Datenmanipulation in SQL (1):

Unter Datenmanipulation wird sowohl der lesende Zugriff auf die Daten (Select Statement) als auch die Änderung von Daten (Insert, Delete, Update) subsummiert. Wir beginnen mit der Betrachtung des Select Statement. Ein wesentlicher Begriff ist der des Subselect (und damit zusammenhängend der der Subquery).

Subselect:

Ein Subselect ist ein Statement der Form

Select [distinct | all] Kommaliste von Spaltenausdrücken

From Kommaliste von Tabellenausdrücken

Where Suchbedingung auf Zeilenebene

Group By Kommaliste von Spalten(ausdrücken)

Having Suchbedingung auf Gruppenebene

Unter einer Subquery verstehen wir ein geklammertes Subselect.

In Spaltenausdrücken sind erlaubt:

- **SQL Operatoren**
- **Spalten eines Tabellenausdrucks**
- **Konstanten**
- **Funktionswerte von SQL Funktionen**
- **Subqueries**

Als Tabellenausdruck sind erlaubt:

- **Tabellen**
- **Views**
- **Synonyme**
- **Subqueries**
- **Join – Konstrukte**

Eine Suchbedingung ist ein Spaltenausdruck, der einen boole'schen Wert hat.

SQL Operatoren und Funktionen

Operatoren:

Arithmetisch: +, -, *, /

String: ||, (+)

Datetime/Interval: +, -

Vergleich: =, != (<>), <, <=, >, >=

Boolean: not, and, or

Funktionen:

Arithmetisch: mod, round, trunc, floor, ceil

String:

- **length, size**
- **substring, position, replace**
- **lower, upper, initcap**
- **ltrim, rtrim, trim**
- **lpad, rpad, pad**

Datetime/Interval: Proprietär

Typkonversion: Explizit und Implizit

Abbildung NULL=>Exception: nvl, coalesce

Case Ausdrücke: decode, case

- **decode(Expression,s1,W1,s2,W2,...,W)**
- **case Expression**

  when s1 then W1

  when s2 then W2

  else W

  end

Nach dieser allgemeinen Betrachtung soll der Bezug zwischen den Operatoren der relationalen Algebra und dem Select Statement dargestellt werden.

Wir betrachten zunächst den Zugriff auf eine Tabelle:

In der relationalen Algebra gibt es hierzu die Operatoren

- **Projektion**
- **Restriktion**

Die Projektion ist in SQL realisiert durch

Select distinct Spaltenliste

From Tabelle

Die Restriktion ist in SQL realisiert durch

Select *

From Tabelle

Where Suchbedingung

Im Rahmen der Suchbedingung dürfen folgende Operatoren verwendet werden:

- **Boole'sche Operatoren: not, and, or**
- **Vergleichsoperatoren =, != (<>), <, <=, >, >=**
- **Arithmetische Operatoren: +, -, *, /, mod**
- **between / not between**
- **is null / is not null**
- **like**
- **in**

Verwendung von Subqueries:

- **In darf nicht nur zum Vergleich mit konstanten Listen verwendet werden, sondern auch zum Vergleich mit Listen, die durch eine Subquery repräsentiert werden.**

- **Vergleichsoperatoren dürfen ebenfalls zum Vergleich mit Listen, die durch eine Subquery repräsentiert werden, herangezogen werden, wobei für den Fall, daß das Ergebnis der Subquery nicht atomar, also eine Liste aus mehr als einem Element ist, eines der Schlüsselworte any/some oder all nach dem Vergleichsoperator zu verwenden ist.**

- **Exists darf auf eine Subquery angewandt werden, wobei exists subquery genau dann wahr ist, wenn die Subquery mindestens eine Zeile liefert.**

In der Subquery darf Bezug auf die Tabelle genommen werden, auf der die Restriktion durchgeführt wird. Man spricht dann von einer korrelierten Subquery.

Wir betrachten als nächstes den Zugriff auf mehrere Tabellen, wobei zunächst nicht auf die mengentheoretischen Operatoren eingegangen wird.

In der relationalen Algebra gibt es hierzu die Operatoren

- **Kartesisches Produkt**
- **Join**
- **Quotient**

Das kartesische Produkt zweier Tabellen T1 und T2 ist definiert durch

Select *

From T1,T2

Diese Operation sollte nur dann vom DBS angewandt werden, wenn eine der beteiligten Tabellen aus einer Zeile oder sehr wenigen Zeilen besteht.

Join Konstrukte gibt es in der relationalen Algebra einige:

- **Equi Join: Verknüpfung aufgrund der Gleichheit von Attributwerten**
- **Natural Join: Verknüpfung aufgrund der Gleichheit von Fremdschlüsselwerten und zugehörigen Schlüsselkandidatwerten (und Elimination der Fremdschlüsselattribute oder der Schlüsselkandidatattribute)**
- **Semi Join: Projektion des Natural Join auf eine Tabelle**
- **Inner Join: alternativer Begriff zum Natural Join**
- **Left Outer Join, Right Outer Join, Full Outer Join: Ergänzung des natural / inner Joins durch Berücksichtigung von Masterzeilen ohne passende Details, Detailzeilen ohne passenden Master sowie beidem**
- **Theta Join: Verknüpfung aufgrund von Vergleichen von Attributwerten**
- **Auto / Self Join: Verknüpfung einer Tabelle mit sich selbst**

Wir betrachten hier:

- **Inner Joins**
- **Outer Joins**
- **Auto Join**

(Der Theta Join zeichnet sich nur dadurch aus, daß in der ersten betrachteten Join Variante andere Vergleichsoperatoren möglich sind. Er wird hier nicht weiter betrachtet. Es sollte allerdings bemerkt werden, daß Theta Joins oft ein Performance Thema sind!)

Seien T1 und T2 zwei Tabellen, k Schlüsselkandidat von T1 und zugehöriger Fremdschlüssel von T2, s1 weitere Spalte in T1 und s2 weitere Spalten in T2. Dann ist der Inner Join von T1 und T2 in SQL implementiert durch

Select T1.k, T1.s1, T2.s2

From T1, T2

Where T1.k = T2.k

Alternativ gibt es im SQL Standard Join – Konstrukte analog denen der relationalen Algebra, z.B. für den inner join

Select T1.k, T2.s1, T2.s2

From T1 inner join T2 on T1.k=T2.k

Diese Join Konstrukte haben sich allerdings noch nicht allgemein durchgesetzt, z.B. stehen sie bei Oracle nicht zur Verfügung.

Ausgehend von dem inner join – Konstrukt ist es offensichtlich, wie Outer Joins entsprechend dargestellt werden:

Select T1.k, T2.s1, T2.s2

From T1 left / right / full outer join T2 on T1.k=T2.k

Oracle (Release < 9i) verwendet für den Outer Join lediglich proprietäre Konstrukte:

Left Outer Join:

```
Select T1.k, T1.s1, T2.s2
From T1, T2
Where T1.k = T2.k(+)
```

Right Outer Join:

```
Select T1.k, T1.s1, T2.s2
From T1, T2
Where T1.k(+) = T2.k
```

Ein Auto Join zeichnet sich dadurch aus, daß T1 und T2 die gleiche Tabelle T repräsentieren. In diesem Fall muß ein Konstrukt verwendet werden, das ansonsten auch sinnvoll eingesetzt werden kann: der Korrelationsname. Im Fall des Auto Joins sollte er die Rolle andeuten, in der die Tabelle jeweils verwendet wird, ansonsten werden Korrelationsnamen häufig verwendet als Kürzel von Tabellennamen. Sei T jetzt eine Tabelle mit Schlüsselkandidat sk, Fremdschlüssel fk, der sich auf sk bezieht, und einer weiteren Spalte s. Dann kann der auto join wie folgt definiert werden (mit Korrelationsnamen t1 und t2 für T):

```
Select t1.sk, t1.s, t2.sk, t2.s

From T t1, T t2

Where t1.sk = t2.fk
```

Gruppierung, Aggregate

SQL bietet Aggregate an (statistische Funktionen), die auf der Gruppierung von Daten beruhen. Es sind dies die Aggregate

- **count (*) zum Zählen der Zeilen**

- **count**
- **min**
- **max**
- **sum**
- **avg**
- **var**
- **stddev**

Diese statistischen Funktionen haben Argumente der Form [distinct] Spaltenname, um die jeweilige statistische Funktion auf die [verschiedenen] Werte ungleich NULL in einer Spalte anzuwenden.

I.A. werden Aggregate für Gruppen von Daten gebildet. Diese Gruppen werden in einer group by – Klausel spezifiziert: Die Daten mit gleichen Werten in den Spalten der in der group by – Klausel spezifizierten Spaltenliste repräsentieren eine Gruppe.

Ohne group by – Klausel beziehen sich sämtliche Aggregate auf die Gesamtheit der betrachteten Daten!

Es ist folgende Einschränkung in SQL zu beachten: In der Select – Liste dürfen außerhalb von Aggregaten nur Spalten vorkommen, die in der group by – Klausel stehen!

Wie mit der where – Klausel eine Suchbedingung auf Zeilenebene definiert werden kann, kann mit der having – Klausel eine Suchbedingung auf Gruppenebene definiert werden.

An dieser Stelle kommen wir wieder zurück zum Quotient der relationalen Algebra:

Ist R eine Relation mit zwei Attributkombinationen A und B und S eine Relation, deren Attribute gerade die aus B sind, so ist der Quotient von R und S definiert als die Menge aller Werte a in A, so daß (a,b) für alle b in S in A ist.

Interpretation: Der Quotient von R und S ist die Menge aller Gruppen a (nach Gruppierung bezüglich der Attribute in A), für die die Anzahl zugehöriger Elemente mit b in S gleich der Anzahl der Elemente von S ist.

Diese Interpretation führt zu folgendem SQL – Statement (für den Fall, daß A und B aus einem Attribut bestehen):

```
select R.a
from R, S
where R.b = S.b
group by R.a
having count(*) = (select count(*) from S);
```

Eine weitere Implementierung des Quotienten aus der relationalen Algebra in SQL erhält man, wenn man die Definition des Quotienten folgendermaßen umformuliert:

Der Quotient von R und S ist die Menge aller Werte a in A, so daß kein b in S existiert mit der Eigenschaft, daß (a,b) nicht in R ist, also keine Zeile r in R existiert mit r=(a,b):

```
select a
from R RA
where not exists
  (
  select *
    from S
    where not exists
      (
      select *
      from R
      where a=RA.a
      and b=B.b
      )
    )
```

Diese Formulierung ist i.a. weniger performant!

Wir betrachten schließlich die mengentheoretischen Operatoren in SQL.

In der relationalen Algebra gibt es die mengentheoretischen Operatoren

- **Vereinigung**
- **Durchschnitt**
- **Differenz**

SQL bietet entsprechende Konstrukte an, wobei union in zwei Ausprägungen vorliegt, die nicht kombiniert werden können.

- **Union (mengenorientiert, d.h., Duplikate werden anschließend eliminiert)**
- **Union all (satzorientiert)**
- **Intersect**
- **Minus (Oracle) bzw. Except (DB2)**

Mit dem Einsatz der mengentheoretischen Operatoren wird der syntaktische Rahmen des Subselect verlassen. Es handelt sich jetzt um eine vollständige Select – Anweisung. Man sollte beachten, daß in vielen Situationen nur Subselects verwendet werden dürfen, nicht aber vollständige Selects!

Abschließend soll auf eine Klausel der vollständigen Select – Anweisung eingegangen werden, die nicht den Operatoren der relationalen Algebra zugeordnet werden kann, die aber für die Praxis von großer Bedeutung ist: die order by – Klausel.

Diese Klausel erlaubt es, die Resultate einer Abfrage zu sortieren. Die order by – Klausel ist die zuletzt ausgewertete Klausel einer Select-Anweisung. Hier kann eine Liste von Spaltennamen oder –nummern der resultierenden Relation eingegeben werden und pro Spalte spezifiziert werden, ob aufsteigend (asc, default) oder absteigend (desc) sortiert werden soll.

Die Spaltennamen der resultierenden Relation sind die des ersten beteiligten Subselects, diese werden aus den Namen der referenzierten Spalten in der Select-Liste abgeleitet, falls möglich, ansonsten sollten sie mit einer as – Klausel definiert werden.

Datenmanipulation in SQL (2)

Insert

Satzorientierte Variante:

Insert into Tabelle

(Kommaliste von Spalten)

values

(Kommaliste von Ausdrücken)

Mengenorientierte Variante

Insert into Tabelle

(Kommaliste von Spalten)

Subselect

Bemerkungen:

Fehlt die Spaltenliste für die Tabelle, ist die Liste aller Spalten gemeint

Jede Spalte, die not null definiert ist und keinen Default Wert hat, muß in der Spaltenliste für die Tabelle vorkommen.

Spaltenliste und Ausdrucksliste in der values Klausel bzw. der Select Liste des Subselects müssen verträglich sein.

Einige DBS verbieten, daß das Subselect sich auf die Tabelle bezieht, in die eingefügt wird!

Neben der o.a. mengentheoretischen Variante, die es erlaubt, Daten aus der Datenbank in eine Tabelle einzufügen, gibt es Methoden, Daten aus Hauptspeicherstrukturen oder Dateien in eine Tabelle mengenorientiert einzufügen. Diese Methoden sind allerdings noch wenig standardisiert.

- **Array Insert (Oracle)**
- **Mehrere Zeilen in der Values-Klausel (DB2)**
- **Sqlldr (Oracle)**
- **Load Utility (DB2)**

Delete:

delete

from Tabelle

where Suchbedingung

Bemerkungen:

Einige DBS verbieten es, in der Suchbedingung die Tabelle zu referenzieren, aus der gelöscht wird.

Es ist zu beachten, daß das vollständige Löschen der Daten aus einer Tabelle i.a. effizienter möglich ist als per delete-Anweisung (truncate Anweisung bei Oracle, Load einer leeren Datei in die Tabelle bei DB2)!

Update

Das Update Statement in SQL hat prinzipiell die Form

update tabelle

set

Spalte = Ausdruck,

Spalte = Ausdruck,

...

where suchbedingung

Bemerkungen:

Die Ausdrücke in der set-Klausel können auch Subqueries sein, die maximal eine Zeile liefern. Man spricht dann von einem korrelierten Update.

Einige DBS verbieten, daß in den Ausdrücken in der set-Klausel oder/und in der Suchbedingung die zu ändernde Tabelle vorkommt.

Einige DBMS erlauben es, die set – Klausel in der Form

Spaltenliste = Ausdruckliste

zu formulieren.