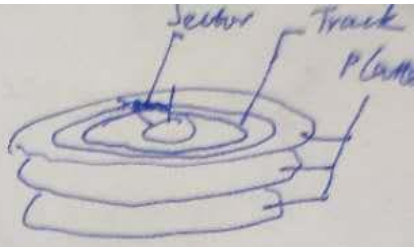


Problem 1 (15 points): Disk Performance

Consider the Megatron 747 disk with the following properties:

- There are four platters providing eight surfaces.
- There are $2^{13} = 8192$, tracks per surface.
- There are (on average) $2^8 = 256$ sectors per track.
- There are $2^9 = 512$ bytes per sector.
- The disk rotates at 3840 RPM.
- The block size is $2^{12} = 4096$ bytes
- Assume 10% of each track is used as overhead.
- The time it takes the head to move n tracks is $(1 + n/500)$ milliseconds.

Calculate the following parameters.

1) The total capacity of the disk. (2 points)

8*8192*256*512=8G $8 \times 2^{13} \times 2^8 \times 2^9 = 8 \times 2^{30} = 8GB$ Byte.

2) The average seek time. (2 points)

1+1/3*1/500 (8192-1) =6.46ms $1 + \frac{1}{3} \times \frac{8192-1}{500}$

3) The average rotational latency. (2 points)

1/2*60/3840*1000=7.81ms $\frac{1}{2} \times \frac{60}{3840} \times 10^3$

4) The transfer time of a block. (2 points)

(8/256*0.95+7/256*0.05)/64*1000=0.485ms 8扇区+7gap $\left(\frac{8}{256} \times 90\% + \frac{7}{256} \times 10\%\right) \times \frac{60 \times 10^3}{3840}$

5) The average time for accessing 10 continuous blocks in one track on the disk. (2 points)

① 6.46+7.81+0.485*10=19.12ms

10 blocks = 80扇区 + 79 gap.

② $6.46 + 7.81 + \left(\frac{80}{256} \times 90\% + \frac{79}{256} \times 10\%\right) \times \frac{60 \times 10^3}{3840}$

6) Suppose that we know that the last I/O request accessed cylinder 2000, what is the expected (average) number of cylinders that will be traveled due to the very next I/O request to this disk? (5 points)

(1999 + ... + 1 + 0 + 1 + ... + 6192)/8192=

((1999*2000/2) +(6192*6193/2))/8192= 2584.54 cylinders

$\frac{1999}{8192} \times \frac{1999}{2} + \frac{6192}{8192} \times \frac{6192}{2}$

单次IO时间 = 寻道时间 + 旋转延迟 + 传送时间.

Problem 2 (15 points) : Data Storage

Suppose blocks of 4096 bytes are used to store variable-length records. The header of the block will have one 8-byte pointer to each record, and 128 other bytes for fixed data, including a count of the number of records currently in the block. Records may begin at any multiple of 4 bytes.

1) How many records of size 100 bytes could we fit in such a block? Why? (5 points)

2) Would the following combinations of records in such a block: A record of 400 bytes, 6 records of 500 bytes, and 5 records of 100 bytes each? Why? (5 points)

3) Suppose we store records of size 130 bytes, we repeatedly
   Insert 4 records,
   Delete 3 records,
   until there is no more space in the block. When a record is deleted, its pointer in the block header is replaced by a tombstone. What is the maximum number of records this block will hold? How much free space will there be in the block at this point? (5 points)

Problem 3 (15 points): Join

For the join of the relations R (A,B) and S (B,C), state the minimum number of memory blocks you would need to perform each of the following join algorithms. Assume B(R) = 5,000 blocks and B(S) =3,000 blocks. There are five join methods below, please state the minimum number of memory blocks you would need to perform each of the following join algorithms and choose the best join method for the different memory size and state why.

1) The minimum number of memory blocks you would need to perform each of the following join algorithms. (10 points)
   1. One pass join
   2. Nested loop join
   3. Two pass simple sort join
   4. Two pass Sort join (also known as sort merge join)
   5. Two pass hash join

2) Given memory size is 50 (2 points)

3) Given memory size is 80 (2 points)

4) Given the memory size is (1 points)

Problem 4 (15 points) : Index

Consider an extensible hash structure where buckets can hold up to two records and no overflow blocks are allowed. Initially the structure is empty.

1) Simulate inserts of the following keys in the order they are listed. Show the extensible hash structure for these keys using the diagram like we used in class.

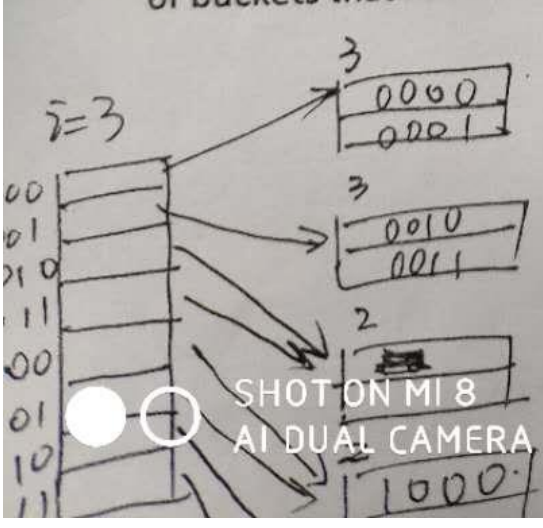
   0001, 0000, 1000, 0010, 1111, 0011, 1100, 1110

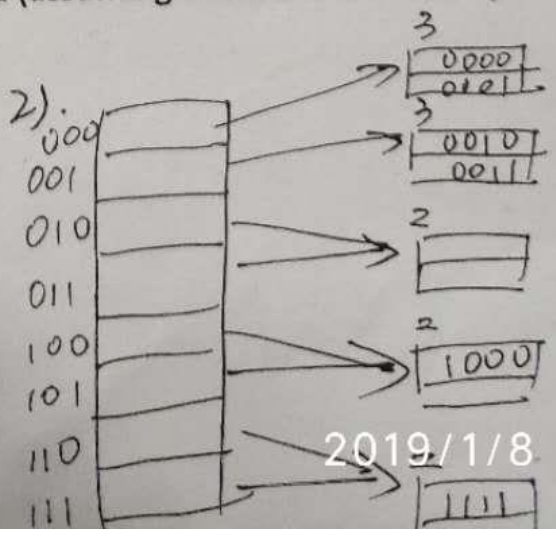
2) Now suppose we execute the following deletes on the same table: 1100, 1110. Show the extensible hash structure at the end of these steps. Assume that deletions re-structure the extensible hash table (i.e. we merge buckets when possible).

3) For this part, ignore the previous inserts and deletes. (Buckets can still hold up to two records and no overflow blocks are allowed.) We start with an empty extensible hash table with a directory that has 2 entries. After some insertions (and no deletions), we are told that the directory has grown to 512 entries.

   (i) What is the minimum number of keys that this hash table can hold? Give a sample key sequence that would generate this worst case behavior for such extensible hash table(you can assume keys are 9 bits long). 3

   (ii) If the table holds the minimum number of keys, what is the minimum number of buckets that have been allocated (assuming there are no deletes)?

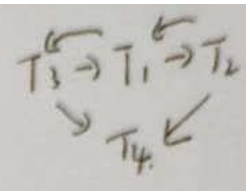




B: $T_2 \to T_1$ $T_1 \to T_3$

C: $T_3 \to T_4$

D:



blem 5 (14 points): Concurrency Control

1) For each of the following schedules, answer the questions below: (8 points)

Sa= R2(B) W3(A) R1(B)W3(C) W4(D)R1(A) R2(A) R3(B)W4(A) W4(C)

Sb= W3(A) R2(B) W2(C) R1(A) R1(B) W4(D) W2(A) R4(A) W3(C) R3(B)

A: $T_3 \to T_1 \to T_2 \to T_4$ ~~B: $T_2 \to T_1 \to T_3$~~ C $T_3 \to T_4$

(a) What is the precedence graph for the schedule Sa and Sb?

Sa: B: 无 A $T_3 \to T_1 \to T_4$ C $T_3 \to T_4$. ∴ Sa:

Sb: 同理. $T_3 \to T_1 \to T_2 \to T_4$

(b) Is the schedule conflict serializable? If so, show all equivalent serial transaction orders. If not, describe why not.

Sa: Yes. T_3 T_1 T_2 T_4 或 T_3 T_2 T_1 T_4

Sb: 不可以、PS中有写不.

2) Consider the following two transactions: (6 points)

T1 = R1(A) W1(A) R1(C) R1(B) W1(B) R1(B);

T2 = R2(C) W2(C) R2(B) W2(A) R2(C) R2(B);

a) 请添加合适的读锁（ls()）、写锁（lx()）和解锁（ul()）命令使事务T1和T2在并发运行时可以满足冲突可串行化调度。（为提高并发度，只涉及读的元素要加读锁）

按 A.B.C 顺序.

T1加锁和解锁的顺序: ~~$Lx_1(A)\ R_1(A)\ W_1(A)\ Ls_1(C)\ R_1(C)\ Lx_1(B)\ R_1(B)\ W_1(B)\ R_1(B)$~~ $Lx_1(A)\ R_1(A)\ W_1(A)\ Lx_1(B)\ Ls_1(C)\ R_1(C)\ R_1(B)\ W_1(B)\ R_1(B)\ U_1(A)\ U_1(B)\ U_1(C)$

T2加锁和解锁的顺序: $Lx_2(A)\ Ls_2(B)\ Lx_2(C)\ R_2(C)\ W_2(C)\ R_2(B)\ W_2(A)\ R_2(C)\ R_2(B)\ U_2(A)\ U_2(B)\ U_2(C)$

b) 请说明这两个事务会引起死锁吗？如果会引起死锁，请给出死锁的示例；如果不会引起死锁，请说明为什么？

不会，因为采用加顺序锁、都按 A.B.C 顺序加锁.

B: $T_1 \to T_2 \to T_4$

D: 开 $\to T_1$

Problem 6 (14 points): Transaction Management

Assume that a database using Undo/Redo logging and nonquiescent checkpointing crashes with the log records on disk given below. Record <T,X,v,w> means that transaction T changed the value of database element X; its former value was v, and its new value is w.

<START, T1>
<T1, A, 4, 9>
<START, T2>
<T2, B, 9, 10>
<COMMIT T2>
<START T3>
<START CKPT (T1,T3)>
<T1, C, 34, 19>
<T1, A, 9, 18>
<COMMIT T1>
<T3, A, 18, 36>
<END CKPT>
<T3, B, 10, 23>
<START T4>
<T4, C, 19, 21>
<COMMIT T4>

1) What are the all of the possible values on disk for each of the database elements A, B and C? (3 points)

For element A: 4, 9, 18, 36

For element B: 9, 10, 23

For element C: 34, 19, 21.

2) Which, if any, transactions will need to be redone and undone in the recovery process? (4 points)

Transactions to Redo: T_1, T_2

Transaction to Undo: T_3.

3) If finished the system recovery, what are the values on disk for each of the database elements A, B and C? (3 points)

For element A: 18

For element B: 10.

For element C: 21.

4) How would your answers to parts 1) and 2) change if <END CKPT> were not present in the log? (4 points)

For element A: 4 9 18 36

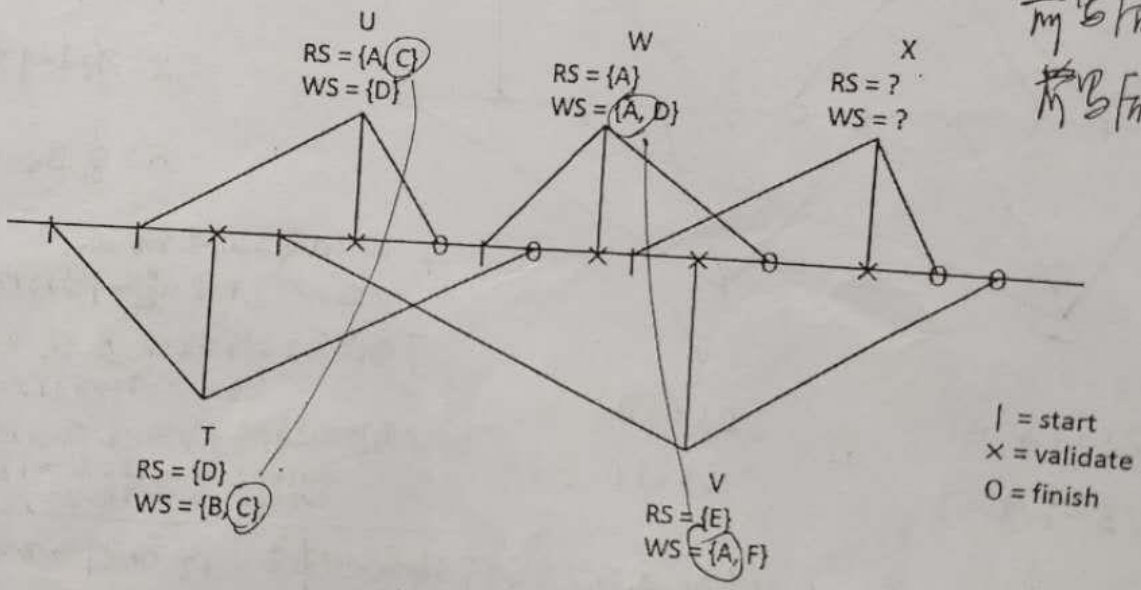
For element B: 9, 10, 23

For element C: 34, 19, 21.

Transactions to Redo: T1, T2, T4.

Problem 7 (12 points): Validation

Consider a database with six elements A, B, C, D, E and F. There are five transactions T, U, V, W and X that read and write to these database elements. The times at which the transactions start, try to validate and finish are as in the diagram below. The read and write sets of transactions T, U, W and V are also indicated.



1) Does the transaction U validate?(Yes/ No) No. (2 points)

2) Does the transaction V validate?(Yes/ No) No (2 points)

3) Does the transaction W validate?(Yes/ No) Yes. (2 points)

4) If we know that the transaction X validates, give the list of possible elements in the read set and write set of X.

Elements that could be in the read set of X A B C D E F (3 points)

Elements that could be in the write set of X A B C D E F (3 points)