WWW.PEARSON-STUDIUM.DE
CWS
COMPANION WEBSITE

it
informatik

Ramez A. Elmasri
Shamkant B. Navathe

Grundlagen von Datenbanksystemen

Bachelorausgabe

3., aktualisierte Auflage

ADDISON-WESLEY

PEARSON
Studium

Abbildung 5.16: Beispiel einer AGGREGATE-FUNCTION-Operation:
(a) $R_{(ABT,\ ANZAHL\_ANGESTELLTE,\ GEHALT\_DURCHSCHNITT)} \leftarrow {}_{ANR}\Im_{COUNT\ SSN,\ AVERAGE\ GEHALT}(ANGESTELLTER)$;
(b) ${}_{ANR}\Im_{COUNT\ SSN,\ AVERAGE\ GEHALT}(ANGESTELLTER)$;
(c) $\Im_{COUNT\ SSN,\ AVERAGE\ GEHALT}(ANGESTELLTER)$.

(a)

R	ANR	ANZAHL_ANGESTELLTE	GEHALT_DURCHSCHNITT
	5	4	33250
	4	3	31000
	1	1	55000

(b)

ANR	COUNT_SSN	GEHALT_DURCHSCHNITT
5	4	33250
4	3	31000
1	1	55000

(c)

COUNT_SSN	GEHALT_DURCHSCHNITT
8	35125

Abbildung 5.16(b) zeigt beispielsweise das Ergebnis der folgenden Operation:

$${}_{ABT}\Im_{COUNT\ SSN,\ AVERAGE\ GEHALT}(ANGESTELLTER)$$

Werden keine Gruppierungsattribute spezifiziert, werden die Funktionen auf die Attributwerte *aller Tupel* der Relation angewandt, so dass die resultierende Relation *nur ein einziges Tupel* beinhaltet. Abbildung 5.16(c) zeigt z.B. das Ergebnis der folgenden Operation:

$$\Im_{COUNT\ SSN,\ AVERAGE\ GEHALT}(ANGESTELLTER)$$

Wichtig ist die Feststellung, dass Duplikate vor der Berechnung der Aggregatwerte im Allgemeinen *nicht entfernt* werden. Auf diese Weise wird die normale Interpretation von Funktionen wie SUM und AVERAGE berechnet.[12] Das Ergebnis der Berechnung einer Aggregatfunktion ist eine Relation und keine Skalarzahl, auch wenn diese aus nur einem einzigen Wert besteht.

5.5.2 Rekursive Abschlussoperationen

Eine weitere Operation, die im Allgemeinen nicht mit der grundlegenden relationalen Algebra definiert werden kann, ist die **rekursive Hülle** (Recursive Closure). Diese Operation wird auf eine **rekursive Beziehung** zwischen Tupeln gleichen Typs angewandt, z.B. die Beziehung zwischen einem Angestellten und einem Vorgesetzten. Diese Beziehung wird durch den Fremdschlüssel SUPERSSN der Relation ANGESTELLTER in

12. In SQL können Duplikate vor der Anwendung der Aggregatfunktion entfernt werden, indem man das Schlüsselwort DISTINCT einbindet (siehe Kapitel 6).

den Abbildungen 5.6 und 5.7 beschrieben, der jedes Angestellten-Tupel (in der Rolle des Untergebenen) mit einem anderen Angestellten-Tupel (in der Rolle des Vorgesetzten) in Beziehung stellt. Ein Beispiel einer rekursiven Operation ist der Abruf aller Untergebenen eines Angestellten e auf allen Ebenen, d.h. alle direkt von e beaufsichtigten Angestellten e'; alle direkt von jedem Angestellten e' beaufsichtigten Angestellten e''; alle direkt von jedem Angestellten e'' beaufsichtigten Angestellten e''' usw. Obwohl es nicht schwierig ist, in der relationalen Algebra alle von e beaufsichtigten Angestellten *auf einer bestimmten Ebene* zu spezifizieren, ist es nicht möglich, alle Untergebenen *auf allen Ebenen* zu spezifizieren, falls die Anzahl der Ebenen nicht bekannt ist. Um beispielsweise die SSN aller Angestellten e' zu spezifizieren, die auf der *ersten Ebene* direkt von dem Angestellten e namens 'James Borg' (siehe Abbildung 5.6) beaufsichtigt werden, können wir folgende Operation schreiben:

$$\text{BORG\_SSN} \leftarrow \pi_{\text{SSN}}(\sigma_{\text{VNAME='James' AND NNAME='Borg'}}(\text{ANGESTELLTER}))$$

$$\text{AUFSICHT}(\text{SSN1}, \text{SSN2}) \leftarrow \pi_{\text{SSN, SUPERSSN}}(\text{ANGESTELLTER})$$

$$\text{RESULTAT1}(\text{SSN}) \leftarrow \pi_{\text{SSN1}}(\text{AUFSICHT} \bowtie_{\text{SSN2=SSN}} \text{BORG\_SSN})$$

Um alle Angestellten abzurufen, die von Borg auf Ebene 2 beaufsichtigt werden, d.h. alle Angestellten e'', die von einem Angestellten e' beaufsichtigt werden, der wiederum direkt von Borg beaufsichtigt wird, können wir einen weiteren JOIN auf das Resultat der ersten Anfrage wie folgt anwenden:

$$\text{RESULTAT2}(\text{SSN}) \leftarrow \pi_{\text{SSN1}}(\text{AUFSICHT} \bowtie_{\text{SSN2=SSN}} \text{RESULTAT1})$$

Um beide Angestelltenmengen abzurufen, die auf Ebene 1 und 2 von 'James Borg' beaufsichtigt werden, können wir die UNION-Operation auf die beiden Ergebnisse wie folgt anwenden:

$$\text{RESULTAT} \leftarrow \text{RESULTAT2} \cup \text{RESULTAT1}$$

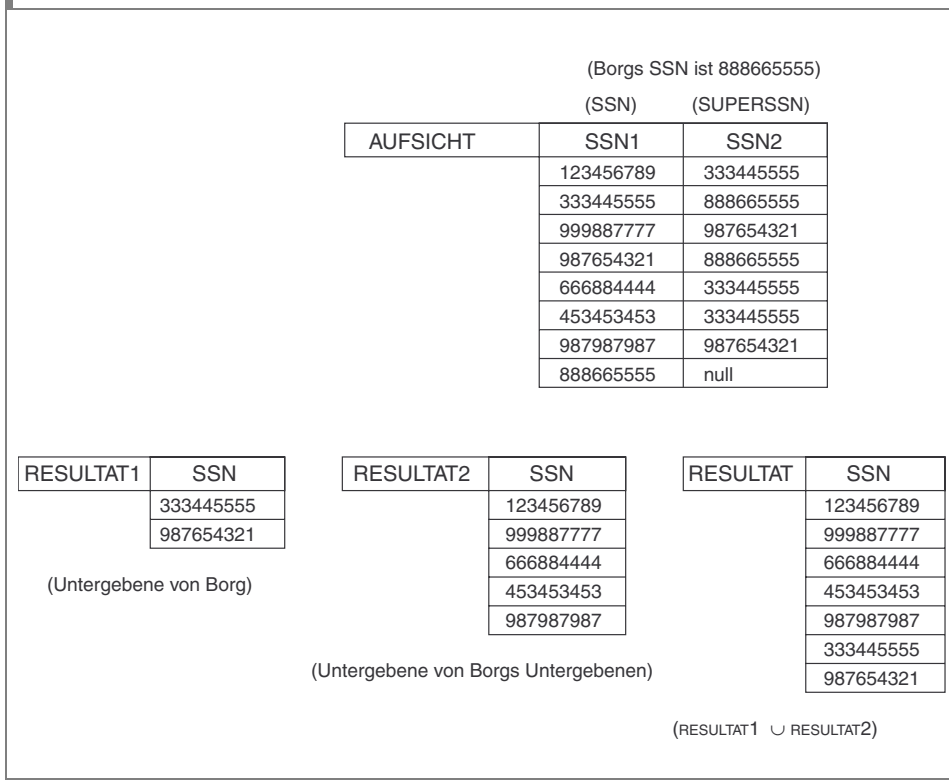
Die Ergebnisse dieser Anfragen sind in Abbildung 5.17 dargestellt. Obwohl es möglich ist, Angestellte auf jeder Ebene abzurufen und dann ihre UNION zu ermitteln, können im Allgemeinen keine Anfragen wie »Suche nach allen Untergebenen von 'James Borg' auf allen existierenden Ebenen« formuliert werden, ohne einen Schleifenmechanismus[13] zu verwenden. Eine Operation namens *transitiver Hülle* (Transitive Closure) von Relationen wurde vorgeschlagen, um die rekursive Beziehung zu berechnen.

5.5.3 OUTER-JOIN- und OUTER-UNION-Operationen

In diesem Abschnitt werden Erweiterungen der JOIN- und UNION-Operationen beschrieben. Die weiter oben beschriebenen JOIN-Operationen stimmen Tupel miteinander ab, die die JOIN-Bedingung erfüllen. Für eine NATURAL-JOIN-Operation R * S erscheinen z.B. nur Tupel aus R im Resultat, die passende Tupel in S haben – und umgekehrt. Folglich werden Tupel ohne *passendes* (oder *zusammenhängendes*) Tupel aus dem JOIN-Resultat eliminiert. Tupel mit Null-Werten in den JOIN-Attributen werden ebenfalls eliminiert. Eine Menge von Operation namens OUTER JOIN kann benutzt

13. Rekursive Anfragen werden in Kapitel 25 der ungekürzten Ausgabe in Zusammenhang mit einer Übersicht deduktiver Datenbanken wieder aufgegriffen. Der SQL3-Standard beinhaltet eine Syntax für den rekursiven Abschluss, der SQL2-Standard nicht.

Abbildung 5.17: Beispiel einer rekursiven Anfrage über zwei Ebenen.



werden, wenn alle Tupel in R oder alle in S oder alle in beiden Relationen im Resultat des JOIN erscheinen sollen, gleichgültig, ob sie in der jeweils anderen Relation passende Tupel haben. Dies erfüllt die Anforderung für Anfragen, bei denen Tupel aus zwei Tabellen durch eine JOIN-Bedingung kombiniert werden sollen, einige Tupel aber mangels passender Werte verloren gehen. In solchen Fällen ist eine Operation wünschenswert, die alle Tupel erzeugt, ungeachtet der Tatsache, ob sie einen »JOIN-Partner« besitzen oder nicht.

Nehmen wir beispielsweise an, wir benötigen eine Liste aller Angestelltennamen und der Namen der Abteilungen, die sie leiten, *falls sie eine Abteilung leiten*. Wir können eine Operation **LEFT OUTER JOIN** anwenden, die durch das Symbol ⟕ bezeichnet wird, um das Ergebnis wie folgt zu berechnen:

$$\text{TEMP} \leftarrow (\text{ANGESTELLTER} \text{ ⟕}_{\text{SSN=MGRSSN}} \text{ABTEILUNG})$$

$$\text{RESULTAT} \leftarrow \pi_{\text{VNAME, INITIAL, NNAME, ANAME}}(\text{TEMP})$$

Die LEFT-OUTER-JOIN-Operation gibt jedes Tupel der *ersten* bzw. *linken* Relation R in R ⟕ S aus. Wird kein passendes Tupel in S gefunden, werden die Attribute von S im JOIN-Resultat mit Nullwerten »aufgefüllt«. Das Ergebnis dieser Operationen ist in Abbildung 5.18 dargestellt. Eine ähnliche Operation, **RIGHT OUTER JOIN**, die durch ⟖ bezeichnet wird, behält jedes Tupel der *zweiten* bzw. *rechten* Relation S im Resultat von R ⟖ S. Eine dritte Operation, **FULL OUTER JOIN**, die durch ⟗ bezeichnet wird, behält alle Tupel der linken und rechten Relation, wenn keine Join-Tupel gefunden werden,

und füllt sie nach Bedarf mit Nullwerten auf. Die drei OUTER-JOIN-Operationen sind Teil des SQL2-Standards (siehe Kapitel 6).

Abbildung 5.18: Beispiel einer LEFT-OUTER-JOIN-Operation.

RESULTAT	VNAME	INITIAL	NNAME	ANAME
	John	B	Smith	null
	Franklin	T	Wong	Research
	Alicia	J	Zelaya	null
	Jennifer	S	Wallace	Administration
	Ramesh	K	Narayan	null
	Joyce	A	English	null
	Ahmad	V	Jabbar	null
	James	E	Borg	Headquarters

Die **OUTER-UNION**-Operation wurde entwickelt, um die Vereinigung von Tupeln aus zwei Relationen zuzulassen, die *nicht unionkompatibel* sind. Diese Operation erzeugt die Vereinigung aus Tupeln zweier Relationen, die **partiell kompatibel** sind, d.h., dass nur einige ihrer Attribute union-kompatibel sind. Es wird erwartet, dass die Liste der kompatiblen Attribute einen Schlüssel für beide Relationen beinhaltet. Tupel aus den Basisrelationen mit gleichem Schlüssel werden im Resultat nur einmal dargestellt und beinhalten Werte für alle Attribute. Die nicht union-kompatiblen Attribute aus beiden Relationen werden im Resultat beibehalten. Tupel, die für diese Attribute keine Werte haben, werden mit Nullwerten aufgefüllt. Eine OUTER UNION kann z.B. auf zwei Relationen angewandt werden, deren Schemas STUDENT(Name, SSN, Department, Advisor) und FACULTY(Name, SSN, Department, Rank) sind. Das resultierende Relationsschema ist R(Name, SSN, Department, Advisor, Rank) und alle Tupel beider Relationen werden im Resultat berücksichtigt. Studenten-Tupel besitzen im Rank-Attribut einen Nullwert, während Faculty-Tupel im Advisor-Attribut einen Nullwert beinhalten. Ein Tupel, das in beiden existiert, hat Werte für alle Attribute.[14]

In den meisten Anfragesprachen kommerzieller DBMS (jedoch nicht in der relationalen Algebra) sind weitere Funktionen verfügbar, mit denen Operationen auf Werte spezifiziert werden können, nachdem diese aus der Datenbank erzeugt wurden. Arithmetische Operationen wie +, – und * können z.B. auf numerische Werte angewandt werden.

5.6 Beispiele von Anfragen in relationaler Algebra

In diesem Abschnitt führen wir mehrere Beispiele zur Verwendung relationaler Algebra-Operationen auf. Alle Beispiele beziehen sich auf die Datenbank aus Abbildung 5.6. Im Allgemeinen kann die gleiche Anfrage mit Hilfe verschiedener Operationen auf unterschiedliche Weise angegeben werden. Wir zeigen jeweils eine Möglichkeit

14. Man beachte, dass OUTER UNION einem FULL OUTER JOIN entspricht, wenn *alle* JOIN-Attribute die gemeinsamen Attribute der beiden Relationen sind.

der Anfrage und überlassen es dem Leser, selbst entsprechende weitere Formulierungen zu finden.

Anfrage 1

Liste die Namen und Adressen aller Angestellten auf, die in der Abteilung 'Research' arbeiten.

$$\text{ABT\_RESEARCH} \leftarrow \sigma_{\text{ANAME='Research'}}(\text{ABTEILUNG})$$

$$\text{ANGEST\_RESEARCH} \leftarrow (\text{ABT\_RESEARCH} \bowtie_{\text{AbtNUMMER=ANR}} \text{ANGESTELLTER})$$

$$\text{RESULTAT} \leftarrow \pi_{\text{VNAME, NNAME, ADRESSE}}(\text{ANGEST\_RESEARCH})$$

Diese Anfrage lässt sich auch auf andere Arten definieren. Beispielsweise kann die Reihenfolge der JOIN- und SELECT-Operationen umkehrt werden oder der JOIN (nach der Umbenennung) durch einen NATURAL JOIN ersetzt werden.

Anfrage 2

Liste für jedes Projekt in 'Stafford' die Projektnummer, die Nummer der kontrollierenden Abteilung sowie Nachname, Adresse und Geburtsdatum des Abteilungsleiters auf.

$$\text{STAFFORD\_PROJEKTE} \leftarrow \sigma_{\text{PSTANDORT='Stafford'}}(\text{PROJEKT})$$

$$\text{KONTR\_ABT} \leftarrow (\text{STAFFORD\_PROJEKTE} \bowtie_{\text{ABTNR=AbtNUMMER}} \text{ABTEILUNG})$$

$$\text{PROJ\_ABT\_MGR} \leftarrow (\text{KONTR\_ABT} \bowtie_{\text{MGRSSN=SSN}} \text{ANGESTELLTER})$$

$$\text{RESULTAT} \leftarrow \pi_{\text{PNUMMER, ABTNR, NNAME, ADRESSE, GDATUM}} (\text{PROJ\_ABT\_MGR})$$

Anfrage 3

Finde die Namen der Angestellten, die an *allen* Projekten arbeiten, die von Abteilung 5 kontrolliert werden.

$$\text{ABT5\_PROJEKTE(PNR)} \leftarrow \pi_{\text{PNUMMER}} (\sigma_{\text{ANR=5}} (\text{PROJEKT}))$$

$$\text{ANGEST\_PROJEKT(SSN, PNR)} \leftarrow \pi_{\text{ESSN, PNR}} (\text{ARBEITET\_AN})$$

$$\text{RESULTAT\_ANGEST\_SSNS} \leftarrow \text{ANGEST\_PROJEKT} \div \text{ABT5\_PROJEKTE}$$

$$\text{RESULTAT} \leftarrow \pi_{\text{NNAME, VNAME}} (\text{RESULTAT\_ANGEST\_SSNS} * \text{ANGESTELLTER})$$

Anfrage 4

Erstelle eine Liste mit den Projektnummern von Projekten, an denen ein Angestellter arbeitet, dessen Nachname 'Smith' ist, der ein Mitarbeiter oder Abteilungsleiter der Abteilung ist, die das Projekt kontrolliert.

$$\text{SMITH(ESSN)} \leftarrow \pi_{\text{SSN}}(\sigma_{\text{NNAME='Smith'}}(\text{ANGESTELLTER}))$$

$$\text{SMITH\_MITARBEITER\_PROJ} \leftarrow \pi_{\text{PNR}}(\text{ARBEITET\_AN} * \text{SMITH})$$

$$\text{MGR} \leftarrow \pi_{\text{NNAME, ANUMMER}}(\text{ANGESTELLTER} \bowtie_{\text{SSN=MGRSSN}} \text{ABTEILUNG})$$

$$\text{SMITH\_LEITET\_ABT(ABTNR)} \leftarrow \pi_{\text{ANUMMER}}(\sigma_{\text{NNAME='Smith'}}(\text{MGR}))$$

$$\text{SMITH\_MGR\_PROJEKTE(PNR)} \leftarrow \pi_{\text{PNUMMER}}(\text{SMITH\_LEITET\_ABT} * \text{PROJEKT})$$

$$\text{RESULTAT} \leftarrow (\text{SMITH\_MITARBEITER\_PROJEKTE} \cup \text{SMITH\_MGR\_PROJEKTE})$$

Anfrage 5

Erstelle eine Liste mit den Namen aller Angestellten mit zwei oder mehr Angehörigen.
Diese Anfrage kann mit der *grundlegenden relationalen Algebra* nicht ausgeführt werden. Wir müssen die AGGREGATE-FUNCTION-Operation mit der COUNT-Aggregatfunktion verwenden. Wir gehen davon aus, dass die Angehörigen eines *bestimmten* Angestellten *unterschiedliche* ANGEHÖRIGER_NAME-Werte haben.

$$T_1(\text{SSN, ANZAHL\_ABTEILUNGEN}) \leftarrow {}_{\text{ESSN}}\Im_{\text{COUNT ANGEHÖRIGER\_NAME}}(\text{ANGEHÖRIGER})$$
$$T_2 \leftarrow \sigma_{\text{ANZAHL\_ABTEILUNGEN} \geq 2}(T_1)$$
$$\text{RESULTAT} \leftarrow \pi_{\text{NNAME, VNAME}}(T_2 * \text{ANGESTELLTER})$$

Anfrage 6

Liste die Namen aller Angestellten auf, die keine Angehörigen haben.

$$\text{ALLE\_ANGEST} \leftarrow \pi_{\text{SSN}}(\text{ANGESTELLTER})$$
$$\text{ANGEST\_MIT\_ANGEHÖRIGEN(SSN)} \leftarrow \pi_{\text{ESSN}}(\text{ANGEHÖRIGER})$$
$$\text{ANGEST\_OHNE\_ANGEHÖRIGE} \leftarrow (\text{ALLE\_ANGEST} - \text{ANGEST\_MIT\_ANGEHÖRIGEN})$$
$$\text{RESULTAT} \leftarrow \pi_{\text{NNAME, VNAME}}(\text{ANGEST\_OHNE\_ANGEHÖRIGE} * \text{ANGESTELLTER})$$

Anfrage 7

Erstelle eine Liste mit den Namen von Abteilungsleitern, die mindestens einen Angehörigen haben.

$$\text{MGR(SSN)} \leftarrow \pi_{\text{MGRSSN}}(\text{ABTEILUNG})$$
$$\text{ANGEST\_MIT\_ANGEHÖRIGEN(SSN)} \leftarrow \pi_{\text{ESSN}}(\text{ANGEHÖRIGER})$$
$$\text{MGR\_MIT\_ANGEHÖRIGEN} \leftarrow (\text{MGR} \cap \text{ANGEST\_MIT\_ANGEHÖRIGEN})$$
$$\text{RESULTAT} \leftarrow \pi_{\text{NANME, VNAME}}(\text{MGR\_MIT\_ANGEHÖRIGEN} * \text{ANGESTELLTER})$$

Wie zuvor erwähnt, lassen sich Anfragen im Allgemeinen auf vielerlei Art spezifizieren. Die Operationen können z.B. oft in verschiedenen Reihenfolgen benutzt werden. Außerdem können einige Operationen durch andere ersetzt werden. Beispielsweise kann die INTERSECTION-Operation in Anfrage 7 durch einen NATURAL JOIN ersetzt werden. Als Übung versuche man, jede der obigen Beispielanfragen mit unterschiedlichen Operationen umzuschreiben.[15] In Kapitel 6 und 7 wird gezeigt, wie diese Anfragen in anderen relationalen Sprachen formuliert werden können.

5.7 Zusammenfassung

In diesem Kapitel wurden die Modellierungskonzepte vorgestellt, die für das relationale Datenmodell definiert sind. Außerdem wurde die relationale Algebra gemeinsam mit weiteren Operationen beschrieben, die zur Manipulation von Relationen benutzt werden können. Wir haben mit der Einführung der Konzepte von Werteberei-

15. Wenn Anfragen optimiert werden (siehe Kapitel 18 der ungekürzten Ausgabe), wählt das System eine bestimmte Sequenz von Operationen, die einer Ausführungsstrategie entspricht, die sich effizient ausführen lässt.

chen, Attributen und Tupeln begonnen. Danach wurde ein Relationsschema als Liste von Attributen definiert, die die Struktur einer Relation beschreiben. Eine Relation oder ein Relationszustand ist eine Tupel-Menge, die dem Schema entspricht.

Relationen unterscheiden sich von gewöhnlichen Tabellen oder Dateien durch mehrere Merkmale. Erstens sind Tupel in einer Relation nicht geordnet. Zweitens müssen für die Ordnung von Attributen in einem Relationsschema auch die Werte innerhalb eines Tupels entsprechend geordnet werden. Wir haben eine alternative Definition einer Relation genannt, die diese beiden Ordnungen nicht voraussetzt, haben der Vereinfachung halber aber weiterhin die erste Definition verwendet, welche die Ordnung der Attribute und Tupel-Werte voraussetzt. Danach wurden Werte in Tupeln behandelt und Nullwerte für die Darstellung fehlender oder unbekannter Werte eingeführt.

Anschließend wurden die Einschränkungen im relationalen Modell behandelt, d.h. Einschränkungen für Wertebereiche, Schlüssel, einschließlich der Konzepte des Superschlüssels, der Schlüsselkandidaten und des Primärschlüssels, sowie die NICHT-NULL-Einschränkung für Attribute. Weiterhin wurden relationale Datenbanken und relationale Datenbankschemas beschrieben. Zu den weiteren relationalen Einschränkungen zählt die Entitätsintegritätseinschränkung, die verbietet, dass Primärschlüsselattribute Nullwerte enthalten. Die Einschränkung der referenziellen Integrität wird benutzt, um die Konsistenz zwischen Tupeln unterschiedlicher Relationen zu wahren, die miteinader in Beziehung stehen.

Zu den Veränderungsoperationen für das relationale Modell zählen INSERT, DELETE und UPDATE. Jede Operation kann bestimmte Einschränkungsarten verletzen. Wenn eine Operation angewandt wird, muss der Datenbankzustand nach der Ausführung der Operation überprüft werden, um sicherzustellen, dass keine Einschränkungen verletzt sind.

Anschließend wurde die relationale Algebra beschrieben. Dabei handelt es sich um eine Menge von Operationen zur Suche auf Relationen, um Anfragen formulieren zu können. Wir haben die verschiedenen Operationen und die Anfragearten, für die sie sich eignen, anhand von Beispielen beschrieben. Tabelle 5.1 enthält eine Aufstellung der verschiedenen relationalen Algebra-Operationen, die in diesem Kapitel behandelt wurden. Danach wurden binäre Mengenoperationen beschrieben, die voraussetzen, dass Relationen, auf die sie angewandt werden, unionkompatibel sind; hierzu zählen UNION, INTERSECTION und DIFFERENCE. Die CARTESIAN-PRODUCT-Operation (kartesisches Produkt) ist eine weitere Mengenoperation, mit der Tupel aus zwei Relationen vollständig kombiniert werden können. Wir haben gezeigt, wie das kartesische Produkt, gefolgt von SELECT, benutzt werden kann, um zusammenhängende Tupel aus zwei Relationen zu identifizieren. Die Join-Operationen THETA JOIN, EQUI-JOIN und NATURAL JOIN können Tupel direkt identifizieren und kombinieren.

Schließlich wurden einige wichtige Anfragearten behandelt, die nicht mit den relationalen Algebra-Operationen formuliert werden können. Wir haben die AGGREGATE-FUNCTION-Operation zur Formulierung von Aggregatsanfragen eingeführt. Ferner wurden rekursive Anfragen beschrieben und die Art, in der rekursive Anfragen spezifiziert werden können. Den Abschluss bildeten die Operationen OUTER JOIN und OUTER UNION, die JOIN und UNION erweitern.

Tabelle 5.1: Operationen der relationalen Algebra.

Operation	Zweck	Notation
SELECT	Wählt alle Tupel, die die Auswahlbedingung erfüllen, aus einer Relation R.	$\sigma_{<\text{Auswahlbedingung}>}(R)$
PROJECT	Erzeugt eine neue Relation mit nur einigen Attributen von R und entfernt Duplikat-Tupel.	$\pi_{<\text{Attributliste}>}(R)$
THETA JOIN	Erzeugt alle Kombinationen von Tupeln aus R_1 und R_2, die die JOIN-Bedingung erfüllen.	$R_1 \bowtie_{<\text{Join-Bedingung}>} R_2$
EQUIJOIN	Erzeugt alle Kombinationen von Tupeln aus R_1 und R_2, die eine JOIN-Bedingung erfüllen, ausschließlich mit Gleichheitsvergleichen.	$R_1 \bowtie_{<\text{Join-Attribute 1}>, <\text{Join-Attribute 2}>} R_2$ $R_1 \bowtie_{(<\text{Join-Attribute 1}>), (<\text{Join-Attribute 2}>)} R_2$
NATURAL JOIN	Wie EQUIJOIN, außer dass die JOIN-Attribute von R_2 nicht in die resultierende Relation einbezogen werden. Wenn die JOIN-Attribute die gleichen Namen haben, müssen sie überhaupt nicht spezifiziert werden.	$R_{1*<\text{Join-Bedingung}>}R_2$ oder $R_{1*(<\text{Join-Attribute 1}>),}$ $_{(<\text{Join-Attribute 2}>)}R_2$ oder $R_1 * R_2$
UNION	Erzeugt eine Relation, die alle Tupel von R_1 oder R_2 oder beiden beinhaltet. R_1 und R_2 müssen unionskompatibel sein.	$R_1 \cup R_2$
INTERSECTION	Erzeugt eine Relation, die alle Tupel von R_1 und R_2 beinhaltet. R_1 und R_2 müssen unionskompatibel sein.	$R_1 \cap R_2$
DIFFERENCE	Erzeugt eine Relation, die alle Tupel von R_1 beinhaltet, die nicht in R_2 sind. R_1 und R_2 müssen unionskompatibel sein.	$R_1 - R_2$
CARTESIAN PRODUCT	Erzeugt eine Relation, die die Attribute von R_1 und R_2 umfasst und als Tupel alle möglichen Tupel-Kombinationen aus R_1 und R_2 beinhaltet.	$R_1 \times R_2$
DIVISION	Erzeugt eine Relation R(X), die alle Tupel t[X] aus $R_1(Z)$ beinhaltet, die in R_1 in Kombination mit jedem Tupel aus $R_2(Y)$ erscheinen, wobei $Z = X \cup Y$.	$R_1(Z) \div R_2(Y)$

WIEDERHOLUNGSFRAGEN

1. Definieren Sie die folgenden Begriffe: *Wertebereich*, *Attribut*, *n-Tupel*, *Relationsschema*, *Relationszustand*, *Grad einer Relation*, *relationales Datenbankschema*, *relationaler Datenbankzustand*.
2. Warum sind Tupel in einer Relation nicht geordnet?
3. Warum sind Duplikat-Tupel in einer Relation nicht zulässig?
4. Was ist der Unterschied zwischen einem Schlüssel und einem Superschlüssel?
5. Warum definieren wir einen der Kandidatenschlüssel einer Relation als Primärschlüssel?
6. Diskutieren Sie die Merkmale von Relationen, durch die sie sich von gewöhnlichen Tabellen und Dateien unterscheiden.
7. Diskutieren Sie die verschiedenen Gründe, die zum Auftreten von Nullwerten in Relationen führen.
8. Diskutieren Sie die Entitätsintegritätseinschränkungen und referenzielle Integrität. Warum gelten diese beiden Einschränkungen als wichtig?
9. Definieren Sie den Begriff *Fremdschlüssel*. Wofür wird dieses Konzept benutzt? Welche Rolle spielt es in der JOIN-Operation?
10. Diskutieren Sie die verschiedenen UPDATE-Operationen für Relationen und die Arten von Integritätseinschränkungen, die für jede UPDATE-Operation geprüft werden müssen.
11. Führen Sie alle Operationen der relationalen Algebra mit dem jeweiligen Zweck auf.
12. Was versteht man unter Unionkompatibilität? Warum setzen die Operationen UNION, INTERSECTION und DIFFERENCE voraus, dass die Relationen, auf die sie angewandt werden, unionkompatibel sind?
13. Diskutieren Sie einige Anfragearten, bei denen die Umbenennung von Attributen erforderlich ist, um die Anfrage unzweideutig spezifizieren zu können.
14. Diskutieren Sie die verschiedenen Arten von JOIN-Operationen. Wofür wird ein THETA JOIN benötigt?
15. Was ist die FUNCTION-Operation? Wozu dient sie?
16. Wodurch unterscheiden sich OUTER-JOIN-Operationen von den (inneren) JOIN-Operationen? Wodurch unterscheidet sich die OUTER-UNION-Operation von einer UNION?

ÜBUNGEN

17. Zeigen Sie das Ergebnis jeder Beispielanfrage in Abschnitt 5.6 für die Datenbank aus Abbildung 5.6.

18. Spezifizieren Sie die folgenden Anfragen für das Datenbankschema aus Abbildung 5.5 unter Verwendung der in diesem Kapitel beschriebenen relationalen Operatoren. Zeigen Sie auch das Resultat jeder Anfrage für die Datenbank aus Abbildung 5.6.

    a. Listen Sie die Namen aller Angestellten in Abteilung 5 auf, die mehr als 10 Stunden pro Woche am Projekt 'ProduktX' arbeiten.

    b. Erstellen Sie eine Liste mit den Namen aller Angestellten, die einen Angehörigen mit dem gleichen Vornamen wie sie selbst haben.

    c. Suchen Sie die Namen aller Angestellten, die direkt von 'Franklin Wong' beaufsichtigt werden.

    d. Erstellen Sie für jedes Projekt eine Liste mit dem Projektnamen und den Gesamtstunden pro Woche, die (von allen Angestellten) an diesem Projekt aufgewendet werden.

    e. Suchen Sie die Namen aller Angestellten, die an jedem Projekt arbeiten.

    f. Suchen Sie die Namen aller Angestellten, die an keinem Projekt arbeiten.

    g. Listen Sie für jede Abteilung den Abteilungsnamen und das Durchschnittsgehalt aller Angestellten auf, die in dieser Abteilung arbeiten.

    h. Listen Sie das Durchschnittsgehalt aller weiblichen Angestellten auf.

    i. Suchen Sie die Namen und Adressen aller Angestellten, die mindestens an einem Projekt in Houston arbeiten, dessen zuständige Abteilung ihren Standort aber nicht in Houston hat.

    j. Erstellen Sie eine Liste mit den Nachnamen aller Abteilungsleiter (Manager), die keine Angehörigen haben.

19. Wir nehmen an, dass jede der folgenden Update-Operationen direkt auf die Datenbank von Abbildung 5.7 angewandt wird. Diskutieren Sie *alle* Integritätseinschränkungen, die von jeder Operation verletzt werden, soweit zutreffend, und die verschiedenen Arten der Auferlegung dieser Einschränkungen.

    a. Fügen Sie <'Robert', 'F', 'Scott', '943775543', '1952-06-21', '2365 Newcastle Rd, Bellaire, TX', M, 58000, '888665555', 1> in ANGESTELLTER ein.

    b. Fügen Sie <'ProduktA', 4, 'Bellaire', 2> in PROJEKT ein.

    c. Fügen Sie <'Produktion', 4, '943775543', '1998-10-01'> in ABTEILUNG ein.

    d. Fügen Sie <'677678989', null, '40.0' in ARBEITET_AN ein.

    e. Fügen Sie <'453453453', 'John', M, '1970-12-12', 'EHEFRAU'> in ANGEHÖRIGER ein.

    f. Löschen Sie das Tupel ARBEITET_AN mit ESSN = '333445555'.

    g. Löschen Sie das Tupel ANGESTELLTER mit SSN = '987654321'.

h. Löschen Sie das Tupel PROJEKT mit PNAME = 'ProduktX'.

i. Ändern Sie MGRSSN und MGR_ANFANGSDATUM des Tupels ABTEILUNG mit ABTNUMMER = 5 in '123456789' bzw. '1999-10-01'.

j. Ändern Sie das Attribut SUPERSSN des Tupels ANGESTELLTER mit SSN = '999887777' in '943775543'.

k. Ändern Sie das Attribut STUNDENT des Tupels ARBEITET_AN mit ESSN = '999887777' und PNR = 10 in '5.0'.

20. Betrachten Sie das relationale Datenbankschema AIRLINE in Abbildung 5.19, das eine Datenbank für ein Fluginformationssystem beschreibt. Jeder Flug (FLIGHT) wird durch eine Flugnummer (NUMBER) identifiziert und besteht aus einem oder mehreren Flugabschnitten (FLIGHT_LEGS) mit Abschnittsnummern (LEG_NUMBERS) 1, 2, 3 usw. Jeder Flugabschnitt hat planmäßige Lande- und Abflugzeiten, Flughäfen und viele Streckeninstanzen (LEG_INSTANCES) – je eine für jedes Flugdatum (DATE). Pro Flug werden Flugpreise (FARES) verwaltet. Für jede Streckeninstanz werden Sitzreservierungen (SEAT_RESERVATIONS) geführt, ebenso wie das Flugzeug (AIRPLANE) für den jeweiligen Abschnitt und die tatsächlichen Lande- und Abflugzeiten und Flughäfen. Ein Flugzeug (AIRPLANE) wird durch eine Kennung (AIRPLANE_ID) und einen Flugzeugtyp (AIRPLANE_TYPE) identifiziert. CAN_LAND bezieht sich auf Flugzeugtypen (AIRPLANE_TYPES) die auf bestimmten Flughäfen (AIRPORTS) landen können. Ein Flughafen (AIRPORT) wird durch einen Flughafencode (AIRPORT_CODE) identifiziert. Spezifizieren Sie die folgenden Anfragen in relationaler Algebra:

a. Erstellen Sie für jeden Flug eine Liste mit der Flugnummer, dem Abflughafen für den ersten Flugabschnitt und den Ankunftsflughafen für den letzten Flugabschnitt.

b. Erstellen Sie eine Liste mit den Flugnummern und Wochentagen aller Flüge oder Flugabschnitte, die vom Houston Intercontinental Airport (Flughafencode 'IAH') abfliegen und in Los Angeles International Airport (Flughafencode 'LAX') landen.

c. Erstellen Sie eine Liste mit der Flugnummer, dem Abflughafencode, der planmäßigen Abflugzeit, dem Landeflughafencode, der planmäßigen Landezeit und den Wochentagen aller Flüge oder Flugabschnitte, die von einem Flughafen der Stadt Houston abfliegen und auf einem Flughafen der Stadt Los Angeles landen.

d. Erstellen Sie eine Liste aller Flugpreisinformationen für Flugnummer 'CO197'.

e. Listen Sie die Anzahl der auf Flugnummer 'CO197' am '1999-10-09' verfügbaren Sitzplätze auf.

21. Betrachten Sie ein Update für die AIRLINE-Datenbank, um eine Reservierung für einen bestimmten Flug oder Flugabschnitt an einem bestimmten Datum einzugeben.

a. Geben Sie die Operationen für dieses Update an.

b. Welche Einschränkungstypen sollen erwartungsgemäß geprüft werden?

c. Welche dieser Einschränkungen betreffen Schlüssel, Entitätsintegrität und Referenzintegrität und welche nicht?

d. Spezifizieren Sie alle Referenzintegritätseinschränkungen für Abbildung 5.19.

Copyright