Datenbanksysteme WS 2016/17
Prof. Dr.-Ing. Sebastian Michel
BSc. Peter Brucker
TU Kaiserslautern, FB Informatik – Lehrgebiet Informationssysteme
http://dbis.informatik.uni-kl.de
Übungsblatt 6: Ausgabe 06.12.2016, Abgabe 12.12.2016

Aufgabe 1: Min-Hashing (1 P.)

In dieser Aufgabe geht es darum, Kunden auf Grund ihrer bestellten Teile zu vergleichen. Kunden werden im TPC-H-Schema etwa durch die Spalte `c_custkey` in der Tablle `customer` identifiziert, Teile durch die Spalte `l_partkey` in der Tabelle `lineitem`. Die Information "Welcher Kunde hat was bestellt?" kann über einen Join mit der Tabelle `orders` herausgefunden werden. Die Ähnlichkeit selbst wird mittels Jaccard-Koeffizienten beziehungsweise Min-Hashing gemessen.

(a) Sichten erstellen

(i) Erstellen Sie eine materialisierte Sicht[1] im TPC-H Schema `cust_parts(custkey, partkey)`. Diese Sicht stellt dar, welche Kunden welche Teile gekauft haben.

(ii) Erstellen Sie eine materialisierte Sicht `cust_part_hashes(custkey, h1, h2, h3, h4)`, die auf der vorherigen aufbaut. Diese Sicht soll zu jedem custkey die Minima der Hashfunktionen h_1 bis h_4 angewendet auf die jeweiligen partkeys beinhalten. Die Hashfunktionen sind $h_1(x) = x \bmod 5531$, $h_2(x) = x \bmod 6173$, $h_3(x) = x \bmod 6803$, $h_4(x) = x \bmod 7919$.

(b) Anfragen auf die Sichten stellen

(i) Ein neuer Kunde K interessiert sich für die Teile mit partkey = 91103, 24814, 32395. Als Ähnlichkeitsmaß zwischen Kunden wählen Sie die Anzahl der bestellten Teile. Schreiben Sie eine SQL-Abfrage, die alle Kunden aus der Datenbank liefert, die eine geschätzte Ähnlichkeit echt größer als 0 mit K haben.

(ii) Der Kunde mit custkey = 494 will beraten werden. Mit welchen anderen Kunden hat er eine geschätzte Ähnlichkeit ≥ 0.5?

(iii) Welche Anfrage liefert alle Kundenpaare, die mindestens eine geschätzte Ähnlichkeit von 0.5 haben? Und welche Indexe können Sie anlegen, um diese Anfrage zu unterstützen? Vergleichen Sie Ihre Annahmen mit der Ausgabe von PostgreSQLs EXPLAIN.

(c) Wir wollen nun den Fehler unserer Schätzung für den Kunden mit custkey = 494 bewerten. Legen Sie dazu eine neue Tabelle an, in der Sie die custkey-Nummern aller anderen Kunden zusammen mit den geschätzten Ähnlichkeitswerten basierend auf den vier in (a.ii) genannten Hashfunktionen ablegen. In weiteren Spalten sollen nun der korrekte Jaccard-Koeffizient basierend auf den jeweils bestellten Teilenummern sowie der Fehler gespeichert werden. Wie groß ist der durchschnittliche Fehler?

Hinweise: Die Berechnungen der letzten Teilaufgabe können schnell sehr groß werden und damit auch sehr lange dauern. Machen Sie sich daher vorher Gedanken, wie ihre Anfrage aussehen könnte. Wenn Sie wissen wollen, ob die Anfrage syntaktisch richtig ist, reicht es auch, sich den Ausführungsplan ausrechnen zu lassen. Falls die Anfragen auf Ihrem System zu lange dauern, schränken Sie die materialisierten Sichten darauf ein, nur einen Teil der Kundendaten zu benutzen.

Aufgabe 2: kd-Baum (1 P.)

(a) Fügen Sie folgende Werte in der angegebenen Reihenfolge in einen zunächst leeren kd-Baum mit $k = 3$ ein:

[1] `create materialized view cust_part_hashes(custkey, h1, h2, h3, h4) AS ....` Siehe auch `https://www.postgresql.org/docs/9.6/static/sql-creatematerializedview.html`

1. (15, 94, 42)
2. (42, 50, 24)
3. (8, 49, 32)
4. (13, 92, 18)
5. (20, 43, 75)
6. (3, 29, 85)
7. (28, 79, 66)
8. (6, 14, 95)
9. (9, 88, 22)
10. (16, 93, 23)
11. (7, 55, 60)
12. (12, 60, 29)

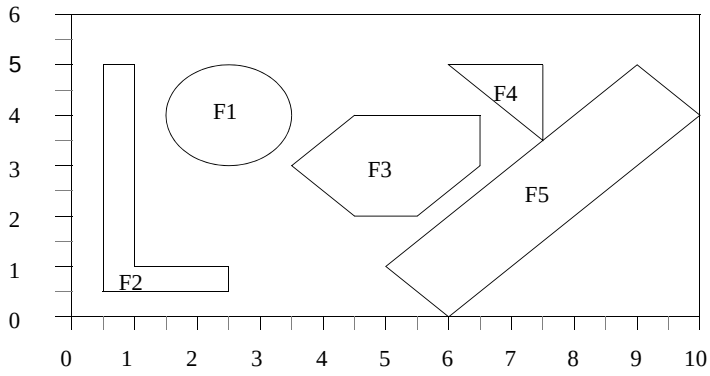
Zeichnen Sie das Ergebnis als Baumstruktur analog zur Darstellung im Skript. Machen Sie deutlich, welche Werte zum Splitten benutzt werden.

(b) Adaptiver kd-Baum

Geben Sie einen Algorithmus an, der aus einer gegebenen Liste von Koordinaten in k Dimensionen einen kd-Baum konstruiert, in dessen Blättern höchstens zehn Datenpunkte sein dürfen. Die gewählte Dimension für jeden Split soll von der Varianz abhängen, die Splitkoordinate vom Durchschnitt der entsprechenden Werte der relevanten Punkte.

Aufgabe 3: R-Baum (1 P.)

(a) Einfügen und Suchen im R-Baum: Gegeben sind die wie folgt angeordneten zweidimensionalen Objekte.



(i) Speichern Sie die Objekte in einen R-Baum in der Reihenfolge F1, F2, F3, F4, F5. In einen Knoten passen zwischen 1 und 2 Einträge.

(ii) Verwenden Sie Ihren R-Baum, um alle Objekte zu finden, die den Punkt (6, 3.5) enthalten.

(iii) Verwenden Sie Ihren R-Baum, um alle Objekte zu finden, die sich vollständig im Rechteck Q, welches durch (3.5, 0) und (8, 5.5) definiert ist, befinden.

(iv) Verwenden Sie Ihren R-Baum, um alle Objekte zu finden, die sich mit dem Rechteck Q': (2,2) (4,4) überlappen.

(b) NN-Suche im R-Baum

(i) Erläutern/beweisen Sie die Korrektheit der Pruning-Strategien 1 und 3 aus der Vorlesung.

(ii) Geben Sie ein Beispiel für $d = 2$ an, für das die MINDIST-Sortierung der Priority-Queue günstiger ist als eine MINMAXDIST-Sortierung.