Datenbank-Grundlagen

SS 2005

3. Relationale Algebra

Prof. Dr. Stefan Böttcher
Universität Paderborn

mit Material von
Prof. Dr. Gregor Engels

Agenda:

Eigenschaften relationaler Anfragesprachen

- **Ad-Hoc-Formulierung:** Benutzer kann Anfragen formulieren, ohne vollständiges Programm schreiben zu müssen.
- **Deskriptivität:** Benutzer kann formulieren „Was will ich haben?" und nicht „Wie komme ich an das, was ich haben will?".
- **Mengenorientiertheit:** Operationen bearbeiten Mengen von Daten, statt einzelne Elemente (navigierend).
- **Abgeschlossenheit:** Anfrage-Ergebnis ist Relation → wieder in Anfragen als Eingabe verwendbar.
- **Adäquatheit:** Alle Konstrukte des zugrunde liegenden Datenmodells werden unterstützt.
- **Orthogonalität:** Die Operationen sind frei kombinierbar.

Eigenschaften relationaler Anfragesprachen

- **Optimierbarkeit:** Sprache besteht aus wenigen Operationen Optimierungsregeln vorhanden.
- **Effizienz von Anfrageoperationen:** jede Operation (X,∪,-, …) hat Komplexität ≤ $O(n^2)$ (n=Tupelanzahl/beteiligter Relation).
- **Sicherheit:** keine korrekte Anfrage liefert unendliches Ergebnis.
- **Eingeschränktheit:**
  Anfragesprache ist keine komplette Programmiersprache (folgt aus Sicherheit, Optimierbarkeit, Effizienz).
- **Vollständigkeit:** mindestens Standard-Anfragen ausdrückbar (d.h Anfragen der Algebra oder des sicheren Relationenkalküls)

Grund-Operationen der relationalen Algebra

Projektion	Spalten nach Namen auswählen
Selektion	Zeilen nach Inhalten auswählen
Vereinigung	von 2 Tabellen mit gleichen Spaltennamen
Differenz	von 2 Tabellen mit gleichen Spaltennamen
kartesisches Produkt	alle Kombinationen von Tupeln beider Tabellen
Spalten umbenennen	

Projektion am Beispiel

Projektion : Spalten nach Namen auswählen

R

bestNr	datum	teil	kunde
21	21.6.04	13	Reich
22	22.6.04	13	Reich

$P_{(teil,kunde)}R$

teil	kunde
13	Reich
~~13~~	~~Reich~~

doppelte Ergebnistupel werden eliminiert !

Projektion (formal)

- **Syntax:** t(*Attributliste*)
- **Semantik:** Tupel, das aus dem Tupel t erzeugt wird, indem man Spalten von t auf *Attributliste* einschränkt
- **Syntax:** $\pi_{(Attributliste)}(Relation)$ (oder $P_{(Attributliste)}(Relation)$)
- **Semantik:** Menge aller Tupel *t(X)*, die durch Spalten-Einschränkung auf *Attributliste* aus Tupeln *t* aus *relation* erzeugt werden

  $\pi_X(Relation) := \{ t(X) \mid t \in Relation \}$
  wobei $X \subseteq schema(Relation)$
- doppelte Ergebnistupel werden eliminiert !

Selektion am Beispiel

Selektion   Zeilen nach Inhalten auswählen

$\underbrace{S_{(kunde='Reich' \wedge bestNr > 10)}}_{\text{Selektionsbedingung}}$ (Bestellungen)

Bestellungen

bestNr	datum	teil	kunde
22	22.6.04	13	Reich
222	22.6.04	132	Reich

Syntax der Selektionsbedingungen

Vergleichsoperator θ steht für

- = oder ≠,
- bei geordneten Wertebereichen auch für ≤, <, ≥ oder >

elementare Selektionsbedingungen sind

- *Attribut θ Konstante*   *(monadischer Vergleich)*
  Beispiel: *σ [kunde ≥ 'R'] (Bestellungen)*
- *Attribut$_1$ θ Attribut$_2$*   *(dyadischer Vergleich)*
- Beispiel: *σ [teil < bestNr] (Bestellungen)*
- *true*
- *false*

Syntax der Selektionsbedingung

Selektionsbedingungen (allgemein):

- **Jede elementare Selektionsbedingung ist eine Selektionsbedingung**
- **Wenn S1 und S2 Selektionsbedingungen sind, dann auch**
- **S1 ∧ S2**
- **S2 v S2**
- **not S2**

nicht anderes ist eine Selektionsbedingung

Syntax und Semantik der Selektion (formal)

Syntax der Selektion:

σ [F] (Relation)   (oder   S *[F] (Relation)*)

Semantik: Menge aller Tupel t aus *Relation*, die Bedingung *F* erfüllen

σ_F *(Relation) := { t | t ∈ Relation ∧ F(t) }*

F ist Selektionsbedingung

Anfrage-Operationen auf Relationen

Vereinigung, Schnitt und Differenzen erfordern
2 Relationen (hier R und S) mit gleichen Spaltennamen

R

bestNr	datum	teil	kunde
21	21.6.04	13	Reich
22	22.6.04	13	Reich

S

bestNr	datum	teil	kunde
22	22.6.04	13	Reich
23	23.6.04	13	Reich

R ∩ S

bestNr	datum	teil	kunde
22	22.6.04	13	Reich

R ∪ S

bestNr	datum	teil	kunde
21	21.6.04	13	Reich
22	22.6.04	13	Reich
23	23.6.04	13	Reich

R - S

bestNr	datum	teil	kunde
21	21.6.04	13	Reich

S - R

bestNr	datum	teil	kunde
23	23.6.04	13	Reich

Syntax und Semantik von Vereinigung, Schnitt und Differenz

Syntax der Vereinigung: **R ∪ S**

Semantik: Menge aller Tupel t, die in R oder in S vorkommen

Syntax des Durchschnitts: **R ∩ S**

Semantik: Menge aller Tupel t, die in R und in S vorkommen

Syntax der Differenz: **R - S**

Semantik: Menge der Tupel, die in R aber nicht in S vorkommen

∩ ist keine Grundoperation, sondern *ableitbar*,
denn **R ∩ S = R - (R-S)**

Kartesisches Produkt B x K

= Menge aller Kombinationen von Tupeln beider Relationen B und K

Bestellungen

bestNr	datum	teil	kunde
22	22.6.04	13	1
29	29.6.04	26	1
32	29.6.04	13	2

Kunden-Stammdaten

kundenNr	name	plz	ort	str	nr
1	Reich				
2	Meier				

Kartesisches Produkt B x K

= Menge aller Kombinationen von Tupeln beider Relationen B und K

Bestellungen

bestNr	datum	teil	kunde
22	22.6.04	13	1
29	29.6.04	26	1
32	29.6.04	13	2

Kunden-Stammdaten

kundenNr	name	plz	ort	str	nr
1	Reich				
2	Meier				

Bestellungen x Kunden-Stammdaten

bestNr	datum	teil	kunde	kundenNr	name	plz	ort	str	nr
22	22.6.04	13	1	1	Reich				
29	29.6.04	26	1	1	Reich				
32	29.6.04	13	2	1	Reich				
22	22.6.04	13	1	2	Meier				
29	29.6.04	26	1	2	Meier				
32	29.6.04	13	2	2	Meier				

Kartesisches Produkt B x K (formal)

Kombination *b conc k* zweier Tupel b und k
= 1 Tupel, das aus b und k so zusammengesetzt wird,
das *b conc k* die Konkatenation der Attributwerte
von b und von k enthält

Beispiel: (1,2) conc (3,4) = (1,2,3,4)

Kartesisches Produkt
= Menge aller Kombinationen von Tupeln
beider Relationen B und K

B x K = { bk | ∃ b ∈ B ∃ k ∈ K (bk = b conc k) }

Umbenennung von Spalten

Umbenennung:

β [neu ← alt] (Relation)
(ändert Attributnamen von *alt* in *neu*)

Beispiel: *Buch = β [Autor ← Author] (Book)*

Buch

Autor
James
Heuer
Vossen
Ullman
Wirth

Book

Author
James
Heuer
Vossen
Ullman
Wirth

Häufig erst dadurch Vereinigung, Differenz und Schnitt möglich.

ermöglicht zudem:
- Kartesisches Produkt, wo sonst nur Natural Join möglich wäre
- oder Join, wo sonst nur Kartesisches Produkt möglich wäre

Join B |x| K
kunde = kundenNr

= Menge aller Kombinationen von Tupeln beider Relationen B und K,
die eine Join-Bedingung (=Verbund-Bedingung) erfüllen

Bestellungen

bestNr	datum	teil	kunde
22	22.6.04	13	1
29	29.6.04	26	1
32	29.6.04	13	2

Kunden-Stammdaten

kundenNr	name	plz	ort	str	nr
1	Reich				
2	Meier				

Bestellungen |x| Kunden-Stammdaten
kunde = kundenNr

bestNr	datum	teil	kunde	kundenNr	name	plz	ort	str	nr
22	22.6.04	13	1	1	Reich				
29	29.6.04	26	1	1	Reich				
32	29.6.04	13	2	2	Meier				

Allgemeiner Join B $|x|_\theta$ K (formal)

Sei θ ein Vergleichsoperator (wie bei der Selektion)

B $|x|_\theta$ K = Menge aller Kombinationen bk von Tupeln b aus Tabelle B und k aus Tabelle K, die die Join-Bedingung θ erfüllen

B $|x|_\theta$ K = { bk | ∃ b ∈ B ∃ k ∈ K
(bk = b conc k ∧ b θ k) }

Der Join (=Verbund) ist aus
Kartesischem Produkt und Selektion ableitbar

B $|x|\theta$ K = $S_{(\theta)}$ **(B x K)**

Semi-Join B |x[$_\theta$K (formal)

Sei θ ein Vergleichsoperator (wie bei Join und Selektion)

B |x[$_\theta$K = alle Tupel aus B, zu denen es ein Tupel aus K gibt, so dass die Join-Bedingung von **B |x|$_\theta$ K** gilt

B |x[$_\theta$K $= \{ b \mid b \in B \wedge \exists k (k \in K \wedge b\, \theta\, k) \}$

Sofern die Attribute von B und K verschieden sind,
sagt man auch:
Der Semi-Join **B |x[$_\theta$ K** ist
die Projektion des Joins **B |x|$_\theta$ K** auf die Attribute von B

Natural Join (natürlicher Verbund)

Wenn A1, …, An die Attribute sind,
die in schema(R) und schema(S) gemeinsam auftreten,
bilde Menge aller Paare von Tupeln r aus R und Tupeln s aus S,
die in A1,…,An übereinstimmen,
entferne doppelte Spalten Ai anschließend durch Projektion

{A1,…,An} := schema(R) $\cap$ schema(S)

R |X| S = { rs | $\exists$ r$\in$R $\exists$ s$\in$S (
r({A1,…,An}) = s ({A1,…,An}) $\wedge$
rs = r conc s(schema(S)-{A1,…,An})
}

Relationale Algebra (formal)

Allgemein:
Algebra $\boldsymbol{A} = (A, \Omega)$ gegeben durch

- Wertebereich *A* und
- darauf definierte Operationen *op:* $A^n \rightarrow A \in \Omega$.

Relationale Algebra:

- A ist Menge aller Relationen Ri über Relationenschemata schema(Ri) zu einem festen Universum *U*. Formal: *A = { Ri | schema(Ri)* $\subseteq$ *U }* , d.h. Ergebnis ist immer Relation.
- Ω ist Menge von Operationen auf Relationen wie Projektion π [_] : A → A, Selektion σ [_] : A → A, kartesisches Produkt A x A → A , Vereinigung A x A → A , Differenz A x A → A , etc.

Relationale Algebra (formale Syntax)

Spalten ausblenden: Projektion π [_] (_)
in [_]: welche Spalten behalten;
in (_): auf welche Relation anwenden

Zeilen heraussuchen: Selektion σ [_] (_)
in [_]: unter welchen Bedingungen Zeilen behalten
in (_): auf welche Relation anwenden

Tabellen verknüpfen: kartesisches Produkt _ **X** _
Alle Kombinationen von Tupeln beider Relationen bilden

Tabellen vereinigen: Vereinigung _ ∪ _
Tupel beider Relationen sammeln, doppelte entfernen

Mengen-Differenz bilden: Differenz _ - _
Tupel der linken Relation sammeln, ohne die der rechten

Division (Beispiel)

R_1

PILOT	FLUGZ.
Snoopy	707
Snoopy	727
Snoopy	747
Meyer	707
Meyer	727
Müller	707
Müller	727
Müller	747
Müller	777
Lüdenscheid	727

R_2

FLUGZ.
707
727
747

R_3

FLUGZ.
707

Piloten, die **alle** Flugzeuge aus R_2 fliegen können

$R_1 \div R_2$ ergibt $R_{1 \div 2}$

PILOT
Snoopy
Müller

Piloten, die **alle** Flugzeuge aus r_3 fliegen können

$R_1 \div R_3$ ergibt $R_{1 \div 3}$

PILOT
Snoopy
Meyer
Müller

Ziel:
All-Quantor in relationaler Algebra ausdrücken, obwohl in Selektionsbedingungen nicht erlaubt

Division – Ersatz für All-Quantoren

Division: (kann aus Ω hergeleitet werden)

Seien schema(R_1) und schema(R_2)$\subseteq$ schema(R_1) gegeben mit schema(R') = schema(R_1) - schema(R_2).
Dann ist Division von R_1 durch R_2
ausdrückbar in der Relationenalgebra durch:

$$R_1 \div R_2 = \pi_{Schema(R')}(R_1) - \pi_{Schema(R')}\left((\pi_{Schema(R')}(R_1) \,|X|\, R_2) - R_1 \right)$$

Division ist also *ableitbare* Operation

Division im Kalkül (mit All-Quantor $\forall$) ausgedrückt:

$$R_1 \div R_2 := R' = \{ t \mid \forall t_2 \in R_2 \ \exists t_1 \in R_1 : t_1(schema(R')) = t \wedge t_1(schema(R_2)) = t_2 \}$$

Unabhängigkeit und relationale Vollständigkeit

- Relationale Vollständigkeit:
  jede andere Menge **L** von Operationen,
  die *mindestens genauso mächtig** ist wie Ω ,
  heißt relational vollständig → Ω als Standardmenge

  * **Das heißt:** zu jedem Ausdruck über Ω gibt es einen Ausdruck über **L,** der bei gleicher Belegung der freien Variablen die gleiche Ergebnisrelation berechnet.

- Unabhängigkeit: kein Operator kann weggelassen werden ohne *Vollständigkeit* zu verlieren;

- Minimale unabhängige Relationenalgebren:
  $\Omega = \{ \pi, \sigma, \mathbf{X}, \beta, \cup, - \}$
  $\Omega = \{ \pi, \sigma, |\mathbf{X}|, \beta, \cup, - \}$

Menge der Tupel mit maximalem Wert

- Bsp: Welche Bestellungen haben die größte Bestellnummer ?
- Problem: Maximum-Operator nachbilden
- Lösung: Alle Bestellungen B ohne die, zu denen es eine Bestellung mit größerer Bestellnummer gibt.

  $B - (B \; |X[_{BestNr<BestNr} \; B)$