2. Einführung der NF^2-Algebra

In diesem Kapitel wird das NF^2-Datenmodell und die NF^2-Algebra eingeführt. Die dabei verwendeten Definitionen stammen aus der Arbeit von Scholl [18]. Im ersten Abschnitt wird das neue Datenmodell vorgestellt, in dem nicht-atomare Attribute erlaubt sind. Danach werden basierend auf den Definitionen des NF^2-Datenmodells die Basisoperatoren der relationalen Algebra (abgekürzt als "1NF-Algebra") eingeführt. Dies ist möglich, da sich auch flache Relationen im NF^2-Datenmodell darstellen lassen. Da die NF^2-Algebra eine Erweiterung der 1NF-Algebra ist, wird diese im letzten Abschnitt dieses Kapitels vorgestellt.

2.1 NF^2-Datenmodell

Im flachen relationalen Datenmodell besteht eine Relation aus einer Menge von atomaren Attributen. Hingegen können im NF^2-Datenmodell Attribute selbst Relationen als Werte enthalten. Die formale Definition des NF^2-Datenmodells wird nun in diesem Abschnitt eingeführt. Dazu ist es sinnvoll, zuerst nochmals einen Blick auf das klassische relationale Datenmodell zu werfen, um die Unterschiede zum NF^2-Datenmodell zu verdeutlichen. Im 1NF-Modell besitzt eine Relation ein Schema und einen Wert (also die Menge der in der Relation enthaltenen Tupel). Das Schema wiederum besteht aus einer Menge von Attributen. Dabei werden Attribute durch ihren Namen und ihre Domäne charakterisiert. Im NF^2-Datenmodell kann ein Attribut selbst wieder ein Schema besitzen. Daher ist es für die kommenden Definitionen wichtig zwischen dem *Namen* und der *Beschreibung* eines Attributes zu unterscheiden. Die Beschreibung eines Attributes besteht aus seinem Namen und seinem Schema. Dies ist in Definition 2.1.1 festgehalten.

Definition 2.1.1 (Attributname, Schema und Beschreibung). *Gegeben sei $\mathcal{N}$ als eine unendliche Menge von Attributnamen. Die Menge $\mathcal{S}$ der gültigen Schemata und die Menge $\mathcal{B}$ der Beschreibungen sind rekursiv definiert durch folgende Implikationen.*

1. $n \in \mathcal{N} \Rightarrow \langle n, \emptyset \rangle \in \mathcal{B}$
2. $i = 1, ..., k : (b_i = \langle n_i, s_i \rangle \in \mathcal{B}) \wedge (i \neq j \Rightarrow n_i \neq n_j) \Rightarrow \{b_1, ..., b_k\} \in \mathcal{S}$
3. $n \in \mathcal{N} \wedge s \in \mathcal{S} \Rightarrow \langle n, s \rangle \in \mathcal{B}$

J. Hölsch, *Optimierung von Nested Queries unter Verwendung der NF²-Algebra,*
BestMasters, DOI 10.1007/978-3-658-12610-0_2, © Springer Fachmedien Wiesbaden 2016

Wie ersichtlich, besteht ein Schema aus einer Menge von Beschreibungen, wobei $\langle n, \emptyset \rangle$ die Beschreibung eines atomaren Attributes ist. Es sei angemerkt, dass zur Notation eines Tupels hier die Klammerung $\langle ... \rangle$ verwendet wird. In Beispiel 2.1.1 werden die eben eingeführten Begriffe nochmals verdeutlicht.

Beispiel 2.1.1. Angenommen es sei eine modifizierte Version der TPC-H PartSupp Relation gegeben, bei der zu jedem Teil die entsprechenden Lieferanten in einem nicht-atomaren Attribut "supplier" gespeichert werden (der Kürze halber ist in der Tabelle nur ein Beispieltupel enthalten).

ps_partkey	supplier	
	s_suppkey	s_name
1	9	Supplier#9
	15	Supplier#15

Die Beschreibungen der Attribute sehen wie folgt aus:

$$\langle \text{ps\_partkey}, \emptyset \rangle, \langle \text{supplier}, \{ \langle \text{s\_suppkey}, \emptyset \rangle, \langle \text{s\_name}, \emptyset \rangle \} \rangle.$$

◇

In der nächsten Definition werden Funktionen eingeführt, mit denen auf die einzelnen Komponenten einer Beschreibung zugegriffen werden kann.

Definition 2.1.2.

1. $nam : \mathcal{B} \to \mathcal{N}$
   $nam(\langle n, s \rangle) = n$
2. $sch : \mathcal{B} \to \mathcal{S}$
   $sch(\langle n, s \rangle) = s$
3. $attr : \mathcal{B} \to \mathcal{P}(\mathcal{N})$
   $attr(\langle n, s \rangle) = \{n_i | \langle n_i, s_i \rangle \in s\}$

Für eine gegebene Beschreibung liefert die Funktion *nam* den Namen und die Funktion *sch* das Schema zurück. Des Weiteren gibt *attr* die im Schema enthaltenen Attributnamen aus. Bei Beschreibungen von atomaren Attributen liefern die Funktionen *sch* und *attr* die leere Menge zurück. Zusätzlich sei eine Funktion d gegeben, die für einen Attributnamen n_i die dazugehörige Beschreibung b_i zurückliefert. Somit gilt

$$d(n_i) = b_i \Leftrightarrow nam(b_i) = n_i.$$

Beispiel 2.1.2. Es sei die folgende Beschreibung des Attributs "supplier" gegeben:

$$b = \langle \text{supplier}, \{\langle \text{s\_suppkey}, \emptyset\rangle, \langle \text{s\_name}, \emptyset\rangle\}\rangle.$$

Die Funktionen aus Definition 2.1.2 liefern folgende Werte zurück:

- $nam(b) = \text{supplier}$,
- $sch(b) = \{\langle \text{s\_suppkey}, \emptyset\rangle, \langle \text{s\_name}, \emptyset\rangle\}$,
- $attr(b) = \{\text{s\_suppkey}, \text{s\_name}\}$.

◇

Als nächstes stellt sich die Frage, wie der Domain eines nicht-atomaren Attributes definiert wird. Dazu wird Funktion 2.1 eingeführt, die jedem Attributnamen $n_i \in attr(b)$ einer Beschreibung b einen Wert aus der zum Attribut dazugehörigen Domain $dom(d(n_i))$ zuordnet (jedoch ignorieren wir hier für den Moment die genaue Definition der Domain $dom(d(n_i))$, die später noch dargestellt wird).

$$\begin{aligned} f : attr(b) &\rightarrow \bigcup_{b_i \in sch(b)} dom(b_i) \\ f : n_i &\mapsto d_i \in dom(d(n_i)) \end{aligned} \tag{2.1}$$

Somit lässt sich durch eine Funktion ein bestimmter Tupel darstellen, wodurch der Wert einer Relation einer Menge von Funktionen entspricht. Alle möglichen Funktionen werden durch die Menge F beschrieben, wobei der konkrete Wert einer Relation eine Teilmenge von F darstellt.

$$F = \left\{ f : attr(b) \rightarrow \bigcup_{b_i \in sch(b)} dom(b_i) \middle| \forall_{n_i \in attr(b)} : f(n_i) \in dom(d(n_i)) \right\}$$

Daher ist der Domain eines gegebenen nicht-atomaren Attributes mit der Beschreibung b die Potenzmenge von F. Einem atomaren Attribut wird hingegen ein gegebener atomarer Domain D_i zugeordnet. In Definition 2.1.3 ist dies festgehalten.

Definition 2.1.3 (Domain). *Gegeben sei eine Menge $\mathcal{D} = \{D_1, ..., D_d\}$ von atomaren Domains, sowie eine Funktion $\delta : \mathcal{B} \rightarrow \mathcal{D}$, die jedem atomaren Attribut eine atomare Domain zuordnet. Der Domain einer Beschreibung b ist dann wie folgt definiert:*

$$dom(b) = \begin{cases} \delta(b) & \text{falls } sch(b) = \emptyset \\ \mathcal{P}(F) & \text{sonst} \end{cases}.$$

Wie es in Definition 2.1.3 zu sehen ist, wird einem atomaren Attribut durch die Funktion δ der dazugehörige atomare Domain zugeordnet. Bei einem nicht-atomaren Attribut ist der Domain wie bereits beschrieben die Potenzmenge von F. Somit wird für ein nicht-atomares Attribut die Funktion *dom* rekursiv angewendet, um den Domain des Attributes zu erhalten (da *dom* wiederum selbst in der Definition von F vorhanden ist).

Beispiel 2.1.3. Angenommen der Tupel aus Beispiel 2.1.1 sei repräsentiert durch die Funktion f_{t_1}, so lässt sich der Wert des Attributes "ps_partkey" wie folgt ermitteln:

$$f_{t_1}(\text{ps\_partkey}) = 1.$$

◇

Als nächstes kann die Definition einer Relation in der NF²-Algebra eingeführt werden. Diese besteht aus ihrer Beschreibung und besitzt einen Wert aus ihrem Domain.

Definition 2.1.4 (($\mathbf{NF^2}$-)Relation). *Es sei eine Beschreibung b mit $sch(b) \neq \emptyset$ und eine Funktion $\delta : \mathcal{B} \to \mathcal{D}$, die einem atomaren Attribut einen atomaren Domain zuordnet, gegeben. Des Weiteren sei $v \in dom(b)$ gegeben. Somit ist $R = \langle b, v \rangle$ eine (NF²-)Relation unter δ.*

Um auf die einzelnen Komponenten einer Relation $R = \langle b, v \rangle$ zugreifen zu können, werden die Funktionen:

$$desc(R) := b \quad \text{und}$$
$$val(R) := v$$

eingeführt, wobei *desc* die Beschreibung und *val* den Wert einer Relation zurückliefert. Zur einfacheren Lesbarkeit werden die Abkürzungen verwendet:

$$nam(R) := nam(desc(R)),$$
$$sch(R) := sch(desc(R)),$$
$$attr(R) := attr(desc(R)).$$

Eine Relation ist in der ersten Normalform, wenn für alle $b \in sch(R)$ gilt, dass das Schema von b der leeren Menge entspricht. Als zusätzliche Vereinfachung wird nicht weiter zwischen unterschiedlichen atomaren Domains unterschieden, d.h. nur die unterschiedlichen Schemata sind von Relevanz. Somit wird die ursprünglich definierte Menge der atomaren Domains $\mathcal{D} = \{D_1, ..., D_d\}$ ersetzt durch $\mathcal{D} = \{D\}$. Daher kann im Folgenden auch die Funktion δ vernachlässigt werden und für eine atomare Beschreibung b gilt nun $dom(b) = D$.

Als nächstes wird noch der Begriff "Tupel" in Definition 2.1.5 formal festgehalten. Wie bereits zuvor erwähnt, entspricht ein Tupel einer Funktion.

Definition 2.1.5 (Tupel). *Die Elemente $t \in val(R)$ einer Relation R heißen Tupel. Dabei ist $t(A)$ der Attributwert von $A \in attr(R)$ in Tupel t.*

Für ein Tupel $t \in val(R)$ und ein Attributname $A \in attr(R)$ ist $\langle d(A), t(A)\rangle$ selbst wieder eine Relation, genau dann wenn $sch(d(A)) \neq \emptyset$ gilt. Zusätzlich wird in Definition 2.1.6 ein sogenanntes *NF²-Objekt* eingeführt. Ein NF²-Objekt besteht ebenfalls wie eine NF²-Relation aus einer Beschreibung und einem Wert. Im Vergleich zur NF²-Relation kann das Schema eines NF²-Objektes der leeren Menge entsprechen. Somit ist ein NF²-Objekt mit der Beschreibung b ein atomares Attribut, wenn $sch(b) = \emptyset$ gilt. Gilt hingegen $sch(b) \neq \emptyset$ ist das NF²-Objekt eine Relation.

Definition 2.1.6 (NF²-Objekt). *Ein NF²-Objekt $\langle b, v\rangle$ besteht aus einer Beschreibung $b \in \mathcal{B}$ und einem Wert $v \in dom(b)$.*

Zuletzt muss noch der Begriff der *Umgebung* eingeführt werden. Eine Umgebung ε ist eine Menge von NF²-Objekten. Zum Beispiel besteht die Umgebung einer Datenbank aus einer Menge von Relationen. Des Weiteren sei $\delta = \{b | \langle b, v\rangle \in \varepsilon\}$ als Menge der Beschreibungen der Umgebung ε gegeben.

2.2 1NF-Algebra

Vor Einführung der NF²-Algebra wird in diesem Abschnitt zuerst die 1NF-Algebra, basierend auf den vorherigen Definitionen, vorgestellt. Dies ist dadurch begründet, dass die NF²-Algebra eine Erweiterung der 1NF-Algebra ist. Durch die folgenden Punkte wird rekursiv definiert, was gültige algebraische Ausdrücke der 1NF-Algebra sind.

1. Eine Relation R aus ε ist ein gültiger algebraischer Ausdruck in ε.

2. Sind E_1 und E_2 gültige algebraische Ausdrücke in ε, so sind auch $E_1 \times E_2$, $E_1 \cup E_2$ und $E_1 \setminus E_2$ gültige algebraische Ausdrücke in ε.

3. Ist E ein gültiger algebraischer Ausdruck in ε, so sind auch $\pi[L](E)$ und $\sigma[F](E)$ gültige algebraische Ausdrücke in ε.

Alle weiteren relationalen Operatoren, wie z.B. der Join, lassen sich mit Hilfe der eben genannten Operatoren darstellen. In diesem Abschnitt werden nun diese "Basisoperatoren" eingeführt und darauf eingegangen, wie jeweils das Ergebnisschema und die Ergebnismenge definiert ist. Als erstes werden dazu die Mengenoperationen *Vereinigung* und *Differenz* vorgestellt.

Definition 2.2.1 (Vereinigung). *Angenommen E_1 und E_2 sind gültige algebraische Ausdrücke, die das selbe Schema besitzen (also $sch(E_1) = sch(E_2)$ gilt), so ist die Vereinigung $E_1 \cup E_2$ definiert durch:*

$$sch(E_1 \cup E_2) := sch(E_1)(= sch(E_2)),$$
$$val(E_1 \cup E_2) := \{t | t \in val(E_1) \vee t \in val(E_2)\}.$$

Definition 2.2.2 (**Differenz**). *Angenommen E_1 und E_2 sind gültige algebraische Ausdrücke, die das selbe Schema besitzen (also $sch(E_1) = sch(E_2)$ gilt), so ist die Differenz $E_1 \setminus E_2$ definiert durch:*

$$sch(E_1 \setminus E_2) := sch(E_1)(= sch(E_2)),$$
$$val(E_1 \setminus E_2) := \{t | t \in val(E_1) \wedge t \notin val(E_2)\}.$$

In der nächsten Definition wird das *relationale Produkt* eingeführt, bei dem gefordert wird, dass die Mengen der Attributnamen der beteiligten algebraischen Ausdrücke disjunkt sind. Da ein Tupel durch eine Funktion repräsentiert wird, entsteht durch die Konkatenation zweier Tupel eine neue Funktion, die in Definition 2.2.3 dargestellt wird.

Definition 2.2.3. *Für zwei gegebene Funktionen $f : A \to B$ und $g : C \to D$ ist die Funktion $f \oplus g$ definiert durch:*

$$f \oplus g : A \cup C \to B \cup D,$$
$$f \oplus g(x) = \begin{cases} f(x) & falls\ x \in A \\ g(x) & falls\ x \in C \end{cases}.$$

Ist der Input der Funktion $f \oplus g$ ein Attributname des von f repräsentierten Tupels, so wird der dazugehörige Wert von f zurückgeliefert. Ist der Input ein Attributname von g, so wird entsprechend der Wert von g zurückgegeben. Daraus wird auch ersichtlich, warum die Disjunktheit zwischen den Mengen der Attributnamen gefordert wird. Mit dieser eingeführten Funktion kann nun das relationale Produkt definiert werden.

Definition 2.2.4 (**Relationales Produkt**). *Das relationale Produkt $E_1 \times E_2$ zweier gültiger algebraischer Ausdrücke E_1 und E_2 mit $attr(E_1) \cap attr(E_2) = \emptyset$ ist definiert durch:*

$$sch(E_1 \times E_2) := sch(E_1) \cup sch(E_2),$$
$$val(E_1 \times E_2) := \{t_1 \oplus t_2 | t_1 \in val(E_1) \wedge t_2 \in val(E_2)\}.$$

Als nächstes wird die *Projektion* eingeführt. Mit der Projektion lassen sich bestimmte Attribute einer Input-Relation herausfiltern (die Menge der Attribute, die beibehalten werden soll, wird als *Projektionsliste* bezeichnet). Somit ändert der Projektionsoperator das Schema der Input-Relation.

Definition 2.2.5 (**Projektion**). *Angenommen es sei ein gültiger algebraischer Ausdruck E und eine Projektionsliste $L \subseteq attr(E)$ gegeben, so ist die Projektion $\pi[L](E)$ definiert durch:*

$$sch(\pi[L](E)) := \{d(n) | n \in L\},$$
$$val(\pi[L](E)) := \{t(L) | t \in val(E)\}.$$

Das resultierende Schema besteht aus den Beschreibungen, die zu den Attributnamen aus L gehören. Die Ergebnismenge enthält die Tupel von E mit ausschließlich den Attributen,

die in L vorkommen. Obwohl in Definition 2.1.5 die Anwendung der Funktion $t \in val(E)$ nur auf einem einzelnen Attributnamen definiert ist, erhält hier zur kürzeren Darstellung t als Input eine Menge L, wobei $t(L) = \{t(A)|A \in L\}$ gilt.

Als letztes wird in diesem Abschnitt die *Selektion* eingeführt. Mit der Selektion lassen sich Tupel einer Input-Relation auswählen, die eine bestimmte Bedingung erfüllen. Ist $x_1, x_2 \in \mathcal{N} \cup D$ und $\theta \in \{<, \leq, >, \geq, =, \neq\}$ gegeben, so ist eine gültige Bedingung wie im Folgenden definiert.

1. $x_1 \theta x_2$ ist eine gültige Bedingung.
2. Sind F_1 und F_2 gültige Bedingungen, so sind auch $F_1 \wedge F_2$, $F_1 \vee F_2$ und $\neg F_1$ gültige Bedingungen.

Wie zu sehen ist, kann eine gültige Bedingung F Attributnamen enthalten. Bei der Ausführung der Selektion auf eine Input-Relation wird für jeden Tupel ein konkreter Wert für den Attributnamen eingesetzt und die Bedingung nach *true* oder *false* ausgewertet. $F(t)$ beschreibt eine solche Funktion, die als Input einen Tupel t erhält und für die Attributnamen die entsprechenden Werte des Tupels einsetzt und die Bedingung auswertet. Mit dieser Funktion kann nun die Selektion definiert werden.

Definition 2.2.6 (Selektion). *Angenommen es sei ein gültiger algebraischer Ausdruck E und eine gültige Bedingung F gegeben, bei der für alle enthaltenen Attributnamen $n \in attr(E)$ gilt, so ist die Projektion $\sigma[F](E)$ definiert durch:*

$$sch(\sigma[F](E)) := sch(E),$$
$$val(\sigma[F](E)) := \{t \in val(E)|F(t)\}.$$

Bei der Selektion wird also das Schema nicht verändert, sondern nur diejenigen Tupel ausgewählt, die die Bedingung F erfüllen.

2.3 NF^2-Algebra

In diesem Abschnitt wird die NF^2-Algebra vorgestellt. Wie bereits erwähnt, stellt sie eine Erweiterung der 1NF-Algebra dar. Es wird der Selektions- und Projektionsoperator so erweitert, dass an den Stellen, wo in der 1NF-Algebra Attribute auftreten können (z.B. in der Selektionsbedinung), nun auch NF^2-Ausdrücke stehen können. So wird zum Beispiel die Anwendung von Selektionen bzw. Projektionen auf nicht-atomaren Attributen möglich. Neben diesen Erweiterungen existieren noch zwei neue Operatoren namens *Nest* und *Unnest*, mit denen aus einer flachen eine geschachtelte Relation (und umgekehrt) erzeugt werden kann.

Da ein NF^2-Ausdruck (z.B. in einer Projektionsliste) ein neues nicht-atomares Attribut erzeugen kann, muss zuerst die sogenannte *algebraische Zuweisung* eingeführt werden, mit der einem NF^2-Ausdruck ein Name zugewiesen werden kann.

Definition 2.3.1 (**Algebraische Zuweisung**). *Angenommen es sei ein gültiger algebraischer Ausdruck E und ein Name $N \in \mathcal{N}$ gegeben, so ist der folgende Ausdruck eine algebraische Zuweisung:*

$$N := E.$$

N ist selbst wieder eine NF²-Relation, die charakterisiert ist durch:

$$\begin{aligned} nam(N) &:= N, \\ sch(N) &:= sch(E), \\ val(N) &:= val(E). \end{aligned}$$

Die algebraische Zuweisung ermöglicht daher auch das Umbenennen von Attributen. Nachdem die algebraische Zuweisung eingeführt wurde, kann nun die *verschachtelte Projektion* definiert werden. Bei der verschachtelten Projektion können in der Projektionsliste wieder NF²-Ausdrücke stehen. Damit lassen sich zum Beispiel bestimmte Attribute eines nicht-atomaren Attributs auswählen. Besondere Aufmerksamkeit bei der Definition der verschachtelten Projektion (bzw. später auch bei der verschachtelten Selektion) bekommt die Umgebung. Wie bereits erwähnt, beschreibt die Umgebung die ansprechbaren NF²-Objekte. Im Fall von flachen Operatoren bei der 1NF-Algebra entspricht die Umgebung der Datenbank (also einer Menge von Relationen). Hingegen ist bei der NF²-Algebra die Umgebung abhängig von der Schachtelungstiefe innerhalb des entsprechenden NF²-Ausdruckes, d.h. die im Ausdruck angesprochenen Operanden müssen in der dazugehörigen Umgebung vorhanden sein. Bei den folgenden Definitionen wird zusätzlich bei der *val*-Funktion eine Umgebung ε und bei der *sch*-Funktion die Menge δ der in ε enthaltenen Beschreibungen mit angegeben.

Definition 2.3.2 (**Verschachtelte Projektion**). *Angenommen es sei ein gültiger algebraischer Ausdruck E in der Umgebung ε mit der Umgebungsbeschreibung δ und $L = \{N_i := E_i | i = 1, ..., n\}$ eine Menge von Zuweisungen für ein n gegeben, so ist die verschachtelte Projektion $N := \pi[L](E)$ wie folgt definiert:*

$$\begin{aligned} sch_\delta(N := \pi[L](E)) := \{\langle N_i, sch_{\delta'}(E_i)\rangle | i = 1, ..., n\} \\ mit\ \delta' = \delta \cup \{d(A) | A \in sch_\delta(E)\}, \end{aligned}$$

$$\begin{aligned} val_\varepsilon(N := \pi[L](E)) := \{t | t(N_i) = val_{\varepsilon'}(E_i) \wedge i = 1, ..., n \wedge \\ \varepsilon' = \varepsilon \cup \{\langle d(A), t'(A)\rangle | A \in sch_\delta(E)\} \wedge t' \in val_\varepsilon(E)\}. \end{aligned}$$

Zuletzt wird noch die Selektion erweitert, so dass auch NF²-Ausdrücke in der Selektionsbedingung stehen können. Dazu muss die ursprüngliche Definition einer gültigen Bedingung, wie sie in Abschnitt 2.2 eingeführt wurde, so abgeändert werden, dass überall wo ein Attribut erlaubt ist, nun ein beliebiges NF²-Objekt stehen kann. In Abschnitt 2.2 wurde eine Bedingung F als eine Funktion $F(t)$ betrachtet, die für einen gegebenen Tupel t die Bedingung F nach *true* oder *false* auswertet. Bei der Definition der verschachtelten Selektion wird zusätzlich die Umgebung mit einbezogen, so dass $F(t)$ zu $F_\varepsilon(t)$ abgeändert wird. Für F_ε wird gefordert, dass alle darin angesprochenen Objekte in der Umgebung ε vorkommen.

Definition 2.3.3 (**Verschachtelte Selektion**). *Angenommen es sei ein gültiger algebraischer Ausdruck E in der Umgebung ε mit der Umgebungsbeschreibung δ und einer gültigen Bedingung F gegeben, so ist die verschachtelte Selektion $\sigma[F](E)$ wie folgt definiert:*

$$sch_\delta(\sigma[F](E)) := sch_\delta(E),$$
$$val_\varepsilon(\sigma[F](E)) := \{t \in val_\varepsilon(E) | F_{\varepsilon'}(t) \wedge \varepsilon' = \varepsilon \cup \{\langle d(A), t(A)\rangle | A \in sch_\delta(E)\}\}.$$

Neben der Erweiterung vorhandener Operatoren werden in der NF^2-Algebra noch zwei neue Operatoren eingeführt. Dies ist zum einen der Nest-Operator (dargestellt mit ν) und zum anderen der Unnest-Operator (dargestellt mit μ). Der Nest-Operator $\nu[A := (A_1, ..., A_n)](E)$ erzeugt ein neues nicht-atomares Attribut A, welches als Subschema die Attribute $A_1, ..., A_n \in attr(E)$ besitzt. Die ursprüngliche Ergebnis-Relation des algebraischen Ausdruckes E wird nach den Attributen $attr(E) \setminus \{A_1, ..., A_n\}$ gruppiert und die zu einer Gruppe dazugehörigen Subtupel mit den Attributen $A_1, ..., A_n$ als Subrelation im Attribut A festgehalten. Somit kann aus einer flachen eine verschachtelte Relation erzeugt werden. Im Folgenden wird der Nest-Operator formal definiert.

Definition 2.3.4 (**Nest-Operator**). *Angenommen es sei ein gültiger algebraischer Ausdruck E mit $X = \{A_1, ..., A_n\} \subseteq attr(E)$, $Y = attr(E) \setminus X$ und $A \in \mathcal{N}$ mit $A \notin Y$ gegeben, so ist der Nest-Operator $\nu[A := (X)](E)$ definiert durch:*

$$sch(\nu[A := (X)](E)) := sch(E) \setminus d(X) \cup \{\langle A, d(X)\rangle\},$$
$$val(\nu[A := (X)](E)) :=$$
$$\{t | t(Y) \in \pi[Y](E) \wedge t(A) = \{t'(X) | t' \in val(E) \wedge t'(Y) = t(Y)\}\}.$$

Beispiel 2.3.1. Angenommen es sei die Relation "Part" gegeben (siehe Tabelle unten links). Der Kürze halber ist nur eine Teilmenge der Attribute enthalten. Mit Part' := ν[p_part := (p_partkey, p_name)](Part) lässt sich die Relation Part nach dem Attribut "p_type" gruppieren (für das Ergebnis, siehe Tabelle unten rechts).

p_partkey	p_name	p_type
1	Part#1	TypeA
2	Part#2	TypeA
3	Part#3	TypeB
4	Part#4	TypeC
5	Part#5	TypeC
6	Part#6	TypeC

p_type	p_part	
	p_partkey	p_name
TypeA	1	Part#1
	2	Part#2
TypeB	3	Part#3
TypeC	4	Part#4
	5	Part#5
	6	Part#6

◇

Der Unnest-Operator $\mu[A](E)$ stellt die inverse Operation zum Nest-Operator dar, wobei A ein nicht-atomares Attribut von E ist, welches als Schema die Attribute $A_1, ..., A_n$ besitzt. Durch Anwendung des Ausdruckes $\mu[A](R)$ wird das Attribut A aus E entfernt und die Attribute $A_1, ..., A_n$ als neue Top-Level Attribute zu E hinzugefügt, wobei für jeden äußeren Tupel von E ein Kreuzprodukt mit den dazugehörigen Tupel der Subrelation

von A gebildet wird. Somit lässt sich aus einer verschachtelten wieder eine flache Relation erzeugen. Auch der Unnest-Operator wird im Folgenden formal definiert.

Definition 2.3.5 (Unnest-Operator). *Angenommen es sei ein gültiger algebraischer Ausdruck E und ein nicht-atomares Attribut $A \in attr(E)$ sowie $Y = attr(E) \setminus \{A\}$ mit $attr(A) \cap Y = \emptyset$ gegeben, so ist der Unnest-Operator $\mu[A](E)$ definiert durch:*

$$sch(\mu[A](E)) := sch(E) \setminus d(A) \cup sch(A),$$
$$val(\mu[A](E)) := \{t | t(Y) = t'(Y) \wedge t(attr(A)) \in t'(A) \wedge t' \in val(E)\}.$$

Mit μ[p_part](Part') lässt sich die verschachtelte Relation aus Beispiel 2.3.1 wieder in die ursprüngliche flache Relation transformieren.

http://www.springer.com/978-3-658-12609-4

Optimierung von Nested Queries unter Verwendung der NF2-Algebra

Hölsch, J.

2016, X, 83 S., Softcover
ISBN: 978-3-658-12609-4