TU München, Fakultät für Informatik
Lehrstuhl III: Datenbanksysteme
Prof. Alfons Kemper, Ph.D.

Übung zur Vorlesung *Einsatz und Realisierung von Datenbanksystemen* im SoSe19

Maximilian {Bandle, Schüle} (i3erdb@in.tum.de)
http://db.in.tum.de/teaching/ss19/impldb/

Blatt Nr. 09

Hausaufgabe 1

Schätzen sie die Anzahl der Cache Misses die entstehen, wenn man 1001 32-Bit Integer Werte (0-1000) in aufeinanderfolgender Reihenfolge in einen ART Baum einfügt. Wäre ein B+ Baum besser oder schlechter? Bei den Baumknoten müssen die Header nicht berücksichtigt werden, Pointer haben eine Größe von 64 Bit.

Hausaufgabe 2

Sie sollen für die Alexander-Maximilians-Universität (AMU) ein Hauptspeicherdatenbanksystem optimieren. In dem System sind die Daten aller Studenten gespeichert. Schätzen Sie für jede der untenstehenden Anfragen einzeln, ob ein Row- oder Column-Store besser geeignet ist.

Relationen

Studenten: MatrNr (8 Byte), Name (48 Byte), Studiengang (4 Byte), Semester (4 Byte)

MatrNr ist der Primärschlüssel der indiziert ist.

Anfragen:

1. select * from Studenten;
2. select Semester, count(*) from Studenten group by Semester;
3. select Name, Studiengang, Semester from Studenten where MatrNr = 42;
4. select Studiengang from Studenten where MatrNr = 42;
5. select * from Studenten where Semester < 5;
6. select * from Studenten where Semester = 25;
7. insert into studenten values(4242, Max Meyer, Info, 7);

Gruppenaufgabe 3

Beschäftigen wir uns mit *Multi-Version Concurrency Control* am Beispiel unserer verfügbaren Ärzte („Doctors on call/duty“), in dem wir sicherstellen wollen, dass immer mindestens ein Arzt verfügbar ist.

Uns stehen drei Operationen zur Verfügung, $\sum$ zählt alle verfügbaren Ärzte, $(X)++$ ändert Xs Status in verfügbar, $(X)--$ zählt alle verfügbaren Ärzte und ändert Xs Status auf nicht verfügbar, wenn mindestens ein Arzt noch anwesend ist.

1. Welche Bedingungen gelten für die Zeitstempel?

Name	verfügbar	Versionsvektor
House	ja	-
Green	ja	-
Brinkmann	ja	-

Abbildung 1: Hauptspeicher Column-Store

TID	Startzeit	Commitzeit	Aktion
Ta	$T0$	-	$\sum$
Tb	$T2$	-	$(Green) - -$
Tc	$T3$	-	$\sum$
Td	$T5$	-	$\sum$

Abbildung 2: Transaktionen (bereits committete gekennzeichnet durch eine Commitzeit)

2. Green möchte zum Zeitpunkt $T2$ seinen Feierabend antreten. Vervollständigen Sie Tabelle 2 und legen Sie einen geeigneten Undo-Puffer (Zeitstempel, Attribut, Undo-Image) an. Wann muss Tb committen, damit Td bereits die Änderung von Tb liest? Was lesen Ta und Tb?
3. Brinkmann und House wollen zeitgleich den Feierabend antreten. House startet bei $T8$, Brinkmann bei $T9$. Wer darf gehen? Wie sorgt *Precision Locking* dafür, dass nur ein Arzt das Krankenhaus verlässt? Vervollständigen Sie die Einträge.

Hausaufgabe 4

Gegeben seien die folgenden Anfragen:

T1: insert into foo (select Note from Noten where MatrNr=12345)

T2: insert into bar (select count(*) from Noten where Note<1.5)

T3: insert into Noten(MatrNr,Note) values (54321, 3.0)

T4: update Noten set Note=1.4 where MatrNr=32154

T5: insert into Noten(MatrNr,Note) values (54321, 1.3)

T6: update Noten set Note=1.6 where MatrNr=12345

Analysieren Sie, ob die folgenden Historien unter dem MVCC Model, wie in der Vorlesung vorgestellt, auftreten können. Jede Historie steht für sich selbst und startet jeweils von einem ursprünglichen Datenzustand. Die Buchstaben innerhalb der Klammer entsprechen dabei jeweils den Tupeln auf die zugegriffen wird. Wenn in T2 z.B. drei Werte das 'Prädikat Note<1.5' erfüllen, gäbe es entsprechend drei $r(...)$ Einträge auf die jeweiligen Tupel.

H1 (T1 und T3): $bot_1, r_1(A), bot_3, w_3(B), w_1(C), commit_1, commit_3$

H2 (T2 und T3): $bot_2, r_2(A), bot_3, w_3(B), r_2(C), w_2(D), commit_2, commit_3$

H8 (T2 und T3): $bot_2, r_2(A), bot_3, w_3(B), r_2(C), commit_3, w_2(D), commit_2$

H3 (T2 und T4): $bot_2, r_2(A), r_2(B), bot_4, r_4(B), w_4(B), r_2(C), w_2(D), commit_2, commit_4$

H5 (T2 und T4): $bot_2, r_2(A), bot_4, r_4(B), w_4(B), r_2(C), commit_4, w_2(D), commit_2$

H4 (T1 und T6): $bot_1, r_1(B), bot_6, r_6(B), w_6(B), w_1(C), commit_1, commit_6$

H6 (T1 und T6): $bot_1, r_1(B), bot_6, r_6(B), w_6(B), commit_6, w_1(C), commit_1$

H7 (T2 und T5): $bot_2, r_2(A), bot_5, w_5(D), commit_5, r_2(D), w_2(E), commit_2$

H9 (T2 und T5): $bot_2, r_2(A), bot_5, w_5(B), r_2(C), commit_5, w_2(D), commit_2$