

Name

CWID

Quiz 2

April 15th, 2013

CS525 - Advanced Database Organization Solutions

Please leave this empty!

1 2 3

Sum

Instructions

- Things that you are allowed to use
 - Textbook
 - Printed lecture notes
- Things that you are **not** allowed to use
 - Personal notes
- The quiz is **45** minutes long
- Multiple choice questions are graded in the following way: You get points for correct answers and points subtracted for wrong answers. The minimum points for each questions is **0**. For example, assume there is a multiple choice question with 6 answers - each may be correct or incorrect - and each answer gives 1 point. If you answer 3 questions correct and 3 incorrect you get 0 points. If you answer 4 questions correct and 2 incorrect you get 2 points. ...
- For your convenience the number of points for each part and questions are shown in parenthesis.
- There are 3 parts in this quiz
 1. Physical Optimization
 2. Schedules
 3. ARIES

Part 1 Physical Optimization (Total: 40 Points)

Consider the following relations $R(A, B)$, $S(C)$, $T(D, E)$ with $S = \frac{1}{100}$ (100 tuples fit on each page). The sizes and value distributions are:

$$N(R) = 10000$$

$$V(R, A) = 10000$$

$$V(R, B) = 100$$

$$N(S) = 300$$

$$V(S, C) = 300$$

$$N(T) = 5000$$

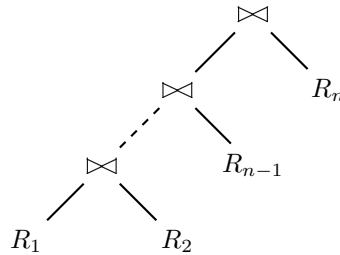
$$V(T, D) = 5000$$

$$V(T, E) = 600$$

Question 1.1 Greedy Enumeration (40 Points)

Use the greedy join enumeration algorithm to find the cheapest plan for the join $R \bowtie_{B=C} S \bowtie_{C=D} T$. Assume that **nested-loop** is the only available join implementation with the left input being the “outer” (for each tuple from the outer we have to scan the whole inner relation). Furthermore, there are no indices defined on any of the relations (that is you have to use **sequential scan** for each of the relations). As a cost model consider the **total number of I/O operations**. For example, if you join two relations with 5,000 and 10,000 tuples with $S = \frac{1}{10}$, where the 5,000 tuple relation is the outer, then the cost would be 5,000,000. Assume that the system supports pipelining for the outer input of a join. That is if you join the result of a join with a relation where the join result is the outer, then there is no I/O cost for scanning the outer. **Hint: You will have to estimate the size of intermediate results. Use the estimation based on the number of values and not the one based on the size of the domain. Use the assumption that the number of values in a join attribute of a join result is the minimum of the number of values in the join attribute of each input.**

Write down the state after each iteration of the algorithm using the following notation. Write $((R_1, R_2), \dots, R_{n-1}), R_n)^{C, S}$ to denote a plan as shown below with I/O cost C and result size S .



Solution

Initialization:

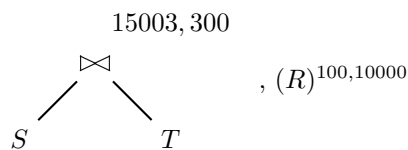
$$(R)^{100,10000}, (S)^{3,300}, (T)^{50,5000}$$

n = 1:

Here we have 6 different options how to join two of the plans from the initialization:

$$(R, S)^{30100,10000}, (R, T)^{500100,10000}, (S, R)^{30003,10000}, (S, T)^{15003,300}, (T, R)^{500050,10000}, (T, S)^{15050,300}$$

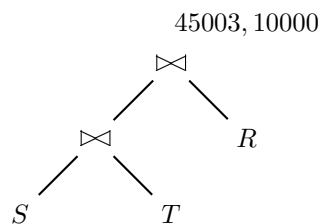
As an example take the join (R, S) . Here R is the outer and S is the inner. Using the formula from class the estimated result size is $\frac{N(R) \cdot N(S)}{\max(V(R,B), V(S,C))} = \frac{10000 \cdot 300}{300} = 10000$. The cost is computed as: For each tuple from R ($N(R)$) we have to scan S once (3 I/Os). Thus, the cost is $B(R) + N(R) \cdot B(S) = 100 + 10000 \cdot 3 = 30100$ I/Os.



n = 2:

Now we need to consider two join options:

$$(R, (S, T))^{45103,10000}, ((S, T), R)^{45003,10000}$$



Part 2 Schedules (Total: 20 Points)

Question 2.1 Schedule Classes (12 Points)

Indicate which of the following schedules belong to which class. Recall transaction operations are modelled as follows:

$w_1(A)$ transaction 1 wrote item A
 $r_1(A)$ transaction 1 read item A
 c_1 transaction 1 commits
 a_1 transaction 1 aborts

$S_1 = r_1(A), w_2(A), r_1(B), c_1, w_3(B), r_3(B), w_3(A), c_3, r_2(C), c_2$

$S_2 = r_1(A), w_2(B), r_1(B), c_1, c_2$

$S_3 = r_1(A), w_2(B), c_2, r_1(B), w_1(B), c_1$

$S_4 = w_1(A), w_2(A), c_2, w_1(A), c_1$

- ☒ S_1 is recoverable
- ☒ S_1 is cascade-less
- ☐ S_1 is strict
- ☐ S_2 is recoverable
- ☐ S_2 is cascade-less
- ☐ S_2 is strict
- ☒ S_3 is recoverable
- ☒ S_3 is cascade-less
- ☒ S_3 is strict
- ☒ S_4 is recoverable
- ☒ S_4 is cascade-less
- ☐ S_4 is strict

Question 2.2 Create a Strict Schedule (8 Points)

Consider the following set of transactions:

$$T_1 = r_1(A), w_1(A), c_1$$

$$T_2 = r_2(B), r_2(A), w_2(B), w_2(A), c_2$$

$$T_3 = r_3(B), w_3(B)$$

1. Write a strict history involving these three transactions.

Solution

Several solutions are correct. For example,

$$S = r_1(a), w_1(A), c_1, r_2(B), r_2(A), w_2(B), w_2(A), c_2, r_3(C), w_3(C)$$

In a correct solution, if one transaction T_i writes an item, then the others cannot read nor write the same item until T_i commits.

Part 3 ARIES (Total: 30 Points)

Question 3.1 Transaction Rollback (12 Points)

Consider the state of the log shown below. For simplicity we do not show the actual undo/redo actions for updates, but instead show only the affected page. Assume that transaction T_1 aborts. Write down the new entries that would be added to the log during the rollback of T_1 .

Log					
LSN	Type	TID	PrevLSN	UndoNxtLSN	Data
1	begin	1	-	-	-
2	update	1	1	-	Page 1
3	begin	2	-	-	-
4	update	2	3	-	Page 1
5	update	1	2	-	Page 2
6	update	1	5	-	Page 2
7	update	2	4	-	Page 4
8	commit	2	7	-	-
9	update	1	6	-	Page 3

Solution

LSN	Type	TID	PrevLSN	UndoNxtLSN	Data
10	CLR	1	-	6	Page 3
11	CLR	1	-	5	Page 2
12	CLR	1	-	2	Page 2
13	CLR	1	-	1	Page 1

Question 3.2 Recovery (18 Points)

Consider the state of the log and pages on disk shown below. For simplicity we do not show the actual undo/redo actions for updates, but instead show only the affected page. Assume a crash occurred after the last log entry. Answer the following questions:

1. **Analysis:** Write down the result of the analysis phase (RedoLSN, Transaction Table, Dirty Page Table)
2. **Redo:** Which pages will be loaded from disk during redo? Which pages will be modified during redo?
3. **Undo:** Write down the additional log entries that will be written during undo.

Log					
LSN	Type	TID	PrevLSN	UndoNxtLSN	Data
1	begin	1	-	-	-
2	begin	2	-	-	-
3	update	2	2	-	Page 3
4	begin	3	-	-	-
5	begin	4	-	-	-
6	commit	2	-	-	-
7	begin_cp	-	-	-	-
8	update	1	1	-	Page 4
9	update	3	4	-	Page 5
10	end_cp	-	-	-	Transaction Table: $\langle T_1, 1 \rangle, \langle T_3, 4 \rangle, \langle T_4, 5 \rangle$, Dirty Page Table: $\langle 3, 3 \rangle$
11	update	3	9	-	Page 3
12	commit	3	11	-	-
13	begin	5	-	-	-

Disk	
PageID	PageLSN
3	0
4	8
5	0

Solution

(1):

RedoLSN: 3

Transaction Table: $\langle T_1, 8 \rangle, \langle T_4, 5 \rangle, \langle T_5, 13 \rangle$

Dirty Page Table: $\langle 3, 3 \rangle, \langle 4, 8 \rangle, \langle 5, 9 \rangle$

(2):

All pages (3,4,5) have to be loaded from disk.

Only pages 3 and 5 will be modified, for page 4 the update from log entry 8 is already reflected on disk.

(3):

Transaction T_1, T_4, T_5 have to be rolled back. Since T_1 is the only transaction that executed an update. The CLR written during undoing this update is shown below.

LSN	Type	TID	PrevLSN	UndoNxtLSN	Data
14	CLR	1	-	1	Page 4

