Aufgabenblatt 3
Anfrageausführung

- Abgabetermin: **Dienstag, 17.12.13 (23:59 Uhr)**
- Zur Prüfungszulassung muss ein Aufgabenblatt mit mind. 25% der Punkte bewertet werden und alle weiteren Aufgabenblätter mit mindestens 50% der Punkte.
- Die Aufgaben sollen in Zweiergruppen bearbeitet werden.
- Abgabesytem unter
  `http://www.dcl.hpi.uni-potsdam.de/submit`
  - ausschließlich pdf-Dateien
  - eine Datei pro Aufgabe namens Aufgabe-<aufgabenNr>.pdf
  - *jedes* Blatt beschriftet mit Namen

Aufgabe 1: Ausführungsstrategien

Betrachte die Relation $R(a, b, c, d)$, die 1 Million Tupel enthält. Nimm an, dass das Attribut a Schlüsselkandidat für R ist und die Werte von a zwischen 0 und 999.999 liegen. Jeder Block der Relation R fasst 10 Tupel. Außerdem sind die Tupel in R nicht sortiert.

Benenne für jede der folgenden Anfragen diejenige Ausführungsstrategie, die die wahrscheinlich wenigsten I/Os für die Bearbeitung der Anfrage benötigt. Begründe deine Antwort durch Angabe der I/Os für *alle* Ausführungsstrategien. Die beiden aufgeführten Indexe existieren bereits und müssen nicht für eine Anfrage neu angelegt werden. Sie liegen allerdings auf der Festplatte, so dass für genutzte Index-Strukturen ebenfalls I/O-Kosten anfallen. Falls nötig, triff geeignete Annahmen bzgl. benötigter Parameter und deren Werte!

Die zu betrachtenden Ausführungsstrategien sind:

- Scannen der kompletten Relation R.
- Nutzen eines B+-Baum-Indexes für $R.a$.
- Nutzen eines Hash-Indexes für $R.a$.

Die Anfragen sind:

a) Finde alle Tupel aus R. **3 P**

b) Finde alle Tupel aus R mit $a < 50$. **3 P**

c) Finde alle Tupel aus R mit $a = 50$. **3 P**

d) Finde alle Tupel aus R mit $a > 50$ und $a < 100$. **3 P**

Aufgabe 2: Sort-Merge Join-Algorithmus

Gegeben sind zwei Relationen $R(\underline{A}, B)$ und $S(\underline{A}, C)$. Die Tupel in R umfassen 2^{13} = 8.192 Blöcke und die Tupel in S umfassen 2^{10} = 1.024 Blöcke. Für die Berechnung von $R \bowtie S$ können insgesamt 2^7 = 128 Blöcke des Hauptspeichers genutzt werden.

a) Wie viele sortierte Teillisten entstehen in der ersten Phase des Two-Phase Multiway Merge-Sort (TPMMS) bei der Sortierung von R? Wie viele für S? **2 P**

b) Wie viele Blöcke müsste eine der beiden Eingaberelationen *mindestens* umfassen, damit die Sortierung mittels des TPMMS *nicht* mehr möglich ist? **2 P**

c) Notiere kurz die Ausführungsschritte des Sort-Merge Join-Algorithmus für die Berechnung von $R \bowtie S$. Notiere zusätzlich für jeden Teilschritt die Anzahl an I/O-Operationen.
Hinweis : Beachte, dass das Attribut A in beiden Relationen Primärschlüssel ist. **4 P**

d) Angenommen die Blockanzahl von S vergrößert sich auf $2^{13} + 1 = 8.193$ Blöcke. Welches Problem ergibt sich dann bei der Berechnung von $R \bowtie S$ unter Verwendung des Sort-Merge Join-Algorithmus? Begründe deine Antwort. **2 P**

Aufgabe 3: Hashjoin

Gegeben seien zwei Relationen R und S. Die Tupel in R umfassen insgesamt 500 Blöcke und die Tupel in S belegen insgesamt 400 Blöcke. Für die Berechnung von $R \bowtie S$ können insgesamt 41 Blöcke des Hauptspeichers genutzt werden. Wir nehmen an, dass unabhängig von der Anzahl bereitgestellter Buckets gleich viele Tupel in jeden Bucket gehashed werden. In jeden erstellten Bucket speichern wir die vollständigen R- bzw. S-Tupel (also nicht Schlüssel-Pointer-Paare), um in der Join-Phase direkt die Ergebnistupel ausgeben zu können.

a) Um I/O-Kosten zu sparen, soll während der Join-Phase stets 1 Bucket komplett im Speicher gehalten werden können (One-pass Join). Ermittle unter dieser Voraussetzung die minimale und die maximale Anzahl k an Buckets, die zur Durchführung des Hashjoins benötigt werden. **6 P**

b) Berechne die Anzahl an I/O-Operationen für das in Teilaufgabe a) beschriebene Vorgehen zur Bestimmung von $R \bowtie S$. Verwende für deine Rechnung k = 20 Buckets. **3 P**

Aufgabe 4: Wahl der Join-Implementierung

Gegeben ist folgende SQL-Anfrage, die nach allen Schauspielerinnen des Films 'King Kong' sucht:

```
SELECT *
FROM SpieltMit M, Schauspieler S
WHERE M.name = S.name
AND titel='King Kong' AND geschlecht='w';
```

Die angefragten Relationen sind:

```
SpieltMit(titel, jahr, name)
Schauspieler(name, adresse, geschlecht, geburtsdatum)
```

In Relationaler Algebra lässt sich die Anfrage folgendermaßen darstellen:

$$\sigma_{\text{titel}='KingKong'}(\text{SpieltMit}) \bowtie \sigma_{\text{geschlecht}='w'}(\text{Schauspieler})$$

Nimm an, dass beide Relationen jeweils etwa 10.000 Tupel umfassen und es einen Primärindex auf Schauspieler.name gibt.

Vergleiche die I/O-Kosten eines Index-basierten, eines Sort-basierten und eines Hash-basierten Joins für die gegebene Anfrage. Falls nötig, triff geeignete Annahmen bzgl. der Parameter der einzelnen Join-Implementierungen (z.B. Blockgrößen). Wähle für jede Join-Variante außerdem die jeweils optimale Ausführungsreihenfolge von Selektionen und Join. Welche Algorithmus-Klasse sollte die Datenbank hier wählen?

4 P