Datenbanksysteme WS 2017/18
Prof. Dr.-Ing. Sebastian Michel
Manuel Hoffmann, Marc Dörr
TU Kaiserslautern, FB Informatik – Lehrgebiet Informationssysteme
http://dbis.informatik.uni-kl.de
Übungsblatt 1: Ausgabe 26.10.2017, Abgabe 06.11.2017 16:00 MEZ

Aufgabe 1: Analyse Seitenreferenzstrings (1 P.)

a) Gegeben folgender Seitenreferenzstring:

⟨ (T1,A) (T1,A) (T1,B) (T1,A) (T2,A) (T2,B) (T1,A) (T1,C) (T2,C)
(T2,D) (T2,E) (T1,G) (T1,A) (T2,F) (T2,C) (T1,F) (T1,G) ⟩

Berechnen Sie Sequentialität und Lokalität. Für Lokalität verwenden Sie die Working-Set-Methode und betrachten Sie den Durchschnitt über den beiden Transaktionen. Nehmen Sie eine Fenstergröße von $\tau = 3$ an.

b) Gegeben folgende Stacktiefenverteilung:

Stacktiefe	**Häufigkeit**	**Stacktiefe**	**Häufigkeit**
1	412	7	25
2	234	8	21
3	152	9	13
4	134	10	12
5	43	11	5
6	36	12	2

Beantworten Sie folgende Fragen:

i) Wir möchten nicht mehr als 5% Cache-Misses haben. Wie groß muss der Puffer sein?

ii) Wie viele Cache-Hits (in Prozent) gibt es bei einer Puffergröße von 4?

iii) Wie viele Cache-Misses werden vermieden, wenn der Puffer von 4 auf 10 vergrößert wird?

Aufgabe 2: Seitenersetzungsstrategien (1 P.)

Gegeben folgende Sequenz von Seitenzugriffen:

D, C, A, D, D, A, B, D, B, B, C, A, C, A, C, A, C

a) Berechnen Sie für diese Sequenz bei einer Puffergröße von 3 die Anzahl von Zugriffen auf Seiten, die sich zu dem Zeitpunkt nicht im Puffer befinden, für die Strategien CLOCK, LRU-k mit $k = 2$ und GCLOCK.
Gehen Sie für GCLOCK von einem Initialwert $E_i = 3$ für jede Seite und einem Inkrementwert bei Referenz $W_i = 1$ aus.

b) Welche Ersetzungen finden für diese Sequenz bei einer optimalen Strategie, die zu jedem Zeitpunkt bereits alle zukünftigen Seitenzugriffe kennt[1], statt?

c) Finden Sie ein Beispiel für eine Seitenzugriffssequenz, bei dem die Strategie FIFO echt weniger Cachemisses hat als LRU, und ein Beispiel bei dem FIFO echt weniger Hits hat als LRU. Nehmen Sie eine Puffergröße von 2 an.

[1] Solch eine Strategie (clairvoyant) kann natürlich nicht implementiert werden. Allerdings wird diese Art der Betrachtung häufig benutzt, um die Qualität anderer Verfahren zu bewerten.

Aufgabe 3: Performance-Überlegungen mit PostgreSQL und pgAdmin (1 P.)

In diesem und den folgenden Übungsblättern werden wir das relationale Datenbanksystem *PostgreSQL* benutzen. PostgreSQL ist unter einer Open-Source-Lizenz und somit kostenlos verfügbar, etwa unter `http://www.postgresql.org/` als installierbare Pakete oder für Linux-Distributionen über die gängigen Paketmanager (`apt-get, yum, pacman, ...`). Installieren Sie sich PostgreSQL (Version 10 oder höher) auf einem Ihrem System angemessenem Weg.

Zusätzlich gibt es mit *pgAdmin* eine graphische Benutzeroberfläche, die die Benutzung und Administration von PostgreSQL vereinfacht. Installieren Sie auch dieses Programm (`http://www.pgadmin.org/` oder Paketmanager). Sollten Sie wider Erwarten Probleme mit der Installation haben und die Videos im OLAT nicht helfen, melden Sie sich frühzeitig(!) per Email unter Angabe Ihres SCI-Benutzernamens an mhoffmann@cs.uni-kl.de, damit Sie einen Zugang zu einer PostgreSQL-Installation im SCI bekommen.

Laden Sie die beiden auf der Website verlinkten Datensätze in die Datenbank. Nutzen Sie die Gelegenheit, um Ihre SQL-Kenntnisse aufzufrischen und ein paar Anfragen gegen die Datenbank stellen. Führen Sie als erstes die Anfrage `VACUUM ANALYZE` aus. Diese Anfrage hat zwar kein Ergebnis, führt aber dazu sofort Statistiken über die importierten Tabellen erstellt werden.

Effekte von Indizes

Sie können den Anfrageplan, der vom Datenbanksystem generiert wird, ausgeben lassen, indem Sie im Query-Tool “Explain” (oder “Analysieren”) wählen. Dazu brauchen Sie die eigentlichen Ergebnisse nicht ausgeben lassen.

a) Welcher Anfrageplan wird für folgende Anfrage generiert? Beschreiben Sie den Plan so gut es geht. Wenn Sie mit der Maus über einen Element im Anfrageplan hovern, wird ein Tooltip mit weiteren Informationen angezeigt.

```
SELECT * FROM lineitem WHERE l_suppkey < 10
```

b) Was ändert sich am Anfrageplan, wenn Sie zuerst den folgenden Index erzeugen lassen?

```
CREATE INDEX index1 ON lineitem(l_suppkey);
```

c) Die SELECT-Anfrage aus Aufgabenteil (a) liefert alle Einträge mit `l_suppkey` $< k = 10$. Ab welcher Größe von k ändert sich der Anfrageplan? Wie verhält die Anzahl von Tupeln, die mit diesem k zurückgegeben würden, zur Gesamtzahl der Einträge der `lineitem`-Tabelle?

d) Welcher Plan wird für folgende Query erstellt?

```
SELECT o_custkey, AVG(o_totalprice) FROM orders GROUP BY o_custkey;
```

Wie ändert er sich, wenn Sie folgenden Index erstellen?

```
CREATE INDEX index2 ON orders(o_custkey, o_totalprice)
```

Aufgabe 4: Indextuning (1 P.)

Im Qualitäts-Universitäts-Informations-System *QUIS*[2] werden folgende Relationen verwendet:

Studenten: [MatNR, Name, FBNR, Fachsemester]
Professoren: [PNR, Name, FBNR]
Fachbereich: [FBNR, Name]
Prüfung: [ID, Datum, Semester, PNR]
teilnehmen: [ID, MatNR, Note]

Es sind jeweils bereits Primärindexe angelegt und Fremdschlüssel sind entsprechend der Namen gesetzt (z.B. ist *Studenten.FBNR* Fremdschlüssel auf *Fachbereich.FBNR*).

Da die Stromrechnung der Universität immer höher wird, hat der Kanzler beschlossen, dass dieses System effizienter werden muss. Dazu wurden die Anfragen, die regelmäßig an das QUIS gestellt werden analysiert und folgende als häufig herausgefunden:

- A1: Die Liste an Teilnehmern einer bestimmten Prüfung zusammen mit den Namen der jeweiligen Fachbereiche.
- A2: Ein Histogramm für einen gegebenen Fachbereich, das aussagt, wie viele Studenten im wievielten Semester sind.
- A3: Das gleiche Histogramm, aber für die gesamte Universität.
- A4: Alle Prüfungen eines Professors in einem bestimmten Prüfungszeitraum.
- A5: Die Notenübersicht für einen Studenten.

Welche Indexe sollten angelegt werden, um diese Anfragen effizienter zu beantworten? Diskutieren Sie Sinn und Zweck der Indexe, evtl. können Indexe gleich in mehreren Anfragen genutzt werden. Bei welchem dieser Indexe bietet sich als Alternative zum B+Baum ein Hash-Index an?

[2] Ähnlichkeiten zu real existierenden System sind reiner Zufall