	Ort	AbtNr
X	1	Köln	5
Y	2	Hamburg	5
Z	3	Weimar	4
New	8	Weimar	4

Anfrage

„Liefere Name und Wohnort der Mitarbeiter, die in der Forschung arbeiten.“

Relationenalgebra

$\rightsquigarrow$ $\mathcal{TAFEL}$

Domänenkalkül

$$\{(x_1, x_3) \mid \exists x_2 \exists x_4 \exists x_5\ \exists y_1 \exists y_2 \exists y_3$$
$$(\text{Mitarbeiter}(x_1, x_2, x_3, x_4, x_5)\ \wedge\ \text{Abteilung}(y_1, y_2, y_3)\ \wedge\ y_1 = \text{'Forschung'}\ \wedge\ y_2 = x_5)\}$$

Relationaler Domänenkalkül

Beispiel 1

Mitarbeiter				
Name	PersNr	Wohnort	ChefPersNr	AbtNr
Smith	1234	Weimar	3334	5
Wong	3334	Köln	8886	5
Zelaya	9998	Erfurt	9876	4

Abteilung		
AbtName	Nr	Manager
Forschung	5	3334
Verwaltung	4	9876
Stab	1	8886

AbtStandort	
AbtNr	Ort
1	Berlin
4	Weimar
5	Hamburg
5	Köln

ArbeitetInProjekt	
PersNr	ProjektNr
1234	1
1234	2
6668	3
4534	1

Projekt			
Name	Nr	Ort	AbtNr
X	1	Köln	5
Y	2	Hamburg	5
Z	3	Weimar	4
New	8	Weimar	4

Anfrage

„Liefere Name und Wohnort der Mitarbeiter, die in der Forschung arbeiten.“

Relationenalgebra

$\leadsto$ *TAFEL*

Domänenkalkül   Konvention: gebundene Variablen sind per Default $\exists$-quantifiziert.

$$\{(x_1, x_3) \mid \exists x_5 \exists y_1 \exists y_2$$
$$(\text{Mitarbeiter}(x_1, x_2, x_3, x_4, x_5) \ \wedge \ \text{Abteilung}(y_1, y_2, y_3) \ \wedge \ y_1 = \text{'Forschung'} \ \wedge \ y_2 = x_5)\}$$

Relationaler Domänenkalkül

Beispiel 1

Mitarbeiter				
Name	PersNr	Wohnort	ChefPersNr	AbtNr
Smith	1234	Weimar	3334	5
Wong	3334	Köln	8886	5
Zelaya	9998	Erfurt	9876	4

Abteilung		
AbtName	Nr	Manager
Forschung	5	3334
Verwaltung	4	9876
Stab	1	8886

AbtStandort	
AbtNr	Ort
1	Berlin
4	Weimar
5	Hamburg
5	Köln

ArbeitetInProjekt	
PersNr	ProjektNr
1234	1
1234	2
6668	3
4534	1

Projekt			
Name	Nr	Ort	AbtNr
X	1	Köln	5
Y	2	Hamburg	5
Z	3	Weimar	4
New	8	Weimar	4

Anfrage

„Liefere Name und Wohnort der Mitarbeiter, die in der Forschung arbeiten.“

Relationenalgebra

$\rightsquigarrow$ *TAFEL*

Domänenkalkül Abkürzung: Konstanten als Parameter.

$$\{(x_1, x_3) \mid \exists x_5 \exists y_2$$
$$(\text{Mitarbeiter}(x_1, x_2, x_3, x_4, x_5) \;\wedge\; \text{Abteilung}('\text{Forschung}', y_2, y_3) \;\wedge\; y_2 = x_5)\}$$

Relationaler Domänenkalkül

Beispiel 1

Mitarbeiter				
Name	PersNr	Wohnort	ChefPersNr	AbtNr
Smith	1234	Weimar	3334	5
Wong	3334	Köln	8886	5
Zelaya	9998	Erfurt	9876	4

Abteilung		
AbtName	Nr	Manager
Forschung	5	3334
Verwaltung	4	9876
Stab	1	8886

AbtStandort	
AbtNr	Ort
1	Berlin
4	Weimar
5	Hamburg
5	Köln

ArbeitetInProjekt	
PersNr	ProjektNr
1234	1
1234	2
6668	3
4534	1

Projekt			
Name	Nr	Ort	AbtNr
X	1	Köln	5
Y	2	Hamburg	5
Z	3	Weimar	4
New	8	Weimar	4

Anfrage

„Liefere Name und Wohnort der Mitarbeiter, die in der Forschung arbeiten.“

Relationenalgebra

$\rightsquigarrow$ *TAFEL*

Domänenkalkül Abkürzung: Unifikation von Domänenvariablen

$\{(x_1, x_3) \mid \exists x_5$

$(\text{Mitarbeiter}(x_1, x_2, x_3, x_4, x_5) \ \wedge \ \text{Abteilung}('\text{Forschung}', x_5, y_3))\}$

Bemerkungen:

- Eine Bedingung, die sich auf eine Domänenvariable und eine Konstante bezieht, entspricht einer Selektion, σ, in der relationalen Algebra.
  Beispiel: $y_1 = {}'\text{Forschung}'$

- Eine Bedingung bzgl. zweier Domänenvariablen, die sich auf zwei verschiedene Relationen beziehen, entspricht einem Verbund (Join), $\bowtie$, in der relationalen Algebra.
  Beispiel: $x_5 = y_2$

Relationaler Domänenkalkül

Beispiel 2

Mitarbeiter				
Name	PersNr	Wohnort	ChefPersNr	AbtNr
Smith	1234	Weimar	3334	5
Wong	3334	Köln	8886	5
Zelaya	9998	Erfurt	9876	4

Abteilung		
AbtName	Nr	Manager
Forschung	5	3334
Verwaltung	4	9876
Stab	1	8886

AbtStandort	
AbtNr	Ort
1	Berlin
4	Weimar
5	Hamburg
5	Köln

ArbeitetInProjekt	
PersNr	ProjektNr
1234	1
1234	2
6668	3
4534	1

Projekt			
Name	Nr	Ort	AbtNr
X	1	Köln	5
Y	2	Hamburg	5
Z	3	Weimar	4
New	8	Weimar	4

Anfrage

„Liefere für jedes Projekt in Weimar dessen Nummer, die Nummer der durchführenden Abteilung sowie Name und Wohnort des Abteilungsmanagers.“

Relationaler Domänenkalkül

Beispiel 2

Mitarbeiter				
Name	PersNr	Wohnort	ChefPersNr	AbtNr
Smith	1234	Weimar	3334	5
Wong	3334	Köln	8886	5
Zelaya	9998	Erfurt	9876	4

Abteilung		
AbtName	Nr	Manager
Forschung	5	3334
Verwaltung	4	9876
Stab	1	8886

AbtStandort	
AbtNr	Ort
1	Berlin
4	Weimar
5	Hamburg
5	Köln

ArbeitetInProjekt	
PersNr	ProjektNr
1234	1
1234	2
6668	3
4534	1

Projekt			
Name	Nr	Ort	AbtNr
X	1	Köln	5
Y	2	Hamburg	5
Z	3	Weimar	4
New	8	Weimar	4

Anfrage

„Liefere für jedes Projekt in Weimar dessen Nummer, die Nummer der durchführenden Abteilung sowie Name und Wohnort des Abteilungsmanagers.“

Relationenalgebra

$$\pi_{\substack{\text{PNr,AbtNr,}\\ \text{Name,Wohnort}}} \left(\left(\rho_{\substack{\text{PNr}\leftarrow\text{Nr,}\\ \text{PName}\leftarrow\text{Name}}} \left(\sigma_{\text{Ort}='\text{Weimar}'}(\text{Projekt}) \right) \right) \bowtie_{\text{AbtNr}=\text{Nr}} \text{Abteilung} \bowtie_{\text{Manager}=\text{PersNr}} \text{Mitarbeiter} \right)$$

Domänenkalkül

$\leadsto$ *TAFEL*

Relationaler Domänenkalkül

Beispiel 3

Mitarbeiter				
Name	PersNr	Wohnort	ChefPersNr	AbtNr
Smith	1234	Weimar	3334	5
Wong	3334	Köln	8886	5
Zelaya	9998	Erfurt	9876	4

Abteilung		
AbtName	Nr	Manager
Forschung	5	3334
Verwaltung	4	9876
Stab	1	8886

AbtStandort	
AbtNr	Ort
1	Berlin
4	Weimar
5	Hamburg
5	Köln

ArbeitetInProjekt	
PersNr	ProjektNr
1234	1
1234	2
6668	3
4534	1

Projekt			
Name	Nr	Ort	AbtNr
X	1	Köln	5
Y	2	Hamburg	5
Z	3	Weimar	4
New	8	Weimar	4

Anfrage

„Liefere die Namen der Mitarbeiter, die in *allen* Projekten der Abteilung 5 arbeiten.“

Relationaler Domänenkalkül

Beispiel 3

Mitarbeiter				
Name	PersNr	Wohnort	ChefPersNr	AbtNr
Smith	1234	Weimar	3334	5
Wong	3334	Köln	8886	5
Zelaya	9998	Erfurt	9876	4

Abteilung		
AbtName	Nr	Manager
Forschung	5	3334
Verwaltung	4	9876
Stab	1	8886

AbtStandort	
AbtNr	Ort
1	Berlin
4	Weimar
5	Hamburg
5	Köln

ArbeitetInProjekt	
PersNr	ProjektNr
1234	1
1234	2
6668	3
4534	1

Projekt			
Name	Nr	Ort	AbtNr
X	1	Köln	5
Y	2	Hamburg	5
Z	3	Weimar	4
New	8	Weimar	4

Anfrage

„Liefere die Namen der Mitarbeiter, die in *allen* Projekten der Abteilung 5 arbeiten.“

Relationenalgebra

$\pi_{\text{Name}}\ ((\text{ArbeitetInProjekt} \div \rho_{\text{ProjektNr}\leftarrow\text{Nr}}(\pi_{\text{Nr}}(\sigma_{\text{AbtNr}='5'}(\text{Projekt})))) \bowtie \text{Mitarbeiter})$

Domänenkalkül

$\rightsquigarrow$ $\mathcal{TAFEL}$

Bemerkungen:

- Die Quantoren können verschoben werden, solange sich keine Variablenbindungen ändern und die Ordnung zwischen den $\exists$- und $\forall$-Quantoren erhalten bleibt:

  $$\{(x_1) \mid \exists x_2 \exists x_3 \exists x_4 \exists x_5\ \forall z_1 \forall z_2 \forall z_3 \forall z_4\ \exists y_1 \exists y_2$$
  $$(\textsf{Mitarbeiter}(x_1, x_2, x_3, x_4, x_5)\ \wedge\ (\neg \textsf{Projekt}(z_1, z_2, z_3, z_4)\ \vee\ \neg(z_4 = 5) \vee$$
  $$(\textsf{ArbeitetInProjekt}(y_1, y_2)\ \wedge\ y_2 = z_2\ \wedge\ y_1 = x_2)))\}$$

- Alternative Formeln, die denselben Sachverhalt „Für alle Projekte der Abteilung 5 gilt ...“ modellieren:

  1. $\forall z_1 \forall z_2 \forall z_3 \forall z_4\ ((\textsf{Projekt}(z_1, z_2, z_3, z_4)\ \wedge\ (z_4 = 5))\ \rightarrow\ \ldots$
  2. $\forall z_1 \forall z_2 \forall z_3 \forall z_4\ (\neg(\textsf{Projekt}(z_1, z_2, z_3, z_4)\ \wedge\ z_4 = 5)\ \vee\ \ldots$
  3. $\forall z_1 \forall z_2 \forall z_3 \forall z_4\ (\neg \textsf{Projekt}(z_1, z_2, z_3, z_4)\ \vee\ \neg(z_4 = 5)\ \vee\ \ldots$
  4. $\forall z_1 \forall z_2 \forall z_3\ (\neg \textsf{Projekt}(z_1, z_2, z_3, 5)\ \vee\ \ldots$
  5. $\forall z_1 \forall z_2 \forall z_3\ (\textsf{Projekt}(z_1, z_2, z_3, 5)\ \rightarrow\ \ldots$

Relationaler Domänenkalkül

Beispiel 4

Buecher	
Titel	Verlag
Harry Potter	Princeton
Heuristics	Addison
Glücksformel	dpunkt
Datenbanken	Springer

Buchhaendler		
Name	Stadt	PLZ
Lehmann	Berlin	99011
Meiersche	Aachen	42100
Amazon	Köln	52100

Angebote	
Titel	Haendler
Harry Potter	Lehmann
Harry Potter	Meiersche
Harry Potter	Amazon
Datenbanken	Amazon
Glücksformel	Amazon
Glücksformel	Lehmann

Anfrage

„Welche Titel sind bei *allen* Buchhändlern im Angebot?“

Relationenalgebra

$\rho_{\text{Name}\leftarrow\text{Haendler}}(\text{Angebote}) \div \pi_{\text{Name}}(\text{Buchhaendler})$

Relationaler Domänenkalkül

Beispiel 4

Buecher	
Titel	Verlag
Harry Potter	Princeton
Heuristics	Addison
Glücksformel	dpunkt
Datenbanken	Springer

Buchhaendler		
Name	Stadt	PLZ
Lehmann	Berlin	99011
Meiersche	Aachen	42100
Amazon	Köln	52100

Angebote	
Titel	Haendler
Harry Potter	Lehmann
Harry Potter	Meiersche
Harry Potter	Amazon
Datenbanken	Amazon
Glücksformel	Amazon
Glücksformel	Lehmann

Anfrage

„Welche Titel sind bei *allen* Buchhändlern im Angebot?“

Relationenalgebra

$\rho_{\text{Name}\leftarrow\text{Haendler}}(\text{Angebote}) \div \pi_{\text{Name}}(\text{Buchhaendler})$

Domänenkalkül

$\{(x_1) \mid \exists x_2$

$\quad \text{Buecher}(x_1, x_2) \wedge$

$\quad\quad \forall z_1 \forall z_2 (\neg \text{Angebote}(z_1, z_2) \vee$ oder: $\forall z_1 \forall z_2 (\text{Angebote}(z_1, z_2) \rightarrow$

$\quad\quad\quad \exists y_1 \exists y_2 \exists y_3 (\text{Buchhaendler}(y_1, y_2, y_3) \wedge z_2 = y_1 \wedge z_1 = x_1))\}$

Relationaler Domänenkalkül

Sichere Anfragen im Domänenkalkül

Folgende Anfrage liefert eine unendliche Zahl von Ergebnissen:

$$\{(x_1, x_2, x_3, x_4, x_5) \mid \neg\text{Mitarbeiter}(x_1, x_2, x_3, x_4, x_5)\}$$

Sei die Domäne einer Formel α wie zuvor definiert. Dann ist eine Anfrage $\{(x_1, x_2, \ldots, x_n) \mid \alpha\}$ des Domänenkalküls sicher, falls folgende Bedingungen erfüllt sind:

1. Ist $(c_1, c_2, \ldots, c_n)$ im Anfrageergebnis enthalten, so muss $\{c_1, c_2, \ldots, c_n\}$ Teilmenge der Domäne von α sein.

2. Für jede Teilformel $\exists x \beta$ muss gelten, dass β höchstens für Elemente aus seiner Domäne erfüllbar sein kann.

   $\Rightarrow$ β ist für alle Elemente, die nicht in seiner Domäne sind, unerfüllbar.

3. Für jede Teilformel $\forall x \beta$ muss gelten, dass $\forall x \beta$ dann und nur dann erfüllt ist, wenn β für alle Elemente aus seiner Domäne erfüllt ist.

   $\Rightarrow$ β ist für alle Elemente, die nicht in seiner Domäne sind, immer erfüllt.

Anfragekalküle

Ausdrucksstärke der Kalküle

Folgende drei Sprachen besitzen die gleiche Ausdruckskraft:

1. die relationale Algebra
2. der relationale Tupelkalkül, eingeschränkt auf sichere Anfragen
3. der relationale Domänenkalkül, eingeschränkt auf sichere Anfragen

Bemerkungen:

- Der Beweis erfolgt induktiv über den Aufbau der Ausdrücke in der jeweiligen Sprache. Unter anderem spezifiziert man äquivalente Ausdrücke des Tupelkalküls zu den Basisoperatoren der relationalen Algebra.
- Weil der (sichere) relationale Tupelkalkül und der (sichere) relationale Domänenkalkül die gleiche Ausdruckskraft wie die relationale Algebra besitzen, sind sie auch relational vollständig.
- Die Aussage, dass der relationale Tupelkalkül und der relationale Domänenkalkül relational vollständig sind, bedarf nicht der Einschränkung auf sichere Anfragen.