informatiCup 2007 • Aufgabe 3

Datenkonsolidierung mittels Subsumption

Einführung

Relationale Datenbanksysteme (RDBMS) sind die vorherrschende Form zur Speicherung und Anfrage großer Mengen strukturierter Daten. In relationalen Datenbanken werden Daten in Form von Tabellen gespeichert. Eine Zeile einer Tabelle nennt man **Tupel**. Ein Tupel setzt sich aus einer Menge von Attributwerten (einer in jeder Spalte) zusammen. Um Anfragen an die Daten beantworten zu können, stehen dem Nutzer in RDBMS eine Reihe von Operatoren zur Verfügung, etwa die Selektion, die Projektion, der Join, aber auch Mengenoperatoren wie die Vereinigung oder Schnittmenge. Diese Grundlagen sind in jedem Lehrbuch für Datenbanksysteme nachzulesen, etwa in [EN02].

Ist ein bestimmter Attributwert für ein Tupel nicht bekannt, wird anstelle eines Wertes ein **NULL-Wert** eingetragen (dargestellt durch ein $\perp$). Beispielsweise kann eine Kundentabelle eine Spalte für die E-Mail-Adresse eines Kunden vorsehen. Allerdings ist nicht davon auszugehen, dass für jeden Kunden eine E-Mail-Adresse bekannt ist, oder dass gar jeder Kunde überhaupt eine solche besitzt. Der entsprechende NULL-Wert an der Stelle kann also verschiedene Bedeutungen haben: Nicht bekannt, nicht vorhanden, trifft nicht zu, usw.

Bei der Integration von Datenbeständen, z.B. Kundendatenbestände aus mehreren Abteilungen oder Kundendatenbestände aus mehreren Unternehmen bei einer Unternehmensfusion, gilt es, überflüssige Informationen zu eliminieren, also z.B. doppelte Informationen über einen Kunden. Die Schwierigkeit dieser Eliminierung besteht darin, dass die zu eliminierenden Daten oft nicht exakt doppelt vorhanden sind. Vielmehr unterscheiden sich die Einträge: Ein Datensatz kann mehr Informationen über einen bestimmten Kunden haben als ein anderer Datensatz. Oder aber zwei Datensätze können voneinander abweichende, also widersprüchliche Informationen speichern. Das Finden solcher Bestände nennt man Duplikaterkennung (siehe auch http://www.hpi.uni-potsdam.de/fileadmin/hpi/FG_Naumann/publications/2007/40-43_IQR_Dubletten_erkennen.pdf). In dieser Aufgabe soll nur der erste Fall betrachtet werden, also der Fall, dass ein Datensatz reicher an Informationen ist als ein anderen – er "subsumiert" den anderen.

Die **Subsumption** ist eine Operation, die innerhalb einer Tabelle sämtliche subsumierten Tupel entfernt. Ein Tupel wird von einem anderen subsumiert wenn es (i) mehr NULL-Werte als das andere Tupel hat und (ii) alle nicht-NULL-Werte mit den Werten des anderen Tupels übereinstimmen [Ull88, GL94, RPZ04]. Das subsumierte Tupel speichert also die gleichen Informationen wie das subsumierende Tupel, hat allerdings mehr NULL-Werte. Man beachte, dass zwei genau gleichen Tupel sich nicht gegenseitig subsumieren. In vielen Anwendungen kann das subsumierte Tupel gelöscht werden – die Daten werden so konsolidiert. Es verbleiben weniger Tupel aber es werden keine relevanten Informationen gelöscht. Zurzeit gibt es kein kommerzielles Datenbankmanagementsystem, das die Subsumptionsoperation unterstützt – effiziente Algorithmen sind also gefragt.

In der folgenden **Beispieltabelle** mit den drei Attributen A, B und C subsumiert das erste Tupel (t_1) sämtliche anderen Tupel. Es besteht auch eine andere Subsumptionsbeziehung: Tupel t_2 subsu-

miert t_4. Man beachte, dass t_2 und t_3 in keiner Subsumptionsbeziehung stehen.

	A	B	C
t_1	a	b	c
t_2	a	b	$\perp$
t_3	a	$\perp$	c
t_4	$\perp$	b	$\perp$

Aufgabenstellung

Ihre Aufgabe ist es, drei effiziente Algorithmen zu entwerfen und zu implementieren, die überflüssige Informationen aus einer oder mehreren Tabellen entfernen.

1. Gegeben sei eine Tabelle. Entfernen Sie alle exakten Duplikate, also alle Datensätze die genau übereinstimmen.

2. Gegeben sei eine Tabelle. Wie oben, aber entfernen Sie zusätzlich alle subsumierten Datensätze.

3. Gegeben seien zwei Tabellen, die jeweils die gleiche Struktur haben und in denen jeweils keine exakt gleichen oder subsumierten Datensätze gespeichert sind. D.h. die einzelnen Tabellen sind "sauber", ein Datensatz einer Tabelle kann höchstens einen oder mehrere Datensätze der anderen Tabelle subsumieren. Vereinigen Sie die beiden Tabellen und entfernen Sie alle exakten Duplikate und alle subsumierten Datensätze. Hinweis: Die Lösung dieser Teilaufgabe sollte effizienter sein als die der vorigen.

Ein Teil der Aufgabestellung ist der Testentwurf: Entwerfen und generieren Sie selbst geeignete Testtabellen. Speichern Sie Testtabellen jeweils in eine Datei; wählen Sie dabei das csv Format mit dem Semikolon als Trennsymbol einzelner Werte. Die darin enthaltenen tatsächlichen Daten sind irrelevant - wählen Sie z.B. für alle Attribute Integerwerte. Fügen Sie Ihren Daten in den Testtabellen eine zusätzliche Spalte zur Identifikation der Tupel hinzu (Integer, aufsteigend vergeben). Diese Spalte sollte die Entscheidung ob ein Tupel ein anderes subsumiert natürlich nicht beeinflussen.

Als Output schreiben Sie zwei Dateien (vergeben Sie geeignete Dateinamen). Die erste Datei enthält die konsolidierte Tabelle; die zweite Datei enthält eine Liste mit den Tupelidentifikatoren gelöschter Tupel.

In der zweiten Runde des Wettbewerbs soll keine elaborierte graphische Schnittstelle (GUI) entworfen werden. Vielmehr soll die Evaluierung der Algorithmen geeignet dokumentiert werden. Variieren Sie bei allen drei Teilaufgaben Faktoren wie beispielsweise Anzahl der Datensätze, Anzahl der Tabellenspalten, Anteil exakter Duplikate, Anteil subsumierter Duplikate, Anzahl der durch einen Datensatz subsumierte andere Datensätze, usw. Stellen Sie die Ergebnisse graphisch dar (in einem Dokument, nicht unbedingt am Bildschirm) und interpretieren Sie sie. Testen Sie auch die Speichergrenzen Ihres Algorithmus – welche Datenmenge können Sie eben noch verarbeiten?

Literatur

[EN02] Ramez Elmasri and Shamkant B. Navathe. *Grundlagen von Datenbanksystemen.* Pearson Studium, München, 2002.

[GL94] César A. Galindo-Legaria. Outerjoins as disjunctions. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD)*, pages 348–358, Minneapolis, Minnesota, May 1994.

[RPZ04] Jun Rao, Hamid Pirahesh, and Calisto Zuzarte. Canonical abstraction for outerjoin optimization. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD)*, pages 671–682, 2004.

[Ull88] Jeffrey D. Ullman. *Principles of Database and Knowledge-Base Systems, Volume I.* Computer Science Press, 1988.