Relationale Algebra

Universität Trier

SQL

- Deklarative Anfragesprache
  Der Benutzer sagt dem System, was er will

Relationale Algebra

- Prozedurale Anfragesprache
  Der Benutzer sagt außerdem, wie die Antwort berechnet werden soll
- Tiefere Ebene als SQL
- Bevorzugt in verteilten Systemen
  Weil es eher den Programmen entspricht, die auf dem Netz ausgetauscht werden

Theorem 2.1
(Einfaches) SQL und relationale Algebra haben die gleiche Ausdrucksmächtigkeit.

Grundlagen

- Relationale Algebra besteht aus Operationen, die auf Relationen arbeiten
- Relationen entsprechen Mengen von Tupeln
- Eingabe eines Operators:
  eine oder mehrere Relationen
- Ausgabe eines Operators:
  eine Ergebnisrelation
- Die Ausgabe eines Operators kann als Eingabe eines anderen Operators verwendet werden

Relationale Algebra

Grundlegende relationale Operatoren

- Selektion
- Projektion
- Mengenvereinigung
- Mengendifferenz
- Kartesisches Produkt
- Umbenennung

Zusätzliche relationale Operatoren, die auf Basis dieser grundlegenden Operatoren definiert werden

- Mengenschnitt
- θ-Join
- Natürlicher Join
- Semijoin
- Division

Grundlegende relationale Operatoren

Input

Selektionsprädikat (Formel/Bedingung) und eine Relation R

Notation

Die Selektion aus einer Relation R wird geschrieben als

$$\sigma_F(R)$$

wobei F das Selektionsprädikat ist.

Output

Eine Relation, die aus einer horizontalen Teilmenge der Inputrelation R besteht.
Alle Tupel, die in dieser Teilmenge enthalten sind, erfüllen das Selektionsprädikat.

$$\sigma_F(R) = \{t | t \in R \wedge t \text{ erfüllt } F\}$$

Selektion . . .

Das Selektionsprädikat besteht aus

- Termen:

$$A\ \theta\ c$$

  wobei A ein Attribut von R ist, θ einer der arithmetischen Vergleichsoperatoren $<, >, =, \neq, \leq, \geq$ ist, und c ein Attribut von R oder eine Konstante aus der Domäne von A ist

- Logischen Operatoren:

$$\wedge, \vee, \neg$$

Das Selektionsprädikat enthält keine Quantoren.

Beispiel für die Selektion

EMPLOYEES

EID	EName	Title
E1	Just Vorfan	Programmer
E2	Ann Joy	Elect. Engineer
E3	Lilo Pause	Programmer
E4	Claire Grube	Mech. Engineer
E5	John Doe	Syst. Analyst

Beispielanfrage:
Wähle alle Tupel für Programmierer

Formulierung in relationaler Algebra:
$\sigma_{Title='Programmer'}(\text{EMPLOYEES})$

Ergebnis der Anfrage

EMPLOYEES

EID	EName	Title
E1	Just Vorfan	Programmer
E3	Lilo Pause	Programmer

Eingabe

Eine Relation R und eine Aufzählung β einer Teilmenge der Attribute von R ($\beta \subseteq \{A_1, \ldots, A_n\}$)

Notation

Die Projektion einer Relation R über Attributen β wird geschrieben als

$$\pi_\beta(R)$$

Ausgabe

Eine Relation, die aus einer vertikalen Teilmenge der Inputrelation R besteht. t_β repräsentiert das Tupel t, nachdem die Attribute entfernt wurden, die nicht in β enthalten sind.

$$\pi_\beta(R) = \{t_\beta | t \in R\}$$

Beispiel für Projektion

ASSIGNMENT

ENo	PNo	Duration
E1	P1	5
E2	P4	4
E2	P1	6
E3	P4	3
E4	P1	4
E4	P3	5
E5	P2	7

Beispielanfrage:
Zeige nur die Attribute PNo und Duration der Relation ASSIGNMENT

Formulierung in relationaler Algebra:
$\pi_{PNo,Duration}(\text{ASSIGNMENT})$

Ergebnis der Anfrage

ASSIGNMENT

PNo	Duration
P1	5
P4	4
P1	6
P4	3
P1	4
P3	5
P2	7

Vereinigung

Eingabe

Zwei Relationen R und S

Notation

Die Vereinigung zweier Relationen R und S wird geschrieben als

$$R \cup S$$

Ausgabe

Eine Relation, die die Menge aller Tupel repräsentiert, die in R, in S, oder in beiden enthalten sind.

$$R \cup S = \{t | t \in R \vee t \in S\}$$

Beispiel für Vereinigung

Voraussetzung

R und *S* müssen *vereinigungskompatibel* sein, d.h. *R* und *S* haben die gleiche Anzahl von Attributen und das *i*. Attribut jeder Relation ist über der gleichen Domäne definiert

Beispielanfrage:
Kombiniere die beiden gegebenen Relationen

Formulierung in relationaler Algebra:
$\text{ASSIGNMENT}_1 \cup \text{ASSIGNMENT}_2$

ASSIGNMENT_1

ENo	PNo	Duration
E1	P1	5
E2	P4	4
E2	P1	6

ASSIGNMENT_2

ENo	PNo	Duration
E3	P4	3
E4	P1	4
E4	P3	5
E5	P2	7

RESULT

ENo	PNo	Duration
E1	P1	5
E2	P4	4
E2	P1	6
E3	P4	3
E4	P1	4
E4	P3	5
E5	P2	7

Universität Trier

Mengendifferenz

Eingabe

Zwei Relationen R und S

Notation

Die Mengendifferenz zweier Relationen R und S wird geschrieben als

$$R - S$$

Ausgabe

Eine Relation, die alle Tupel der Mengendifferenz der Relationen R und S enthält, d.h., alle Tupel, die in R enthalten sind, aber nicht in S

$$R - S = R \setminus S = \{t | t \in R \wedge t \notin S\}$$

Beispiel für Mengendifferenz

Voraussetzung

R und *S* müssen *vereinigungskompatibel* sein, d.h., *R* und *S* haben die gleiche Anzahl von Attributen und das *i*. Attribut jeder Relation ist über der gleichen Domäne definiert

Beispielanfrage:
Bilde die Mengendifferenz der beiden gegebenen Relationen

Formulierung in relationaler Algebra:
ASSIGNMENT − ASSIGNMENT$_1$

ASSIGNMENT

ENo	PNo	Duration
E1	P1	5
E2	P4	4
E2	P1	6
E3	P4	3
E4	P1	4
E4	P3	5
E5	P2	7

ASSIGNMENT$_1$

ENo	PNo	Duration
E3	P4	3
E4	P1	4
E4	P3	5
E5	P2	7

RESULT

ENo	PNo	Duration
E1	P1	5
E2	P4	4
E2	P1	6

Kartesisches Produkt

Eingabe

Zwei Relationen R (k_1 Attribute) und S (k_2 Attribute)

Notation

Das kartesische Produkt zweier Relationen R und S wird geschrieben als

$$R \times S$$

Ausgabe

Eine Relation mit $n + m$ Attributen, deren Tupel aus der Konkatenation eines Tupels von R mit einem Tupel von S gebildet werden

$$R \times S = \{(a_1, a_2, \ldots, a_n, b_1, b_2, \ldots, b_m) | (a_1, a_2, \ldots, a_n) \in R \wedge (b_1, b_2, \ldots, b_m) \in S\}$$

Beispiel für das kartesische Produkt

EMPLOYEES

EID	EName	Title
E1	Just Vorfan	Programmer
E2	Ann Joy	Elect. Engineer

ASSIGNMENT

ENo	PNo	Duration
E1	P1	5
E2	P4	4

Beispielanfrage:
Berechne das kartesische Produkt von EMPLOYEES und ASSIGNMENT
Formulierung in relationaler Algebra:
EMPLOYEES $\times$ ASSIGNMENT

RESULT

EID	EName	Title	ENo	PNo	Duration
E1	Just Vorfan	Programmer	E1	P1	5
E1	Just Vorfan	Programmer	E2	P4	4
E2	Ann Joy	Elect. Engineer	E1	P1	5
E2	Ann Joy	Elect. Engineer	E2	P4	4

Umbenennung

Eingabe

Eine Relation R und eine Menge von Umbenennungsoperationen *neu*/*alt*

Notation

Die Umbenennung des Attributs *alt* in *neu* der Relation R wird geschrieben als

$$\rho_{alt\,/\,neu} R$$

Beispiel für die Umbenennung

EMPLOYEES

EID	EName	Title
E1	Just Vorfan	Programmer
E2	Ann Joy	Elect. Engineer

Beispielanfrage:

Nenne das Attribut *Title* von EMPLOYEES in *Position* um

Formulierung in relationaler Algebra:

$\rho_{Position/Title}$ EMPLOYEES

RESULT

EID	EName	Position
E1	Just Vorfan	Programmer
E2	Ann Joy	Elect. Engineer

Weitere relationale Operatoren

Eingabe

Zwei Relationen R und S

Notation

Der Schnitt zweier Relationen R und S wird geschrieben als

$$R \cap S$$

Ausgabe

Der Schnitt zweier Relationen R und S ist die Menge aller Tupel, die in beiden Relationen enthalten sind

$$R \cap S = \{t | t \in R \wedge t \in S\} = R - (R - S)$$

Beispiel für den Schnitt

Voraussetzung

R und *S* müssen *vereinigungskompatibel* sein, d.h., *R* und *S* haben die gleiche Anzahl von Attributen und das *i*. Attribut jeder Relation ist über der gleichen Domäne definiert

Beispielanfrage:
Bestimme den Schnitt der beiden angegebenen Relationen

Formulierung in relationaler Algebra:
ASSIGNMENT $\cap$ ASSIGNMENT$_1$

ASSIGNMENT

ENo	PNo	Duration
E1	P1	5
E2	P4	4
E2	P1	6
E3	P4	3
E4	P1	4
E4	P3	5
E5	P2	7

ASSIGNMENT$_1$

ENo	PNo	Duration
E3	P4	3
E4	P1	4
E4	P3	5
E5	P2	7

RESULT

ENo	PNo	Duration
E3	P4	3
E4	P1	4
E4	P3	5
E5	P2	7

Eingabe

Joinprädikat (Formel/Bedingung) und zwei Relationen R und S

Notation

Der Join (auch θ-Join) zweier Relationen R und S wird geschrieben als

$$R \bowtie_F S$$

wobei F das Joinprädikat ist.

Ausgabe

Eine Relation, deren Tupel jeweils eine Konkatenation von Tupeln aus R und S sind, für die das Joinprädikat erfüllt wird.

$$R \bowtie_F S = \{r \circ s | r \in R \wedge s \in S \wedge F \text{ wird von } r \text{ und } s \text{ erfüllt}\} = \sigma_F(R \times S)$$

Joinprädikat

Das Joinprädikat hat eine ähnliche Form wie ein Selektionsprädikat, seine Terme sind aber von der Form $R.A\ \theta\ S.B$, wobei A und B Attribute von R und S sind.

Equi-Join

Das folgende Beispiel entspricht einem *Equi-Join*, d.h., das Joinprädikat enthält nur den Vergleichsoperator $=$.

Beispiel für Join

EMPLOYEES

EID	EName	Title
E1	Just Vorfan	Programmer
E2	Ann Joy	Elect. Engineer

ASSIGNMENT

ENo	PNo	Duration
E1	P1	5
E2	P4	4
E2	P1	6

Beispielanfrage:
Berechne den Join zwischen EMPLOYEES und ASSIGNMENT mit dem Joinprädikat EMPLOYEES.*EID* = ASSIGNMENT.*ENo*

Formulierung in relationaler Algebra:

$$\text{EMPLOYEES} \bowtie_{\text{EMPLOYEES}.EID=\text{ASSIGNMENT}.ENo} \text{ASSIGNMENT}$$

RESULT

EID	EName	Title	ENo	PNo	Duration
E1	Just Vorfan	Programmer	E1	P1	5
E2	Ann Joy	Elect. Engineer	E2	P4	4
E2	Ann Joy	Elect. Engineer	E2	P1	6

Natürlicher Join

Eingabe

Zwei Relationen R mit Attributen $\{A_1, \ldots, A_n\}$ und S mit Attributen $\{B_1, \ldots, B_m\}$.

Notation

Der natürliche Join zwischen zwei Relationen R und S wird geschrieben als

$$R \bowtie S$$

Ausgabe

Eine Relation mit Tupeln, die eine Konkatenation von jeweils einem Tupel aus R und einem Tupel aus S sind, die identisch auf ihren gemeinsamen Attributen $C_1, C_2, \ldots, C_k$ sind, wobei gemeinsame Attribute nur einmal vorkommen

$$R \bowtie S = \pi_{\{A_1,\ldots,A_n\} \cup \{B_1,\ldots,B_m\}} \{R \bowtie_{(R.C_1 = S.C_1) \wedge \ldots \wedge (R.C_k = S.C_k)} S\}$$

Beispiel für den natürlichen Join

EMPLOYEES

EID	EName	Title
E1	Just Vorfan	Programmer
E2	Ann Joy	Elect. Engineer

ASSIGNMENT

EID	PNo	Duration
E1	P1	5
E2	P4	4
E2	P1	6

Beispielanfrage:

Berechne den natürlichen Join zwischen EMPLOYEES und ASSIGNMENT

Formulierung in relationaler Algebra:

EMPLOYEES $\bowtie$ ASSIGNMENT

RESULT

EID	EName	Title	PNo	Duration
E1	Just Vorfan	Programmer	P1	5
E2	Ann Joy	Elect. Engineer	P4	4
E2	Ann Joy	Elect. Engineer	P1	6

Semijoin

Eingabe

Joinprädikat und zwei Relationen R und S

Notation

Der Semijoin zwischen zwei Relationen R und S wird geschrieben als

$$R \ltimes_F S$$

(es gibt auch $\rtimes$)

Ausgabe

Eine Relation mit Tupeln aus R, die einen Joinpartner in S finden

$$R \ltimes_F S = \{r | r \in R \wedge \exists s \in S \; F \text{ wird von } r \text{ und } s \text{ erfüllt}\} = \pi_{R.*}(R \bowtie_F S)$$

Beispiel für den Semijoin

EMPLOYEES

EID	EName	Title
E1	Just Vorfan	Programmer
E2	Ann Joy	Elect. Engineer
E3	Lilo Pause	Programmer

ASSIGNMENT

ENo	PNo	Duration
E1	P1	5
E2	P4	4

Beispielanfrage:
Berechne den Semijoin zwischen EMPLOYEES und ASSIGNMENT mit dem Joinprädikat EMPLOYEES.*EID* = ASSIGNMENT.*ENo*

Formulierung in relationaler Algebra:

$$\text{EMPLOYEES} \ltimes_{\text{EMPLOYEES}.EID=\text{ASSIGNMENT}.ENo} \text{ASSIGNMENT}$$

RESULT

EID	EName	Title
E1	Just Vorfan	Programmer
E2	Ann Joy	Elect. Engineer

Division (Quotient)

Eingabe

Zwei Relationen R und S

Notation

Die Division zweier Relationen R und S wird geschrieben als

$$R \div S$$

Ausgabe

Eine Relation mit Tupeln aus R eingeschränkt auf die Attribute, die nur in R vorkommen, für die alle möglichen Kombinationen mit Tupeln von S in R vorkommen

$$R \div S = \pi_{R.*-S.*}(R) - \pi_{R.*-S.*}((\pi_{R.*-S.*}(R) \times S) - R$$

Beispiel für die Division

EMPLOYEES

EName	Title
Just Vorfan	Programmer
Ann Joy	Elect. Engineer
Ann Joy	Programmer
Claire Grube	Mech. Engineer
John Doe	Syst. Analyst

TITLES

Title
Programmer
Elect. Engineer

Beispielanfrage: Berechne die Division von EMPLOYEES und TITLES

Formulierung in relationaler Algebra:
EMPLOYEES ÷ TITLES

RESULT

EName
Ann Joy

Die Ergebnisrelation enthält Angestellte, die Einträge für beide Titel in TITLES haben.

Übersetzung von SQL in relationale Algebra

Übersetzen von SQL in relationale Algebra

Struktur einer SQL-Anfrage:

select distinct $a_1, \ldots, a_n$
from $R_1, \ldots, R_n$
where p

Algorithmus:

1. Übersetzen der **from**-Klausel

Seien $R_1, \ldots, R_k$ die Relationen in der **from**-Klausel der Anfrage.

Konstruiere den Ausdruck:

$$R = \begin{cases} R_1 & \text{falls } k = 1 \\ ((\ldots(R_1 \times R_2) \times \ldots) \times R_k) & \text{sonst} \end{cases}$$

Algorithmus:

2. Übersetzen der **where**-Klausel

Sei F das Prädikat in der **where**-Klausel der Anfrage (falls eine **where**-Klausel existiert).

Konstruiere den Ausdruck:

$$W = \begin{cases} R & \text{falls es keine } \textbf{where}\text{-Klausel gibt} \\ \sigma_F(R) & \text{sonst} \end{cases}$$

Algorithmus:

3. Übersetzen der **select**-Klausel

Sei $a_1, \ldots, a_n$ (oder "*") die Projektion in der **select**-Klausel der Anfrage

Konstruiere den Ausdruck:

$$S = \begin{cases} W & \text{falls die Projektion "*" ist} \\ \pi_{a_1,\ldots,a_n}(W) & \text{sonst} \end{cases}$$

Output:
S

Übersetzen von SQL in relationale Algebra

Beispielanfrage

select distinct *e.EName*, *s.Salary*
from EMPLOYEES *e*, SALARY *s*
where *e.Title* = *s.Title* **and** *s.Salary* $\geq$ 60.000

$$R = \begin{cases} R_1 & \text{falls } k = 1 \\ ((\ldots(R_1 \times R_2) \times \ldots) \times R_k) & \text{sonst} \end{cases}$$

$$R = \text{EMPLOYEES} \times \text{SALARY}$$

$$W = \begin{cases} R & \text{falls es keine } \textbf{where}\text{-Klausel gibt} \\ \sigma_F(R) & \text{sonst} \end{cases}$$

$$W = \sigma_{\text{EMPLOYEES}.Title = \text{SALARY}.Title \textbf{ and } Salary \geq 60.000}(R)$$
$$= \sigma_{\text{EMPLOYEES}.Title = \text{SALARY}.Title \textbf{ and } Salary \geq 60.000}(\text{EMPLOYEES} \times \text{SALARY})$$

Universität Trier

Übersetzen von SQL in relationale Algebra...

$$W = \sigma_{\text{EMPLOYEES}.Title=\text{SALARY}.Title \textbf{ and } Salary \geq 60.000}(\text{EMPLOYEES} \times \text{SALARY})$$

$$S = \begin{cases} W & \text{falls die Projektion “*” ist} \\ \pi_{a_1,\ldots,a_n}(W) & \text{sonst} \end{cases}$$

$$\begin{aligned} S &= \pi_{EName,Salary}(W) \\ &= \pi_{EName,Salary}(\sigma_{\text{EMPLOYEES}.Title=\text{SALARY}.Title \textbf{ and } Salary \geq 60.000}(\text{EMPLOYEES} \times \text{SALARY})) \end{aligned}$$

Algebra-Ausdruck

$$\pi_{EName,Salary}(\sigma_{\text{EMPLOYEES}.Title=\text{SALARY}.Title \textbf{ and } Salary \geq 60.000}(\text{EMPLOYEES} \times \text{SALARY}))$$

Universität Trier

[Özsu Valduriez, 2011] M. Tamer Özsu, P. Valduriez.
Principles of Distributed Database Systems.
Third Edition, Springer, 2011.

[Dadam, 1996] P. Dadam.
Verteilte Datenbanken und Client/Server-Systeme.
Springer-Verlag, Berlin, Heidelberg 1996.

[Toerey, 1999] Toby J. Teorey
Database modeling and design
Third Edition, Morgan Kaufmann Publishers, San Francisco, CA, 1999

[Kossmann, 2000] D. Kossmann.
The State of the Art in Distributed Query Processing,
ACM Computing Surveys,
Vol. 32, No. 4, 2000, S. 422-469.