NATIONAL UNIVERSITY OF SINGAPORE
SCHOOL OF COMPUTING
SEMESTER 2 (2021/2022) MID-TERM EXAMINATION FOR

CS3223: DATABASE SYSTEMS IMPLEMENTATION

March 2022 Time Allowed: 70 minutes

NAME: MATRIC NUMBER:

This paper contains 19 questions, 6 of these are 2-mark questions (indicated at the beginning of the question), and the rest are 1-mark questions. **The maximum score for this paper is 25 marks.** For multiple choice questions, pick the **BEST** answer (only ONE answer) for each question. **Fill in your answers in the following table.** You should read the questions in the order given (as some questions are related).

Q1	D		Q8 (2 marks)	D		Q15	34
Q2	D		Q9	B		Q16	C
Q3	B		Q10	E		Q17	G
Q4	9		Q11 (2 marks)	G		Q18 (2 marks)	3023 (3033)
Q5	13		Q12 (2 marks)	H		Q19 (2 marks)	D
Q6	67.8		Q13	120			
Q7 (2 marks)	58477		Q14	6			

1. The main components of the cost of performing a disk read/write are seek time, rotational delay and transfer time. Which of these components can be reduced by doing sequential reads rather than random reads.

   a) Seek time
   b) Rotational delay
   c) Transfer time
   d) Only (a) and (b) are true.
   e) Only (b) and (c) are true.
   f) Only (a) and (c)) are true.
   g) (a), (b) and (c) are all true.

2. Consider a page reference pattern that performs THREE consecutive scans over a file of FIVE pages. Assume the system starts with an empty buffer pool of FOUR frames. How many I/Os will be incurred with an LRU page replacement policy?
   a) 5 b) 10 c) 11 d) 15 e) None of the above

3. Suppose pages of 1024 bytes are used to store variable-length records. Besides the data records, each page also contains some meta-data: 24 bytes for some fixed data (among these 24 bytes is a counter of the number of records); in addition, for each record, there is a 8-byte pointer to the record. Records may begin at any multiple of 4 bytes (e.g., if a record is 15 bytes, then one byte would be wasted/unused; if a record is 37 bytes, then 3 bytes would be wasted). Consider a file with 1 record of 100 bytes, 1 record of 500 bytes and K records of 20 bytes. What is the maximum K value that can fit into this page?
   a) 10 b) 13 c) 14 d) 16
   e) 20 f) None of the above

4. Consider a B+-tree of order 2. Let T be the smallest tree of height 3 (in terms of number of tree nodes) which is constructed using the insertion algorithm that we learned in the lecture. What is the minimum number of tree nodes?

5. Referring to Question 4, consider the B+-tree T again. What is the minimum number of data entries in the leaf nodes if the tree is created exclusively through inserts (i.e., elements are added without deletions).

6. Consider a disk with the following characteristics:
   * average seek time = 18ms
   * track-to-track seek time = 5ms
   * average rotational delay = 8.3 ms
   * maximum transfer rate/track = 16.7 ms
   * time to transfer a node/page = 0.4 ms

   Suppose a 3-level (height = 3) B+-tree for a file S (in format 2) is stored on the disk such that the root is at track j, the internal nodes are at track j+1, and the leaf nodes are at track j+2. Suppose the file S is stored on track j+3. Where you have no other information, you may assume average seek time and/or average rotational delay. We further assume that the size of a tree node and a data page is the same. What is the

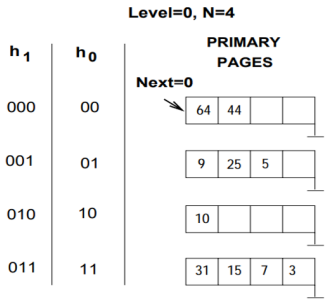
cost to retrieve an existing record from the data file using the index (in ms)? Note that no tree nodes are cached in the buffer.

7. **(2 marks)** Referring to Question 6. Suppose we are joining table R with table S using the index-nested loops join. Suppose R has 10 pages scattered across the disk. Each page holds 100 tuples. Moreover, suppose 30% of these tuples can find one matching tuple in S, while there are no matching records for the remaining 70% of the tuples. Assume that whenever the root is accessed, it is a random access (not a track-to-track seek). You may assume output tuples are returned to users without being stored. What is the total join cost (in ms)?

8. **(2 marks)** Modern disk drives store more sectors on the outer tracks than the inner tracks. Since the rotation speed is constant, the sequential data transfer rate is also higher on the outer tracks. The seek time and rotational delay are unchanged. Which of the following situation(s) is/are reasonable choice(s)?
   a) Consider a small file that is frequently but randomly accessed (i.e., some requests access the small file, then other requests may access files on other parts of the disks, and this pattern repeats). We should place the file in the middle tracks. Sequential speed is not an issue due to the small size of the file, and the seek time is minimized by placing files in the center.
   b) Consider a large file that is often scanned sequentially (e.g., selection from a relation with no index). We should place the file in the outer tracks. Sequential speed is most important and outer tracks maximize it.
   c) Consider a large file with random accesses via an index (e.g., selection from a relation via the index). We should place the file on the outer tracks and index on the inner tracks. The DBMS will alternately access pages of the index and of the file, and so the accesses are random anyway (so the two need not reside in close proximity).
   d) (a) and (b) only
   e) (a) and (c) only
   f) (b) and (c) only
   g) (a), (b) and (c)
   h) None of the above

9. What is the minimum number of buffers needed for index nested loops join assuming a unique B+-tree index (format 2) (including input and output buffers)?
   a) 2 b) 3 c) 4 d) height of tree + 2
   e) None of the above

10. Consider the following Linear Hash File. Assume that a bucket split occurs whenever an overflow page is created.



3: 00000011
5: 00000101
7: 00000111
9: 00001001
10: 00001010
15: 00001111
25: 00011001
31: 00011111
44: 00101100
64: 01000000

What is the maximum number of data entries that can be inserted before you have to split a bucket?

a) 2 b) 3 c) 4 d) 5 e) 6

11. **(2 marks)** Referring to Question 10, what is the minimum number of record insertions that will cause a split of **ALL** buckets?

a) 8 b) 9 c) 10 d) 11
e) 12 f) 13 g) None of the above

12. **(2 marks)** Insert the following keys into an Extendible hash table that is initially empty: 1111, 1000, 0111, 0011, 1011. Each bucket can hold only 2 keys. Which of the following statements is true after all the keys have been inserted.

a) The global depth is 4 (FOUR).
b) 50% of the directory pointers point to one bucket.
c) There are two buckets whose local depths are the same as the global depth.
d) All the buckets contain at least one key (i.e., none of the buckets is empty).
e) (a), (b) only
f) (a), (c) only
g) (a), (d) only
h) (b), (c) only
i) (b), (c) (d) only
j) (a), (b), (c) only
k) (a), (b), (c), (d) only
l) None of (a)-(d)

13. Consider sorting a file R. R has 20 pages, and each page can hold only one record. Suppose we have 4 buffer pages. What is the I/O cost to sort this file using the external multi-way sort algorithm (as covered in the lecture)?

14. Using the same R in Question 13. Suppose we adopt replacement selection instead of internal sort. Suppose using replacement selection results in sorted runs of twice the size of a run (compared to traditional internal sort used in Question 13). Note that the last run is the exception and may be smaller. Count the number of sequential and random I/Os to generate the initial sorted runs. For simplicity, assume that pages of a run or a file is always stored sequentially in the same track. Moreover, assume that different runs or files are always stored on different tracks, and moving from a run to a different run will incur a random I/O. Hint: Reading and writing k pages incurs 1 random I/O and (k-1) sequential I/O. What is the number of sequential I/Os?

15. Referring to Question 14 above. What is the number of random I/Os.

16. Suppose we have 10 buffer pages. Relation R can only be sorted in three passes using the basic external sort that we learned in class, i.e., generate sorted runs (one pass), and then two more additional merge passes are required to produce the final sorted runs. Which of the following statements is true (minimum/maximum R size refers to the smallest/largest size that can be sorted in 3 passes)?
    a) Minimum R size = 11 pages; Maximum R size = 90 pages
    b) Minimum R size = 100 pages; Maximum R size = 900 pages
    c) Minimum R size = 91 pages; Maximum R size = 810 pages
    d) Minimum R size = 100 pages; Maximum R size = 811 pages
    e) None of the above

17. Consider the join of two relations R and S. R has 1,000 pages and S has 500 pages. The join can be performed using a GRACE hash join in two passes (one for the partitioning phase, and one for the joining phase). Assume the ideal case that the tuples are uniformly distributed across all partitions. What is the minimum amount of memory (in pages) needed?
    a) 30 b) 31 c) 32 d) 33 e) 34 f) 35
    g) None of the above

18. **(2 marks)** Referring to Question 17, what is the earliest possible time (in terms of number of I/Os) that the first output tuple can be returned?

19. (**2 marks**) Suppose a page can hold 1000 bytes of data, and there are 201 memory pages. Consider a table R(a, b) such that attribute a requires 16 bytes and attribute b requires 36 bytes. Suppose that each integer value is 4 bytes. Consider the following query: Select a, COUNT(b) from R group biy a. Suppose you have developed a one pass algorithm to process this query over R, i.e., read R once and generates the output in memory (without storing any results on disk). What is the maximum number of R tuples that can be processed to guarantee the one pass scheme will always run successfully (Note: If maximum is k, then it is possible that an instance of R with k+1 tuples cannot be processed in one pass)?
    a) 1000 b) 2000 c) 5000 d) 10000
    e) None of the above since R may contain "infinite" number of duplicate values.

-- END OF PAPER –

1. D. Sequential reads means most of the content would be on the same track (then cylinder). So seek time and rotational delay can be reduced. Transfer time is always fixed regardless of whether the reads are sequential or random.

2. D. Since there are only 4 buffers, every page in the buffer will be replaced. We need to read 15 pages in total (3 scans of 5 pages).

3. B. 1024-byte page. 24 bytes meta-data. So, 1000 available for data and pointer. So, we have 100+500+k*20 + (k+2)*8 <= 1000. K < 13+, So, K = 13.

4. 9. Just draw the tree to insert 1-13.

5. 13. See (4). Generic formula: 2*3*2+1 (root has 2 child nodes, each of these child nodes has minimum occupancy (2 keys, 3 pointers), each of the leaf nodes has minimum occupancy (2) except the last node (which causes the split))

6. You just need to compute the cost to retrieve 1 block from every level. The root will require an average seek time, the other pages require a track-to-track seek time. All needs average rotational delay and transfer time. So 18ms + 8.3 + 0.4 + 3*(5+8.3+0.4) = 67.8

7. Cost to read R = 10*(18+8.3+0.4) = 267. For each matching tuple, 300*67.8. For each non-matching tuple, the time to traverse the index is 18+8.3+0.4+2*(5+8.3+0.4) = 54.1. So for non-matching tuples, the cost is 700*54.1. Total = 58210 + 267 = 58477

8. D. Place the file and index on the inner tracks. The DBMS will alternately access pages of the index and of the file, and so the two should reside in close proximity to reduce seek times. By placing the file and the index on the inner tracks we also save valuable space on the faster (outer) tracks for other files that are accessed sequentially. (Question from Raghu's text).

9. 3. Minimum is 3. 1 input buffer for outer table. 1 output buffer. 1 for index and data page.

10. 6. The maximum number of entries that can be inserted without causing a split is 6 because there is space for a total of 6 records in all the pages. A split is caused whenever an entry is inserted into a full page.

11. G. Consider the list of insertions: (a) insert 3 keys into bucket 0 such that it is split into bucket 0 (1 key: 64) and bucket 4 (4 keys: 44 + any 3 others that has 100 as last 3 bits). (b) insert 2 keys into bucket 1 such that it is split into bucket 1(4 keys: 9, 25, 2 more keys that end with 001 as last 3 bits) and bucket 5 (1 key: 5). (c) insert 1 element into bucket 4 resulting in bucket 2 being split. Bucket 6 is created and bucket 2's keys are redistributed. (d) insert 1 element into bucket 5 resulting in bucket 3 being split. Bucket 7 is created and bucket 3's keys are redistributed. So, a minimum of 7 entries are required to cause all 4 buckets to split.

12. H. Draw the structure.

13. 120. With 4 buffer pages, we need 3 passes. So, 2*20*3 = 120. If you try to optimize, 104 is also acceptable – generate 5 runs of 4 pages. Merge 3 of them. Then merge the 3 runs.

14. We have 2 runs of 8 pages, and 1 run of 4 pages. To generate the first run, we read in 4 buffer pages (1 random and 3 sequential I/Os); for each of the output page of the run, we incur a random I/O – write (next) smallest value, read the next input value – since we have 8-page runs, we basically incur 16 random I/Os. For second run, it is similar, so another 16 random I/Os. Finally, we have 4 pages in the buffer, and they can be written out (1 random I/O and 3 sequential I/Os). So, number of sequential I/Os = 6, and number of random I/Os = 34.

15. See 14.

16. C. With 10 buffer pages, each run is 10 pages. Each merging phase can merge 9 runs. So, if we have 9 runs (file of 90 pages), we can do it in 2 passes. The minimum for 3 passes would then be 91 pages. For maximum, we can merge 9 runs resulting in 90 page runs. In another merge, we can merge 9 90-page runs. So, 810 would be the maximum size.

17. G. 23 partitions each 22 pages (look at the smaller table of the two which is 500 pages). 24 buffers needed.

18. 3023. 2*1000 (partition R) + 2*500 (partition S) + build S1 (22) + read one page of R. To make the question “independent” of q17’s answer, I also accept the case: 2*1000 + 2*500 + build R1 (32) + 1 = 3033.

19. D. Consider γ (a,COUNT(b)). In a one-pass algorithm, we need to record each a we see (16 bytes) and the associated count of b's (4 bytes). We can thus count 50 groups, i.e., 1000/(16+4), per buffer. With 200 buffers, we can count 10,000 groups. Since all tuples of R may have distinct values of a, we can only be certain of success if R has no more than 10,000.