5. Eine temporal erweiterte Relationenalgebra [1]

Klassische Relationenalgebra

Eine **Relation** ist ein Paar $X = (S, R)$:

- S heißt **Schema** von X und ist eine Funktion $S\colon \{a_1, \ldots, a_n\} \longrightarrow \mathcal{D}$, wobei $\{a_1, \ldots, a_n\} =: \mathrm{attr}(S)$ eine Menge von **Attribut(nam)en** und $\mathcal{D}$ eine Menge von **Wertebereichen** oder **Datentypen** ist.

  S ordnet also jedem Attribut a einen Wertebereich $S(a) \in \mathcal{D}$ zu. Die Menge aller den Attributen zugeordneten Wertebereiche heißt $\mathrm{domains}(S) \subseteq \mathcal{D}$.

- Ein Tupel $t\colon \mathrm{attr}(S) \longrightarrow \bigcup_{d \in \mathcal{D}} d$ bzgl. S ordnet jedem Attribut a des Schemas einen Wert aus dem zugehörigen Wertebereich $S(a)$ zu. Eine *endliche* Teilmenge R solcher Tupel heißt **Inhalt** von X.

Die folgenden Operatoren bilden eine Algebra auf der Menge der Relationen. Seien $X = (S^X, R^X)$ und $Y = (S^Y, R^Y)$ zwei Relationen.

[1] basiert auf: Stahlhut,C.: Semantik und Realisierung einer erweiterten Relationenalgebra für temporale Datenbanken. Diplomarbeit, Institut für Informationssysteme, Uni Hannover, 2004

Relationenalgebra (Forts.)

Projektion

Sei A eine Teilmenge der Attributnamen von X: $A \subseteq \text{attr}(S^X)$.

$$\pi_A(X) := (S', R') \text{ mit } S' = S^X|_A \text{ und } R' = \{t|_A : t \in R^X\}$$

Selektion

Sei φ eine Selektionsfunktion $\varphi : R^X \to \{\text{wahr, falsch}\}$.

$$\sigma_\varphi(X) := (S^X, R') \text{ mit } R' = \{t \in R^X : \varphi(t) = \text{wahr}\}$$

Vereinigung (Differenz, Durchschnitt analog)

Falls X und Y das gleiche Schema ($S^X = S^Y$) haben, gilt:

$$X \cup Y := (S', R') \text{ mit } S' = S^X = S^Y \text{ und } R' = \{t : t \in R^X \vee t \in R^Y\}$$

Natürlicher Verbund, Join

$X \bowtie Y := (S', R')$ mit

$$S' : \text{attr}(S^X) \cup \text{attr}(S^Y) \longrightarrow \text{domains}(S^X) \cup \text{domains}(S^X)$$

$$\text{mit } S'|_{\text{attr}(S^X)} = S^X \text{ und } S'|_{\text{attr}(S^Y)} = S^Y$$

$$R' = \{t \text{ bzgl. } S' : \exists x \in R^X \wedge \exists y \in R^Y \text{ mit } t|_{\text{attr}(S^X)} = x \wedge t|_{\text{attr}(S^Y)} = y\}$$

Relationenalgebra (Forts.)

Erweiterung (um ein Attribut b)

Sei M eine Menge, $b \notin \text{attr}(S^X)$ ein neuer Attributname und β eine Funktion der Form $\beta : R^X \to M$. Die Erweiterung von X um $b = \beta$ ist definiert als:

$$\varepsilon_{b=\beta}(X) := (S', R') \text{ mit}$$

$$S' : \text{attr}(S^X) \cup \{b\} \longrightarrow \text{domains}(S^X) \cup \{M\}$$

$$\text{mit } S'|_{\text{attr}(S^X)} = S^X \text{ und } S'(b) = M$$

$$R' = \{t' \text{ bzgl. } S' : \exists t \in R^X \text{ mit } t'|_{\text{attr}(S^X)} = t \text{ und } t'(b) = \beta(t)\}$$

Umbenennung (des Attributs a in b)

Sei $a \in \text{attr}(S^X)$ ein Attribut von X und $b \notin \text{attr}(S^X)$ ein neuer Attributname. Außerdem sei $\beta : R^X \to S(a)$ mit $\beta(t) = t(a)$ die Wertezuordnung für a. Dann kann das Attribut a wie folgt in b umbenannt werden:

$$\rho_{b=a} := \pi_{\text{attr}(S^X) \cup \{b\} - \{a\}}(\varepsilon_{b=\beta}(X))$$

Mehrere Umbenennungen $\ldots(\rho_{b_2=a_2}(\rho_{b_1=a_1}(X)))$ können als $\rho_{b_1=a_1, b_2=a_2, \ldots}(X)$ notiert werden.

Relationenalgebra (Forts.)

Gruppierung und Aggregierung

Sei A eine Teilmenge der Attributnamen von X: $A \subseteq \text{attr}(S^X)$.
Seien seien außerdem $b_i \notin \text{attr}(S^X)$, $i=1,..,n$ neue Attributnamen, M_i Mengen und α_i Aggregierungsfunktionen $\alpha_i : \mathcal{P}(R^X) \longrightarrow \mathcal{P}(M_i)$. Dann gilt[2]:

$$\Gamma_{A\,\#\,b_1=\alpha_1,\ldots,b_n=\alpha_n}(X) := (S', R') \text{ mit}$$

$$S' \colon \{b_1, \ldots, b_n\} \longrightarrow \{M_1, \ldots, M_n\} \text{ mit } S'(b_i) = M_i \quad \text{für } i=1,..,n$$

$$R' = \{\, t' \text{ bzgl. } S' : \exists t \in \pi_A(X) \wedge t'(b_i) \in \alpha_i(\{x \in X : x|_A = t\}) \text{ für } i=1,..,n \,\}$$

- Beachte: In dieser Definition liefert eine Aggregierungsfunktion anstelle eines *Werts* (aus einer Menge M) eine *Menge von Werten* (aus $\mathcal{P}(M)$) zurück.
- Übliche Abkürzungen für eine Gruppe $G \subseteq R^X$ und ein Attribut $a \in \text{attr}(S^X)$:
  - `count`$(*)$ für $\alpha(G) := \{|G|\}$
  - `min`(a) für $\alpha(G) := \{\min_{t \in G} t(a)\}$
  - `sum`(a) für $\alpha(G) := \{\sum_{t \in G} t(a)\}$
  - a für $\alpha(G) := \{t(a) : t \in G\}$

[2]Falls $X = (S^X, R^X)$, schreiben wir auch einfach X statt R^X, z.B. $t \in X$ statt $t \in R^X$.

Relationenalgebra (Forts.)

Beispiele:

Sei folgende Relation gegeben:

$X :=$

Hersteller	Produkt	Preis	Lieferant
Snel	Rennrad	309	Zeus
Robus	Rennrad	235	Zeus
Snel	Mountainbike	235	Zeus

Dann liefern die folgenden Gruppierungen und Aggregierungen:

$\Gamma_{\text{Produkt \# Produkt, Mittel = }\texttt{avg}\text{(Preis)}}(X) =$

Produkt	Mittel
Rennrad	272
Mountainbike	235

$\Gamma_{\text{Lieferant \# Lieferant, Faktor = }\texttt{primfaktoren}\text{(Preis)}}(X) =$

Lieferant	Faktor
Zeus	3
Zeus	103
Zeus	5
Zeus	47

Intervalle

Für $b, e \in \mathbb{Z}$ mit $b \leq e$ sei **das Intervall von b bis e** definiert als

$$[b, e] := \{i \in \mathbb{Z} : b \leq i \leq e\}$$

Sei $\mathcal{I}$ die Menge aller so definierten Intervalle[3].

Operatoren auf Intervallen

Für ein Intervall $I = [b, e] \in \mathcal{I}$ gelte:

- $\texttt{begin}(I) := b$
- $\texttt{end}(I) := e$
- $\texttt{length}(I) := e - b + 1$

Vergleichsoperatoren liefern Information über die Beziehung zweier Intervalle:

$$\mathcal{I} \times \mathcal{I} \longrightarrow \{\text{wahr, falsch}\}$$

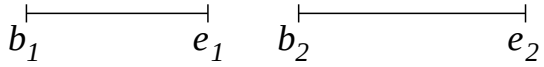
Seien also $I_1 := [b_1, e_1] \in \mathcal{I}$ und $I_2 := [b_2, e_2] \in \mathcal{I}$ zwei Intervalle.

[3] In diesem Kapitel rechnen wir mit nichtleeren, geschlossenen Intervallen; Umrechnung: $[b, e] = [b, e+1)$

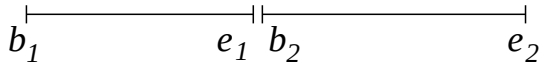
Operatoren auf Intervallen (Forts.)

Folgende Vergleichsoperatoren basieren auf Allen's temporalen Beziehungen:

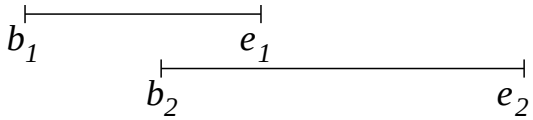
$\overset{asymm.}{\rightarrow}$ I_1 `before`$(<)$ $I_2 :\Leftrightarrow e_1 < b_2$



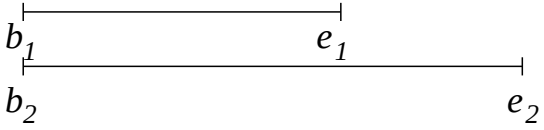
$\overset{symm.}{-}$ I_1 `meets` $I_2 :\Leftrightarrow b_1 = e_2 + 1 \vee b_2 = e_1 + 1$



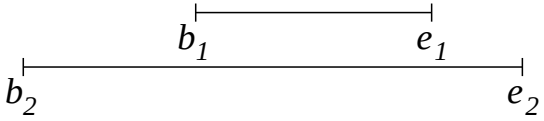
$-$ I_1 `overlaps` $I_2 :\Leftrightarrow b_1 \leq e_2 \wedge b_2 \leq e_1$



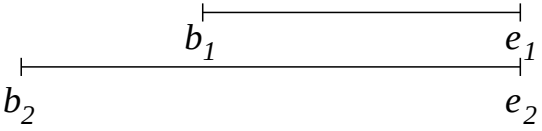
$\rightarrow$ I_1 `begins_with` $I_2 :\Leftrightarrow b_1 = b_2 \wedge e_1 \leq e_2$



$\rightarrow$ I_1 `is_included_in`$(\subseteq)$ $I_2 :\Leftrightarrow b_1 \geq b_2 \wedge e_1 \leq e_2$



$\rightarrow$ I_1 `ends_with` $I_2 :\Leftrightarrow e_1 = e_2 \wedge b_1 \geq b_2$



$-$ I_1 `equals`$(=)$ $I_2 :\Leftrightarrow b_1 = b_2 \wedge e_1 = e_2$

Operatoren auf Intervallen (Forts.)

Weitere Vergleichsoperatoren sind:

$\rightarrow\ I_1\,\texttt{after}(>)\,I_2 :\Leftrightarrow I_2\,\texttt{before}\,I_1$

$\rightarrow\ I_1\,\texttt{includes}(\supseteq)\,I_2 :\Leftrightarrow I_2\,\texttt{is\_included\_in}\,I_1$

$\rightarrow\ I_1 \subset I_2 :\Leftrightarrow I_1 \subseteq I_2 \wedge I_1 \neq I_2$

$\rightarrow\ I_1 \supset I_2 :\Leftrightarrow I_1 \supseteq I_2 \wedge I_1 \neq I_2$

$\rightarrow\ I_1\,\texttt{precedes}\,I_2 :\Leftrightarrow I_1\,\texttt{before}\,I_2 \wedge I_1\,\texttt{meets}\,I_2$ (↑ asymmetrische Operatoren)

$-\ I_1\,\texttt{merges}\,I_2 :\Leftrightarrow I_1\,\texttt{meets}\,I_2 \vee I_1\,\texttt{overlaps}\,I_2$ (symmetrischer Operator)

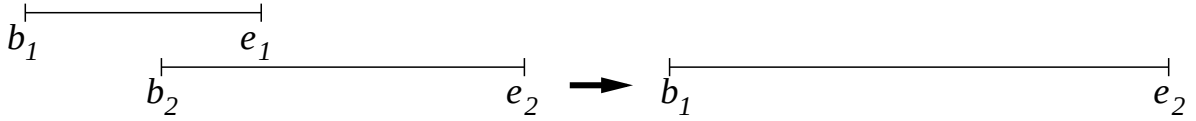
Zusätzlich lässt sich der Spezialfall $i \in I_1$ für ein $i \in \mathbb{Z}$ definieren:

$$i \in I_1 \ :\Leftrightarrow\ [i,i]\,\texttt{is\_included\_in}\,I_1 \ \Leftrightarrow\ b_1 \leq i \leq e_1$$

Um zwei Intervalle miteinander zu einem neuen Intervall zu kombinieren, werden Operatoren der Form $\mathcal{I} \times \mathcal{I} \rightarrow \mathcal{I}$ benötigt. Mit $I_1 := [b_1, e_1] \in \mathcal{I}$ und $I_2 := [b_2, e_2] \in \mathcal{I}$ sei:

Operatoren auf Intervallen (Forts.)

- $$I_1 \,\texttt{union}\, I_2 := [\min(b_1, b_2),\ \max(e_1, e_2)] \quad \text{falls } I_1 \,\texttt{merges}\, I_2$$



- $$I_1 \,\texttt{minus}\, I_2 := \begin{cases} [b_1,\ \min(b_2 - 1, e_1)] & \text{falls } b_1 < b_2 \wedge e_1 \leq e_2 \\ [\max(e_2 + 1, b_1),\ e_1] & \text{falls } b_1 \geq b_2 \wedge e_1 > e_2 \end{cases}$$

- $$I_1 \,\texttt{intersect}\, I_2 := [\max(b_1, b_2),\ \min(e_1, e_2)] \quad \text{falls } I_1 \,\texttt{overlaps}\, I_2$$

- In allen anderen Fällen sei das Ergebnis undefiniert.

Operatoren auf Mengen von Intervallen

Die beiden folgenden Operatoren bilden eine Teilmenge von $\mathcal{I}$ auf eine Teilmenge von $\mathcal{I}$ ab: $\mathcal{P}(\mathcal{I}) \to \mathcal{P}(\mathcal{I})$. Sei nun $\mathcal{X} \in \mathcal{P}(\mathcal{I})$.

Der Operator **expand** zerlegt alle übergebenen Intervalle in Einheitsintervalle (d.h. Intervalle I mit $\texttt{end}(I) = \texttt{begin}(I)$):

$$\texttt{expand}(\mathcal{X}) := \{[b, b] : \exists I \in \mathcal{X} \wedge b \in I\}$$

Z. B.:

$$\texttt{expand}(\{[2,2],[4,6],[5,7],[8,9]\}) = \{[2,2],[4,4],[5,5],[6,6],[7,7],[8,8],[9,9]\}$$

Im Gegensatz dazu „verschmilzt“ **collapse** alle Intervalle, für die dies möglich ist. Z. B.:

$$\begin{aligned}\texttt{collapse}(\{[2,2],[4,6],[5,7],[8,9]\}) &= \{[2,2],[4,9]\} \\ &= \texttt{collapse}(\texttt{expand}(\{[2,2],[4,6],[5,7],[8,9]\}))\end{aligned}$$

`collapse` (Forts.)

(Iterative) Definition: Zu jedem Intervall $X \in \mathcal{X}$ sei die transitive Hülle $T^*(X)$ bezüglich `merges` gegeben:

$$\begin{aligned}
&\mathcal{T}^0(X) := \{X\} \\
&\mathcal{T}^n(X) := \{I \in \mathcal{X} : \exists J \in \mathcal{T}^{n-1}(X) \wedge I \texttt{ merges } J\} \quad \forall n \in \mathbb{N}_{\geq 1} \\
&\mathcal{T}^*(X) := \text{ein (beliebiges) } \mathcal{T}^n(X) \text{ mit } \mathcal{T}^n(X) = \mathcal{T}^{n+1}(X)
\end{aligned}$$

Vereinigt (`union`) man nun alle Intervalle in jeder Hülle zu einem einzigen Intervall, ergibt sich `collapse` als Vereinigung dieser Intervalle:

$$\begin{aligned}
\texttt{collapse}(\mathcal{X}) &:= \bigcup_{X \in \mathcal{X}} \{\, \texttt{union}_{I \in \mathcal{T}^*(X)}\ I \,\} \\
&= \bigcup_{X \in \mathcal{X}} \{\, [\, \min_{I \in \mathcal{T}^*(X)}(\texttt{begin}(I)), \max_{I \in \mathcal{T}^*(X)}(\texttt{end}(I)) \,] \,\}
\end{aligned}$$

Grundbausteine für die Erweiterung der Relationenalgebra:

`pack` und `unpack`

Mit den nun definierten Operatoren auf Intervallen lässt sich die Relationenalgebra um Funktionalität für Zeitstempel (bzw. Intervalle) erweitern.

Hierzu werden **`expand`** und **`collapse`** als Aggregierungsfunktionen aufgefasst, damit sie auf intervallwertige Attribute einer Relation wirken können. Diese neu entstehenden Operatoren dienen dazu eine „atomare“, „zeitpunktartige“ Sicht auf die Information in einer Relation zu erhalten (**`unpack`**), bzw. diese Information möglichst „zusammengefasst“ und „redundanzfrei“ betrachten zu können (**`pack`**).

Im folgenden sei $X = (S, R)$ eine Relation. Außerdem seien

- v der Name eines intervallwertigen Attributs von X, z.B. $v = \mathrm{VT}$;
- $A := \mathrm{attr}(S) - \{v\}$ sei die Menge der Attributnamen von X *ohne* v
- und $G \subseteq R$ eine Gruppe von Tupeln.

`collapse` und `expand` als Aggregierungsfunktionen

Man kann **`collapse`** und **`expand`** als Aggregierungsfunktionen auffassen:

$$\texttt{collapse}(v) : \mathcal{P}(R) \longrightarrow \mathcal{P}(\mathcal{I}) \quad \text{mit}$$

$$\texttt{collapse}(v)(G) := \texttt{collapse}(\{t(v) : t \in G\})$$

und analog

$$\texttt{expand}(v) : \mathcal{P}(R) \longrightarrow \mathcal{P}(\mathcal{I}) \quad \text{mit}$$

$$\texttt{expand}(v)(G) := \texttt{expand}(\{t(v) : t \in G\})$$

`pack` und `unpack` für ein (einzelnes) Attribut

Mit diesen neuen Aggregierungsfunktionen lassen sich nun **`pack`** und **`unpack`** für ein Attribut v definieren:

$$\texttt{pack}^{v}(X) := \Gamma_{A\#A,\, v=\texttt{collapse}(v)}(X)$$

und analog

$$\texttt{unpack}^{a}(X) := \Gamma_{A\#A,\, v=\texttt{expand}(v)}(X)$$

`pack` und `unpack` für ein (einzelnes) Attribut (Forts.)

Beispiel:

$X :=$

Name	VT
Müller	[2, 5]
Müller	[4, 6]
Schmidt	[5, 7]
Schmidt	[8, 9]
Schmidt	[12, 13]

$\texttt{pack}^{\text{VT}}(X) = \Gamma_{\text{Name \# Name, VT=}\texttt{collapse}(\text{VT})}$

$=$

Name	VT
Müller	[2, 6]
Schmidt	[5, 9]
Schmidt	[12, 13]

$\texttt{unpack}^{\text{VT}}(X) =$

Name	VT
Müller	[2, 2]
Müller	[3, 3]
Müller	[4, 4]
Müller	[5, 5]
Müller	[6, 6]
Schmidt	[5, 5]
Schmidt	[6, 6]
Schmidt	[7, 7]
Schmidt	[8, 8]
Schmidt	[9, 9]
Schmidt	[12, 12]
Schmidt	[13, 13]

`pack` und `unpack` für mehrere Attribute

Sei $V := v_1, ..., v_n,\ n \geq 1$ eine (nichtleere) *Liste* von Attributnamen:

$$\texttt{unpack}^V(X) := \texttt{unpack}^{v_n}(\texttt{unpack}^{v_{n-1}}(\ldots \texttt{unpack}^{v_1}(X)\ldots))$$
$$\texttt{pack}^V(X) := \texttt{pack}^{v_n}(\texttt{pack}^{v_{n-1}}(\ldots \texttt{pack}^{v_1}(\texttt{unpack}^V(X))\ldots))$$

Beispiel:

Für $X =$

S#	P#
[1, 2]	[3, 4]
[5, 6]	[7, 8]

gilt:

$$\texttt{unpack}^{\text{S\#,P\#}}(X) =$$

S#	P#
[1, 1]	[3, 3]
[1, 1]	[4, 4]
[2, 2]	[3, 3]
[2, 2]	[4, 4]
[5, 5]	[7, 7]
[5, 5]	[8, 8]
[6, 6]	[7, 7]
[6, 6]	[8, 8]

$= \texttt{unpack}^{\text{P\#,S\#}}(X).$

Diese Kommutativität der Attributangaben gilt für `unpack` allgemein.

`pack` und `unpack` für mehrere Attribute (Forts.)

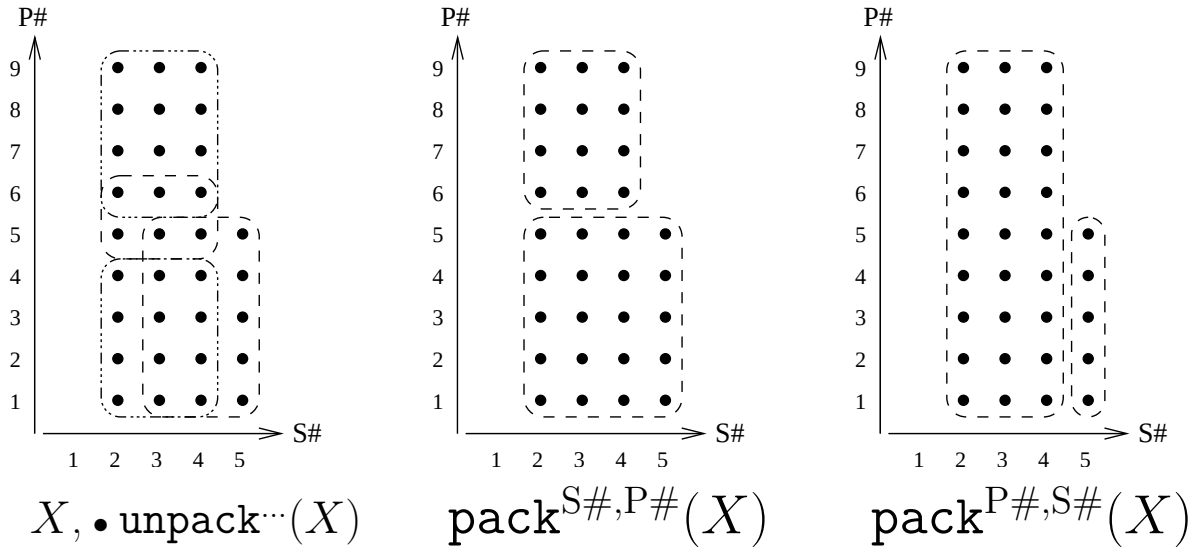
Für $X =$

S#	P#
$[3,5]$	$[1,5]$
$[2,4]$	$[1,4]$
$[2,4]$	$[5,6]$
$[2,4]$	$[6,9]$

gilt jedoch – ACHTUNG –

$$\texttt{pack}^{\mathrm{S\#},\mathrm{P\#}}(X) = \begin{array}{|c|c|}\hline \mathrm{S\#} & \mathrm{P\#} \\ \hline [2,5] & [1,5] \\ [2,4] & [6,9] \\ \hline \end{array} \neq \begin{array}{|c|c|}\hline \mathrm{S\#} & \mathrm{P\#} \\ \hline [2,4] & [1,9] \\ [5,5] & [1,5] \\ \hline \end{array} = \texttt{pack}^{\mathrm{P\#},\mathrm{S\#}}(X).$$

vgl. graphisch:



$X, \bullet\,\texttt{unpack}^{\cdots}(X)$ $\quad$ $\texttt{pack}^{\mathrm{S\#},\mathrm{P\#}}(X)$ $\quad$ $\texttt{pack}^{\mathrm{P\#},\mathrm{S\#}}(X)$

Erweiterung der Relationenalgebra um Sequentielle Operatoren

Mit den obigen Definitionen ist es nun möglich, "sequentielle" Versionen von Operatoren der Relationenalgebra zu definieren. Sei V eine nichtleere *Liste* von Attributnamen

Dyadische Operatoren: So werde etwa die sequentielle Differenz zweier Relationen X und Y neu definiert als:

$$X -^V Y := \mathtt{pack}^V(\mathtt{unpack}^V(X) - \mathtt{unpack}^V(Y))$$

Natürlich müssen auch die gleichen Voraussetzungen gelten wie für die Original-Differenz, d.h. auch hier müssen X und Y das gleiche Schema haben.

Analog werden $\cup^V, \cap^V$ und $\bowtie^V$ definiert.

Monadische Operatoren: Entsprechendes gilt für Operatoren, die auf eine einzige Relation wirken, z.B.:

$$\pi^V_A(X) := \mathtt{pack}^V(\pi_A(\mathtt{unpack}^V(X)))$$

analog für $\sigma^V_\varphi, \varepsilon^V_{b=\beta}$ und $\Gamma^V_{A\#b_1=\alpha_1,\ldots,b_q=\alpha_q}$

Möglicherweise wird nur das anfängliche `unpack` oder das abschließende `pack` ausgeführt, nicht aber beides: Durch Projektion, Umbenennung oder Aggregierung kann das in V aufgeführte Attribut entfallen (kein `pack`) oder erst entstehen (kein `unpack`).

Beispielanfragen

$\rightarrow \sigma,\ \pi$

Erste Beispieldatenbank:

ANG:=

Nr	Name	Gehalt	ChefNr	VT
12	Müller	5000	27	[3, 7]

...

„Wie lauten die Daten der derzeit angestellten Personen?“

$$\sigma_{\texttt{now} \in \mathrm{VT}}(\mathbf{ANG})$$

„Wie lautet der Beschäftigungszeitraum eines Angestellten?“

$$\pi^{\mathrm{VT}}_{\mathrm{Name},\ \mathrm{VT}}(\mathbf{ANG})$$

„In welchen Zeiträumen war das Gehalt eines Angestellten konstant?“

$$\pi_{\mathrm{Name},\ \mathrm{VT}}(\pi^{\mathrm{VT}}_{\mathrm{Name},\ \mathrm{Gehalt},\ \mathrm{VT}}(\mathbf{ANG}))$$

„Wer verdient(e) wann mehr als 5250?“

$$\pi^{\mathrm{VT}}_{\mathrm{Name},\ \mathrm{VT}}(\sigma_{\mathrm{Gehalt}>5250}(\mathbf{ANG}))$$

Beispielanfragen (Forts.) $\rightarrow\bowtie$

„Wer verdiente [gleichzeitig] mehr als sein Chef und wie hieß der Chef?“

$(\mathbf{ANG}\ \mathrm{X})^{\setminus \mathrm{VT}}$ stehe für die Umbenennung aller Attribute bis auf VT:

$$\mathrm{X} := \rho_{\mathrm{X.Nr}\ =\ \mathrm{Nr},\ \mathrm{X.Name}\ =\ \mathrm{Name},\ \ldots}\ (\mathbf{ANG})$$

$$\pi_{\mathrm{X.Name},\ \mathrm{Y.Name}}(\sigma_{\mathrm{X.ChefNr}\ =\ \mathrm{Y.Nr}\ \wedge\ \mathrm{X.Gehalt}\ >\ \mathrm{Y.Gehalt}}((\mathbf{ANG}\ \mathrm{X})^{\setminus \mathrm{VT}}\ \bowtie^{\mathrm{VT}}\ (\mathbf{ANG}\ \mathrm{Y})^{\setminus \mathrm{VT}}))$$

„Welche Angestellten haben jemals eine Gehaltserhöhung erhalten?“

$$\pi_{\mathrm{Name}}(\sigma_{\mathrm{X.VT}\ \texttt{before}\ \mathrm{Y.VT}\ \wedge\ \mathrm{X.Gehalt}\ <\ \mathrm{Y.Gehalt}}((\mathbf{ANG}\ \mathrm{X})^{\setminus \mathrm{Nr}}\ \bowtie\ (\mathbf{ANG}\ \mathrm{Y})^{\setminus \mathrm{Nr}}))$$

„Wie lauteten alle Angestelltendaten, als *jemand 10000 verdient hat?“*

$$\mathbf{ANG}\ \bowtie^{\mathrm{VT}}\ \pi_{\mathrm{VT}}(\sigma_{\mathrm{Gehalt}\ =\ 10000}(\mathbf{ANG}))$$

Beispielanfragen (Forts.)

Zweite Beispieldatenbank:

Relation **S**:

S#	during
2	[2, 4]

...

— stellt die Vertragslaufzeit (during) eines Zulieferers (S# = supplier no.) dar, also hier: *„Zulieferer 2 ist oder war von 2 bis 4 unter Vertrag.“*

Relation **SP**:

S#	P#	during
2	A	[2, 3]

...

— stellt die Angebotszeiten (during) von Teilen (P# = part no.) durch Zulieferer dar, also hier: *„Zulieferer 2 kann Teil ’A’ von 2 bis 3 liefern.“*

Beispielanfragen (Forts.) $\rightarrow \bowtie, -$

„Welche Zulieferer konnten gleichzeitig die Teile 'A' und 'B' liefern?“

$$AB := \pi_{\mathrm{S\#},\,\mathrm{during}}(\pi_{\mathrm{S\#},\,\mathrm{during}}(\sigma_{\mathrm{P\#='A'}}(\mathbf{SP})) \bowtie^{\mathrm{during}} \pi_{\mathrm{S\#},\,\mathrm{during}}(\sigma_{\mathrm{P\#='B'}}(\mathbf{SP})))$$

„Welche Zulieferer konnten nie gleichzeitig 'A' und 'B' liefern?“

$$\pi_{\mathrm{S\#}}(\mathbf{S}) - AB$$

„In welchen Zeiträumen konnte ein Zulieferer keine Teile liefern?“

$$\mathbf{S} -^{\mathrm{during}} \pi_{\mathrm{S\#},\,\mathrm{during}}(\mathbf{SP})$$

„In welchen Zeiträumen war mindestens ein Zulieferer unter Vertrag?“

$$\pi^{\mathrm{during}}_{\mathrm{during}}(\mathbf{S})$$

„In welchen Zeiträumen war kein Zulieferer unter Vertrag?“

Sei C eine Relation mit dem intervallwertigen Attribut 'during', welche das Tupel $[0, \texttt{max}]$ enthält, wobei `max` der letzte darstellbare Zeitpunkt ist.

$$C -^{\mathrm{during}} \pi^{\mathrm{during}}_{\mathrm{during}}(\mathbf{S}) \;=\; C -^{\mathrm{during}} \pi_{\mathrm{during}}(\mathbf{S})$$

Beispielanfragen (Forts.)

→ Spezielles

„Welche Zulieferer, die früher unter Vertrag waren, sind jetzt wieder unter Vertrag?“

$$\pi_{\text{S\#}}(\sigma_{\texttt{now}\in\text{during}}(\mathbf{S})) \cap \pi_{\text{S\#}}(\sigma_{\texttt{now}>\texttt{end}(\text{during})}(\mathbf{S}))$$

„Wie lauten die Paare von (verschiedenen) Zulieferern, die zum selben Zeitpunkt das gleiche Teil neu liefern konnten?“

– unter der Maximalitätsannahme $\mathbf{SP} = \texttt{pack}^{\text{during}}(\mathbf{SP})$ –

$$\pi_{\text{X.S\#, Y.S\#}}(\sigma_{\texttt{begin}(\text{X.during}) = \texttt{begin}(\text{Y.during}) \wedge \text{X.S\#} < \text{Y.S\#}}((\mathbf{SP}\ \text{X})^{\setminus \text{P\#}} \bowtie (\mathbf{SP}\ \text{Y})^{\setminus \text{P\#}}))$$

„Welche Zulieferer konnten wann welche Teile liefern?“

Für eine kompakte Darstellung erscheint es sinnvoll, nicht nur die Zeiträume, sondern auch die Teile(nummern) als Intervalle auszugeben. Hierzu wird zunächst ein Einheitsintervall für jede Teilenummer gebildet. Die größtmöglichen Intervalle werden dann ausgegeben.

$$\pi^{\text{during, parts}}_{\text{S\#, parts, during}}(\varepsilon_{\text{parts}=[\text{P\#,P\#}]}(\mathbf{SP}))$$

Beispielanfragen zu Gruppierung und Aggregierung

Dritte Beispieldatenbank:

Relation **R**:

Hersteller	Produkt	Preis	Zeit
Teua	Rennrad	300	$[1, 7]$
Snel	Rennrad	250	$[8, 14]$
Snel	Rennrad	270	$[15, 20]$
Teua	Rennrad	280	$[18, 25]$
Snel	Mountainbike	270	$[1, 10]$
Robus	Mountainbike	270	$[11, 16]$
Robus	Mountainbike	240	$[17, 30]$
Snel	E-Bike	500	$[11, 32]$

„Was ist der höchste Preis pro Produkt?“

$$PP := \Gamma_{\text{Produkt \# Produkt, Höchstpreis} = \texttt{max}(\text{Preis})}(\mathbf{R})$$

Eine Formulierung mit dem erweiterten Operator $\Gamma^{\text{Zeit}}_{\ldots}$ führt zum gleichen Ergebnis, denn das Maximum ist von der Anzahl der Tupel einer Gruppierung unabhängig: $\texttt{unpack}^{\text{Zeit}}(\mathbf{R})$ enthält zwar mehr Tupel als R, jedoch bleibt der Wertebereich von 'Preis' gleich.

Beispielanfragen (Forts.) $\rightarrow \Gamma$

„Was ist der Höchstpreis pro Produkt und wann wird er angenommen?“

$$\pi^{\text{Zeit}}_{\text{Produkt,Preis,Zeit}} \left(\mathbf{R} \bowtie \left(\rho_{\text{Preis=Höchstpreis}} \left(PP \right) \right) \right)$$

„Wieviele verschiedene Produkte bietet ein Hersteller an?“

$$\Gamma_{\text{Hersteller \# Hersteller, Anzahl} = \texttt{count}(*)} \left(\pi_{\text{Hersteller, Produkt}} \left(\mathbf{R} \right) \right)$$

Diese Anfrage soll z.B. für den Hersteller 'Snel' die Anzahl '3' liefern, unabhängig von Zeiten.

„Wieviele verschiedene Produkte bietet ein Hersteller wann an?“

$$\Gamma^{\text{Zeit}}_{\text{Hersteller, Zeit \# Hersteller, Zeit, Anzahl} = \texttt{count}(*)} \left(\pi_{\text{Hersteller,Produkt,Zeit}} \left(\mathbf{R} \right) \right)$$

Γ^{Zeit} gruppiert und zählt pro Hersteller und Zeitpunkt, und fasst dann zusammen. Z.B. hat der Hersteller 'Snel' im Zeitraum [8,20] immer '2' Produkte angeboten.

Beispielanfragen (Forts.)

→ $\Gamma_{\ldots\text{avg}\ldots}$

„Wie hoch ist der Durchschnittspreis pro Produkt?“

$$\text{(i)}\ \Gamma_{\text{Produkt \# Produkt, Durchschnitt} = \texttt{avg}(\text{Preis})}(\mathbf{R})$$

$$\text{(ii)}\ \Gamma^{\text{Zeit}}_{\text{Produkt \# Produkt, Durchschnitt} = \texttt{avg}(\text{Preis})}(\mathbf{R})$$

$$\text{(iii)}\Gamma^{\text{Zeit}}_{\text{Produkt, Zeit \# Produkt, Zeit, Durchschnitt} = \texttt{avg}(\text{Preis})}(\mathbf{R})$$

(i) liefert etwa $\frac{270+270+240}{3} = 260$ als Durchschnittspreis für 'Mountainbike', wobei sich der Durchschnitt auf die drei Tupel in R bezieht.

(iii) liefert den Durchschnittspreis pro Produkt und Zeiteinheit, also z.B. für 'Rennrad' im Zeitraum [18,20]: $\frac{270+280}{2} = 275$. — $\Gamma^{\text{Zeit}}_{\ldots}$ einschließlich Gruppierung nach Zeit entspricht also der Semantik sequentieller Aggregierungsanfragen aus Kapitel 4.

(ii) hingegen liefert $\frac{270*10+270*6+240*14}{10+6+14} = 256$ als Durchschnittspreis für 'Mountainbike', also bereits die in Kapitel 4 extra gewünschte, zeitlich gewichtete Durchschnittsbildung, weil $\Gamma^{\text{Zeit}}_{\ldots}$ ohne Gruppierung nach Zeit auch Informationen verschiedener Zeitpunkte aggregiert.

Beispielanfragen (Forts.) $\rightarrow \Gamma_{\ldots\text{avg}\ldots}$ mit ε

„Wie hoch ist der Durchschnittspreis pro Produkt pro Jahr?“

Angenommen die Zeiteinheiten im 'Zeit'-Attribut entsprechen fortlaufenden Monaten vom Januar 2000 an, also z.B.: „Das Produkt 'Mountainbike' hat im Zeitraum vom Januar 2000 (1) bis April 2001 (16=12+4) den Preis '270'“. In diesem Fall läßt sich R um ein Attribut 'Jahr' erweitern, welches das zum Zeitstempel gehörige Jahr angibt. Die Benutzung des erweiterten Operators $\varepsilon^{\text{Zeit}}_{\ldots}$ ist dabei nötig, damit ein Zeitstempel eindeutig einem Jahr zugeordnet werden kann; im „gepackten“ Zustand wäre dies hier nicht immer möglich (etwa bei $[11, 16]$).

$$RJ := \varepsilon^{\text{Zeit}}_{\text{Jahr}=\lceil \frac{\texttt{begin}(\text{Zeit})}{12} \rceil + 1999}(\mathbf{R})$$

Wieder sind zwei Formulierungen denkbar (mit oder ohne "Zeit"):

$$\Gamma^{[\text{Zeit}]}_{\text{Produkt, Jahr \# Produkt, Jahr, Durchschnitt} = \texttt{avg}(\text{Preis})}(RJ)$$

Für 'Mountainbike' im Jahr 2001 (Monate [13,24]) ergibt die mit $\Gamma^{\text{Zeit}}_{\ldots}$ formulierte Anfrage $\frac{270*4+240*8}{12} = 250$ und die Anfrage mit $\Gamma_{\ldots}$ $\frac{270+240}{2} = 255$.

„Wie lange (im Durchschnitt) wird ein Produkt von einem Hersteller angeboten?“

$$\Gamma_{\text{Hersteller, Produkt \# Hersteller, Produkt, Durchschnitt} = \texttt{avg}(\text{Dauer})}(\varepsilon_{\text{Dauer}=\texttt{length}(\text{Zeit})}(\mathbf{R}))$$

Optimierung

Vermeidung von `unpack`-Materialisierungen mit Hilfe von `split`
$\longrightarrow$ Zerlegung in nicht überlappende Teilintervalle

Für diesen Abschnitt seien X und Y Relationen. $V = v_1, \ldots, v_n$ sei eine Liste von Attributnamen intervallwertiger Attribute von X oder Y, und $\overline{V} := \text{attr}(S) - V$ sei die Menge aller Attribute von X, die nicht in V vorkommen.

$\texttt{split}^{v}_{P_X}(X)$ zerlegt die Tupel einer Ausgangsrelation X für ein intervallwertiges Attribut v anhand einer Punktmenge P_X. Ein Tupel mit dem v-Intervallwert $[b, e]$ wird durch einen Punkt $p \in P_X$ in zwei „Teile“ mit $[b, p-1]$ und $[p, e]$ gespalten, falls $b < p \leq e$ gilt. Ein Tupel $x \in X$ mit $x(l) = [b, e]$ wird anhand von

$$b < p_1 < \cdots < p_m \leq e \text{ für } p_1, \ldots, p_m \in P_X$$

demnach durch eine Menge von Tupeln mit folgenden v-Werten ersetzt:

$$[b, p_1 - 1], [p_1, p_2 - 1], \ldots, [p_{m-1}, p_m - 1], [p_m, e]$$

Analog zu $\texttt{unpack}^{V}(X)$ wird $\texttt{split}^{V}(X)$ durch die Hintereinanderausführung für alle Attribute in $V = v_1, \ldots, v_n$ definiert:

$$\texttt{split}^{V}_{P_X}(X) := \texttt{split}^{v_n}_{P_X}(\texttt{split}^{v_{n-1}}_{P_X}(\ldots \texttt{split}^{v_1}_{P_X}(X) \ldots))$$

Optimierung (Forts.)

Um zu erreichen, dass die v-Intervalle zweier Tupel aus X nur überlappen, falls sie gleich sind, d.h.

$$\forall x, \tilde{x} \in X : x(v) \,\texttt{overlaps}\, \tilde{x}(v) \Rightarrow x(v) = \tilde{x}(v)$$

ist $\texttt{split}^{v}_{P_X}(X)$ mit folgender Menge P_X (oder eine Obermenge) zu bilden:

$$P_X := \{\, \texttt{begin}(x(v)) : x \in X \wedge \exists\, \tilde{x} \in X \wedge x(v) \neq \tilde{x}(v) \wedge \texttt{begin}(x(v)) \in_{\subset} \tilde{x}(v) \,\}$$
$$\cup \{\, \texttt{end}(x(v))+1 : x \in X \wedge \exists\, \tilde{x} \in X \wedge x(v) \neq \tilde{x}(v) \wedge \texttt{end}(x(v))+1 \in_{\subset} \tilde{x}(v) \,\}$$

(sogar nur möglichst relevante Teilungspunkte; $\in_{\subset}$ steht für "liegt im Innern des Intervalls ..."; die Obermenge der `begin`- und `end`+1-Werte wäre leichter zu bilden)

Beispiel:

$X :=$

Name	VT
Müller	[2, 15]
Müller	[60, 68]
Schmidt	[3, 12]

$P_X = \{3, 13\}$ $\quad \texttt{split}^{\text{VT}}_{P_X}(X) =$

Name	VT
Müller	[2, 2]
Müller	[3, 12]
Müller	[13, 15]
Müller	[60, 68]
Schmidt	[3, 12]

Optimierung: Analyse einzelner Operatoren

$$\pi^V_A(X) = \texttt{pack}^V(\,\pi_A(\texttt{unpack}^V(X))\,) = \texttt{pack}^V(\,\pi_A(X)\,) \quad \text{– kein } \texttt{unpack} \text{ nötig}$$

$$\sigma^V_\varphi(X) = \texttt{pack}^V(\,\sigma_\varphi(\texttt{unpack}^V(X)) = \texttt{pack}^V(\,\sigma_\varphi(X)\,) \quad \text{falls } \varphi = \varphi|_{\overline{V}}$$

Auch $\texttt{pack}^V$ entfällt, falls X bereits entsprechend „gepackt“ ist.

Andernfalls läßt sich $\sigma^V_\varphi(X)$ so in einem einzigen Durchlauf über X erzeugen, dass die Materialisierung von Einheitsintervallen auf einzelne X-Tupel beschränkt bleibt.

$$\begin{aligned} X \bowtie^V Y &= \texttt{pack}^V(\,\texttt{unpack}^V(X) \bowtie \texttt{unpack}^V(Y)\,) \\ &= \texttt{pack}^V(\,\texttt{split}^V_{P_{X\cup Y}}(X) \bowtie \texttt{split}^V_{P_{X\cup Y}}(Y)\,) \end{aligned}$$

Die für `split` benötigte Punktmenge $P_{X\cup Y}$ ergibt sich hier aus den Intervallgrenzen der *beiden* Relationen X und Y. – Die Differenz $-^V$ wird ebenso behandelt.

Beispiel: Seien $X :=$

Name	VT
Meier	$[6, 13]$
Müller	$[10, 15]$
Müller	$[18, 20]$
Schmidt	$[21, 55]$
Müller	$[56, 70]$

und $Y :=$

Name	VT
Müller	$[10, 20]$
Müller	$[33, 44]$
Schmidt	$[22, 32]$
Schmidt	$[45, 50]$
Schulze	$[51, 55]$

.

Dann können die VT-Intervalle anhand der Punktmenge
$P_{X \cup Y} := \{10, 14, 16, 18, 22, 33, 45, 51\}$ zerlegt werden:

Name	VT
Meier	$[6, 9]$
Meier	$[10, 13]$
Müller	$[10, 13]$
Müller	$[14, 15]$
Müller	$[18, 20]$
Schmidt	$[21, 21]$
Schmidt	$[22, 32]$
Schmidt	$[33, 44]$
Schmidt	$[45, 50]$
Schmidt	$[51, 55]$
Müller	$[56, 70]$

$\bowtie$

Name	VT
Müller	$[10, 13]$
Müller	$[14, 15]$
Müller	$[16, 17]$
Müller	$[18, 20]$
Müller	$[33, 44]$
Schmidt	$[22, 32]$
Schmidt	$[45, 50]$
Schulze	$[51, 55]$

$=$

Name	VT
Müller	$[10, 13]$
Müller	$[14, 15]$
Müller	$[18, 20]$
Schmidt	$[22, 32]$
Schmidt	$[45, 50]$

Optimierung: Analyse einzelner Operatoren (Forts.)

Häufig ist es sinnvoll $\texttt{split}^V(X)$ nur für einzelne Partitionen ($\overline{V}$-Wertekombinationen) zu materialisieren und X so „Stück für Stück“ zu bearbeiten. Dieses Pipelining kann z.B. bei Differenz und Grupp.&Aggregierung angewandt werden.

Beispiel für $X -^V Y$*:* Sind X und Y wie im obigen Beispiel, dann gibt es verschiedene Werte für das Attribut 'Name', etwa 'Müller' und 'Schmidt'. Daher sind die folgenden Partitionen zu betrachten:

für X: $X_1 =$

Name	VT
Müller	[10, 15]
Müller	[18, 20]
Müller	[56, 70]

und $X_2 =$

Name	VT
Schmidt	[21, 55]

für Y: $Y_1 =$

Name	VT
Müller	[10, 20]
Müller	[33, 44]

und $Y_2 =$

Name	VT
Schmidt	[22, 32]
Schmidt	[45, 50]

$$P_{X_1 \cup Y_1} = \{16, 18\} \qquad P_{X_2 \cup Y_2} = \{22, 33, 45, 51\}.$$

X_1 (unverändert) $=$

Name	VT
Müller	$[10, 15]$
Müller	$[18, 20]$
Müller	$[56, 70]$

$\texttt{split}^{\text{VT}}_{P_{X_1 \cup Y_1}}(Y_1) =$

Name	VT
Müller	$[10, 15]$
Müller	$[16, 17]$
Müller	$[18, 20]$
Müller	$[33, 44]$

$\texttt{split}^{\text{VT}}_{P_{X_2 \cup Y_2}}(X_2) =$

Name	VT
Schmidt	$[21, 21]$
Schmidt	$[22, 32]$
Schmidt	$[33, 44]$
Schmidt	$[45, 50]$
Schmidt	$[51, 56]$

Y_2 (unverändert) $=$

Name	VT
Schmidt	$[22, 32]$
Schmidt	$[45, 50]$

Ergebnis: $X -^V Y =$

Name	VT
Müller	$[56, 70]$
Schmidt	$[21, 21]$
Schmidt	$[33, 44]$
Schmidt	$[51, 56]$

Optimierung: Weitere Maßnahmen

- `pack` kann in vielen Konstellationen semantisch entfallen.
- Die Berechnung von `pack` kann in einem Durchlauf unter Ausnutzung der Sortierung nach unteren Intervallgrenzen erfolgen.
- Die Punktmengen aller Intervallgrenzen einer Relation sollten redundant vorgehalten werden, um damit die jeweiligen `split`-Mengen zu berechnen. Sie lassen sich gut auch zur temporalen Indexierung verwenden.
- Implementierung in Oracle mittels „Table Functions“ für `unpack` und `pack` möglich; für `split` empfehlen sich Materialisierungen.
- Wenn o.g. Operatoren nicht *eingebaut* sind, wohl aber Intervalloperatoren, müssen Joins alternativ mit `overlaps` und `intersect` gebildet werden. `split` wird weiterhin mindestens für Differenz und Aggregierung benötigt. Auch `split` kann mittels `overlaps`-Verbund mit Zwischenintervallen zu vorbereiteten Punktmengen berechnet werden.
- auf jeden Fall benötigt: Datentyp und Indexstrukturen für Intervalle