COMP 3005: Database Management Systems **(Due: Friday Dec. 10, 2021 (11:59 PM))**

Assignment #5

Instructor: Ahmed El-Roby *Name: , ID:*

Instructions: Read all the instructions below carefully before you start working on the assignment, and before you make a submission.

- The accepted formats for your submission are: pdf and docx. More details below.
- If you use the tex file, make sure you edit line 28 to add your name and ID. Only write your solution and do not change anything else in the tex file. If you do, you will be penalized.
- Late submissions are allowed for 24 hours after the deadline above with a penalty of 10% of the total grade of the assignment. Submissions after more than 24 are not allowed.

Q 1: (4 points)

RAID systems can support replacing failed disks without the system going offline. Which of the RAID levels better support this operation with the least amount of interference between the rebuild and ongoing disk accesses? Explain your answer.

RAID level 1. This is because rebuilding the failed disk involves copying from only the mirror (shadow) of the failed disk. In other RAID levels discussed in class, this will require reading from all other disks.

Q 2: (4 points)

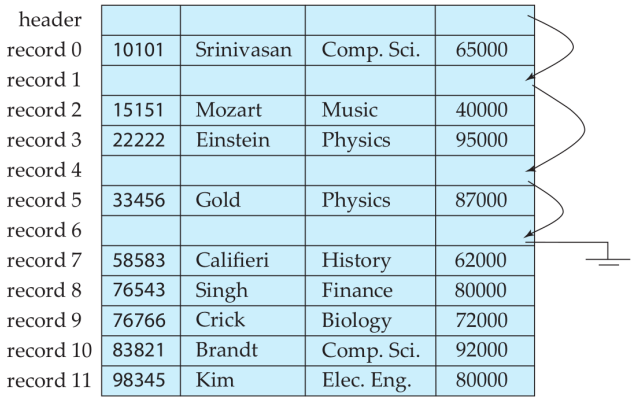
Consider the following arrangement for four disks, where B_i is a data block, and P_i is the parity block for the 4 data blocks that precedes it. What problem will this arrangement cause?

Disk 1	Disk 2	Disk 3	Disk 4
B_1	B_2	B_3	B_4
P_1	B_5	B_6	B_7
B_8	P_2	B_9	B_{10}
⋮	⋮	⋮	⋮

The problem is that one block and its parity block will reside on the same disk. That is, the parity block P_i and B_{4i-3} are on the same disk. When this disk fails, reconstruction of B_{4i-3} won't be possible.

Q 3: (8 points)

Consider the following file organization using free list.



header				
record 0	10101	Srinivasan	Comp. Sci.	65000
record 1				
record 2	15151	Mozart	Music	40000
record 3	22222	Einstein	Physics	95000
record 4				
record 5	33456	Gold	Physics	87000
record 6				
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000
record 11	98345	Kim	Elec. Eng.	80000

Show the structure of the file after each of the following operations (**they follow one another**):

(a) Delete record 11. (2 marks)
Record 11 is deleted.
Header points to record 1.
Record 1 points to record 4.
Record 4 points to record 6.
Record 6 points to record 11. Recorde 11's pointer is set to null.

(b) Insert (12345, John, History, 90000) (3 marks)
New record is inserted in record 1.
Header points to record 4.
Record 4 points to record 6.
Record 6 points to record 11.

(c) Insert (20000, Jamie, Physics, 100000). (3 marks)
New record is inserted in record 4.
Header points to record 6.
Record 6 points to record 11.

Q 4: (3 points)

In variable-length record representation, the record starts with offset and length pairs of variable-size attributes, followed by fixed-size attributes, then the null bitmap, and finally the variable-size attributes. How can we improve this representation if our application is expected to store tables with large number of attributes, most of which are nulls?
We should be able to locate the null bitmap and the offset and length values of non-null attributes using the null bitmap. This can be done by storing the null bitmap at the beginning and then for non-null attributes, store the value (for fixed size attributes), or offset and length values (for variable sized attributes) in the same order as in the bitmap, followed by the values for non-null variable sized attributes. This representation is space efficient but needs extra work to retrieve the attributes.

Q 5: (12 points)

Construct a B^+-tree for the following set of key values: (2, 3, 5, 7, 11, 17, 19, 23, 29, 31). The tree is initially empty and values are added one value at a time in ascending order. Consider the following values of n:
(a) $n = 4$. (4 points)
(b) $n = 6$. (4 points)
(c) $n = 8$. (4 points)

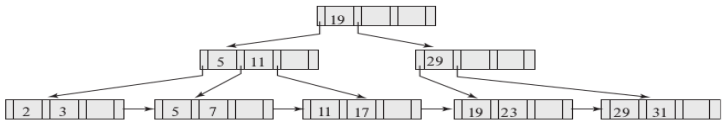


Figure 1: Q4.a ($n = 4$)

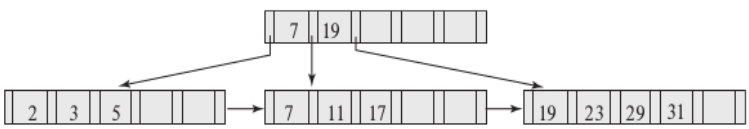


Figure 2: Q4.b ($n = 6$)

Q 6: (8 points)

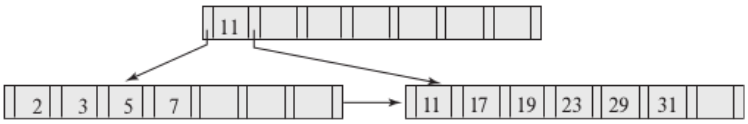
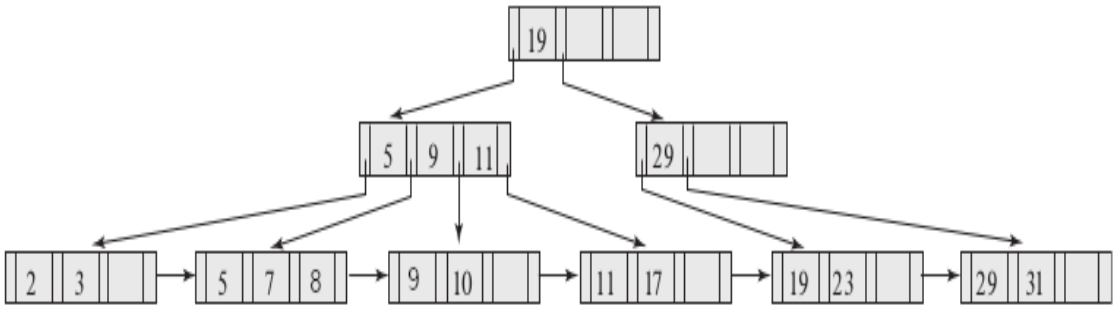


Figure 3: Q4.c ($n = 8$)

Consider the following B^+-tree with $n = 4$:



(a) Delete 23. (4 points)

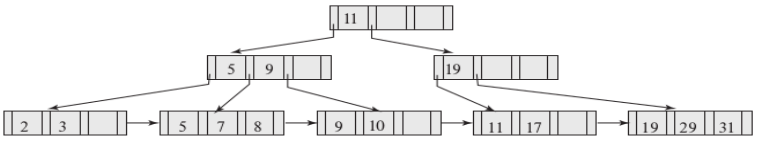


Figure 4: Q5.a (delete 23).

(b) Delete 19. (4 points)

Q 39: (4 points)

Consider the following B^+-tree with $n = 6$: Insert 8 into this tree.

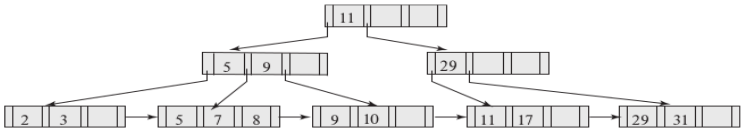
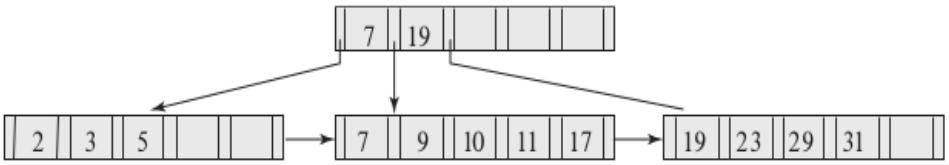


Figure 5: Q5.b (delete 19).



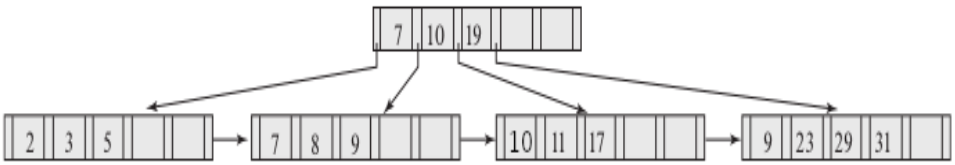


Figure 6: Q6 (Insert 8)