QSQL: Eine Erweiterung der Anfragesprache SQL um Scoring-Werte und Gewichtung

Sebastian Lehrack und Ingo Schmitt
BTU Cottbus – LS Datenbank- und Informationssysteme
Cottbus

Schlüsselworte:

Suchen, Anfragesprachen, SQL, Ähnlichkeitsbedingungen, Scoring-Werte, Gewichtung, Quantenlogik

Einführung und Motivation

Die traditionelle Auswertung einer Datenbankanfrage ermittelt für jedes Tupel der angefragten Tabelle entweder den Wahrheitswert Wahr oder den Wahrheitswert Falsch. Alle „wahren" Tupel bilden abschließend die Ergebnismenge der Anfrage. Für viele Anwendungsszenarien ist diese Auswertungssemantik zu restriktiv, insbesondere wenn ein differenziertes Anfrageergebnis benötigt wird. Dies setzt oft eine Aussage über den *Grad der Erfüllung* einer Bedingung bzw. Anfrage voraus.

Zur Motivation wollen wir ein Beispielszenario betrachten, welches sich mit der Bewertung und Auswahl von TV-Geräten beschäftigt. Dabei sollen neben weiteren Attributen folgende Eigenschaften für jedes TV-Gerät gespeichert werden: Name, Bedienung (BE), Bildqualität (BQ), Optischer Tonanschluss (OTA), Tonqualität (TQ) und Status. Die drei Attribute Bedienung, Bildqualität und Tonqualität sind jeweils mit einer Note zwischen 1 und 6 bewertet, wobei die Note 1 die beste und die Note 6 die schlechteste mögliche Bewertung darstellt. Die möglichen Werte des Attributes Status sind verfügbar (V), ausverkauft (A) und bestellt (B).

Der Besucher eines Internet-Geschäftes für TV-Geräte könnte nun folgende Anfrage stellen: Ich möchte ein Geräte finden, welches eine möglichst einfache Bedienung und eine möglichst sehr gute Bildqualität besitzt. Falls das betrachtete Gerät keinen optischen Tonanschluss zu einem externen Soundsystem anbietet, dann soll das TV-Gerät eine möglichst sehr gute interne Tonqualität aufweisen. Diese Anfrage kann folgendermaßen als logische Bedingung formuliert werden:

BE ~ 1 **AND** BQ ~ 1 **AND** (OTA=Ja **OR** (OTA=Nein **AND** TQ ~ 1))

Die Vagheit der Prädikate „möglichst einfache Bedienung", „möglichst sehr gute Bildqualität" und „möglichst sehr gute Tonqualität" können nur sehr unzureichend mittels der strikten zweiwertigen Auswertungslogik (Wahr oder Falsch) ausgedrückt werden. Zur Demonstration kann die folgende Tabelle betrachtet werden:

TV-Geräte				
Name	**Bedienung**	**Bildqualität**	**Optischer TA**	**Tonqualität**
TV2	2 (Wahr)	1 (Wahr)	Ja	1 (Wahr)
TV1	2 (Wahr)	2 (Wahr)	Ja	2 (Wahr)
TV3	1 (Wahr)	2 (Wahr)	Nein	2 (Wahr)
TV4	3 (Falsch)	1 (Wahr)	Nein	4 (Falsch)
...	...	...	...	...

Hinter den Bewertungsnoten befinden sich in Klammern die Wahrheitswerte für die Auswertung der entsprechenden vagen Teilbedingungen, wenn die Noten 1 und 2 als akzeptabel definiert werden. Entsprechend der oben formulierten Bespielanfrage werden die TV-Geräte *TV1*, *TV2* und *TV3* als gleichwertige Ergebnistupel zurück geliefert, wenn auf die herkömmliche zweiwertige Auswertungssemantik zurück gegriffen wird. Offensichtlich gehen bei dieser Ergebnisberechnung wichtige Differenzierungsmerkmale zwischen den TV-Geräten verloren. Betrachtet man die Bewertungsnoten bezüglich der angefragten Bedingungen genauer, stellt man fest, dass das TV-Gerät TV2 als bestes ermittelt werden müsste.

Der Anfragende ist viel mehr daran interessiert zu wissen, zu welchem Grad das entsprechende Tupel die Anfrage erfüllt. Dieser Erfüllungsgrad, auch Score-Wert genannt, wird durch einen kontinuierlichen Wert zwischen 0 und 1 ausgedrückt. Nimmt man die berechneten Score-Werte als Grundlage für die Ergebnisdarstellung kann ein sogenanntes Ranking zwischen den Ergebnistupeln erzeugt werden, wobei die besten Tupel, d.h. Tupel mit den größten Score-Werten, als erstes in der Ergebnisliste erscheinen.

In den folgenden Abschnitten wird kurz die theoretische Basis für die Erweiterung der Auswertungssemantik um Ähnlichkeitsprädikate skizziert, die praktische Umsetzung QSQL als SQL-Dialekt vorgestellt, sowie die Architektur des JDBC-Treibers für QSQL angerissen.

Theoretische Grundlagen

In diesem Kapitel wird eine kurze Einführung in die theoretischen Grundlagen für QSQL gegeben. Für eine detaillierte Darstellung der zu Grunde liegenden Theorie wird auf die entsprechenden Referenzen verwiesen.

Die Kernidee der neuen Auswertungssemantik ist die Anwendung eines mathematischen Vektorraummodells aus der Quantenmechanik und -logik. Dabei werden die Attributwerte des abgefragten Tupels in einem normalisierten Vektor kodiert. Die zu verarbeitende Anfrage erzeugt dagegen ein charakteristischen Vektor*unter*raum. Das Auswertungsergebnis, d.h. der Score-Wert, ist dann determiniert durch den einschließenden Winkel zwischen Tupelvektor und Anfragevektorunterraum. Der quadrierte Kosinus dieses Winkels ist ein reeller Wert zwischen 0 und 1 und kann dementsprechend als Ähnlichkeitsmaß bzw. als Score-Wert interpretiert werden. Ein Score-Wert von 1 entspricht dabei der maximalen Ähnlichkeit und der Score-Wert 0 der maximalen Unähnlichkeit zwischen dem betrachteten Tupel und der gestellten Anfrage.

Als Beispiel soll das oben eingeführte TV-Beispiel wiederholt betrachtet werden.

TV-Geräte					
Name	**Bedienung**	**Bildqualität**	**Optischer TA**	**Tonqualität**	**Score-Wert**
TV2	2 (0.8)	1 (1.0)	Ja	1 (1.0)	0.9
TV1	2 (0.8)	2 (0.8)	Ja	2 (0.8)	0.72
TV3	1 (1.0)	2 (0.8)	Nein	2 (0.8)	0.64
TV4	3 (0.6)	1 (1.0)	Nein	4 (0.4)	0.32
...	...	...	...	...	...

Im Gegensatz zur obigen Tabelle wird hier in Klammern der Score-Wert für die jeweilige Ähnlichkeitsbedingung angegeben. So ergibt die Note 1 den maximalen Score-Wert von 1 und die Note 3 lediglich ein Score-Wert von 0.6. In der Spalte Score-Wert befindet sich für jedes Tupel der Gesamt-Score-Wert. Deutlich ist nun die gewünschte Ausdifferenzierung der Tupel entsprechend der gestellten Anfrage zu erkennen.

Die SQL-Erweiterung QSQL (Quantum SQL)

Die Anfragesprache SQL ist der etablierte Standard zum Zugriff auf objekt-relationale Datenbanksysteme. Seit seiner Einführung in den 70er Jahren ist ihre praktische Relevanz kontinuierlich gestiegen. Aus diesem Grund wurden die theoretischen Konzepte des quantenlogischen Auswertungsmodell aus dem vorangegangen Kapitel in SQL integriert. Dadurch sind sie nun einer breiten Entwicklerschicht im Datenbankenbereich zugänglich. Der bisher vorhandene Funktionsumfang von SQL bleibt dabei vollständig in QSQL erhalten, d.h. alle SQL-Anfragen können auch in QSQL wie gewohnt ausgewertet werden. Im folgenden Abschnitt werden die Grundprinzipien von QSQL kurz anhand von zwei Beispielanfragen aus dem eingeführten Szenario vorgestellt.

Ein wichtiger Unterschied zur herkömmlichen SQL-Anfrageauswertung besteht in der Berechnung des Score-Wertes für jedes Ergebnistupel. Der Score-Wert wird in dem automatisch erzeugten Attribut `scoreval` abgelegt. Traditionelle SQL-Anfragen ohne Ähnlichkeitsbedingungen erzeugen für alle Ergebnistupel einen Score-Wert von 1.

Die Selektion von Tupeln aus einer oder mehreren Tabellen wird syntaktisch in QSQL im Wesentlichen auf die gleiche Weise formuliert wie in SQL. Die logische Anfragebedingung, welche aus Teilbedingungen und den logischen Operatoren AND, OR und NOT besteht, wird dementsprechend in der `WHERE`-Klausel einer `SELECT`-Anweisung platziert. Gegenüber SQL können in QSQL zusätzlich Ähnlichkeitsbedingungen mittels des Ähnlichkeitsoperators `~` spezifiziert werden. Beispielhaft soll folgende Anfrage betrachtet werden: „Ermittle alle TV-Geräte, welche verfügbar sind und möglichst einfach zu bedienen sind“. Die Anfrage in QSQL lautet:

```
SELECT name
FROM tv_set
WHERE ( Status = 'V' AND Bedienung ~ 1 ) AND scoreval > 0
ORDER BY scoreval DESC
```

Die zweite Beispielanfrage beinhaltet eine Gewichtung von verschiedenen Teilbedingungen: „Ermittle alle TV-Geräte, welche eine möglichst einfache Bedienung und eine möglichst sehr gute Bildqualität besitzen. Dabei soll der Einfluss der Bewertung für die Bedienung doppelt so hoch sein wie die Bewertung für die Bildqualität“. Entsprechend dieser verbal formulierten Gewichtung werden zwei Gewichte für die entsprechenden Teilbedingungen von 0.5 und 1.0 vereinbart. In QSQL werden Gewichte aus dem Interval 0 und 1 den jeweiligen Teilbedingungen mittels des Schlüsselwortes `WEIGHTED BY` zu geordnet. Die korrespondierende QSQL-Anfrage lautet dementsprechend:

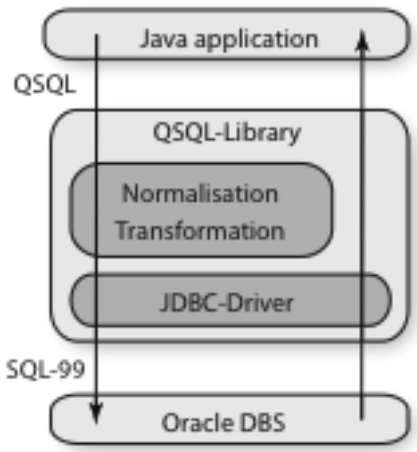
```
SELECT name
FROM tv_set
WHERE ( Bedienung ~ 1 ) WEIGHTED BY 0.5 AND Bildqualitaet ~ 1
```

Das zweite Gewicht von 1.0 muss nicht explizit angegeben werden, da das Gewicht 1.0 der Standardwert für jede Teilbedingung ist.

Die aktuelle Version von QSQL umfasst alle SQL-Standardoperationen wie Selektion von Tupeln, Projektion von Spalten, Verbund, Vereinigung und Durchschnitt von Tabellen und Gruppierung von Tupeln. Objekt-relationale Datentypen und Operationen werden zur Zeit noch nicht unterstützt.

Der Oracle JDBC-Treiber für QSQL

In diesem Abschnitt wird kurz die Benutzung und die Architektur der QSQL-Bibliothek aufgezeigt, welche als JDBC-Treiber implementiert ist. Sie wird in Kombination mit einem Oracle Datenbankmanagementsystem (DBMS) eingesetzt. Innerhalb der QSQL-Bibliothek werden QSQL-Anfragen entgegen genommen, analysiert, normalisiert und in Oracle SQL-99-Anfragen übersetzt. Die so generierten SQL-99-Anfragen werden anschließend an das entsprechende Oracle DBMS gesendet und dort ausgewertet.



Diese Top-Layer-Strategie verhindert eine direkte Manipulation des DBMS und garantiert eine Unabhängigkeit von Applikationen, welche mit Hilfe QSQL erstellt worden sind, von einem speziellen DBMS.

Die QSQL-Bibliothek kann im Vergleich mit dem Standard-JDBC-Treibers völlig transparent benutzt werden, d.h. die Schnittstellenbeschreibung des JDBC-Treibers und der QSQL-Bibliothek sind identisch.

Dadurch können bereits existierende Applikationen und Datenbanken mit einem minimalen Anpassungsaufwand von der erweiterten Auswertungssemantik von QSQL profitieren.

Referenzen:

Lehrack, S. und Schmitt, I. : QSQL: Incorporating Logic-Based Retrieval Conditions into SQL, DASFAA 2010, Seite 429-443 (2010)

Schmitt, I.: QQL: A DB&IR Query Language, The International Journal on Very Large Data Bases, Seite 39-56 (2008)

Kontaktadresse:

Dipl.-Inf. Sebastian Lehrack
LS Datenbank- und Informationssysteme
Walter-Pauer-Str. 2
D-03046 Cottbus

Telefon: +49 (0) 3 55 - 69 34 99
Fax: +49 (0) 3 55 - 69 33 15
E-Mail slehrack@informatik.tu-cottbus.de
Internet: www.dbis.informatik.de