Subject: **Database Management Systems** (**CO3021**)
Semester 1, 2018-2019

Chapter 1: Disk Storage and Basic File Structures

Question 1.1. Describe the memory hierarchy for data storage.

Question 1.2. Distinguish between persistent data and transient data.

Question 1.3. Describe disk parameters when magnetic disks are used for storing large amounts of data.

Question 1.4. Describe double buffering. What kind of time can be saved with this technique?

Question 1.5. Describe the read/write commands with magnetic disks.

Question 1.6. Distinguish between fixed-length records and variable-length records.

Question 1.7. Distinguish between spanned records and unspanned records.

Question 1.8. What is blocking factor? How to compute it?

Question 1.9. What is file organization? What is its goal?

Question 1.10. What is access method? How is it related to file organization?

Question 1.11. Distinguish between static files and dynamic files.

Question 1.12. Compare unordered files, ordered files, and hash files.

Question 1.13. Which operations are more efficient for each file organization: unordered, ordered, hash? Why?

Question 1.14. Distinguish between static hashing and dynamic hashing.

Question 1.15. Compare three dynamic hashing techniques: Extendible Hashing, Dynamic Hashing, and Linear Hashing.

NOTE: The following exercises are extracted from those in chapter 16 of [1]: *R. Elmasri, S. R. Navathe, Fundamentals of Database Systems- 7th Edition, Pearson, 2016.*

Question 1.16. Consider a disk with the following characteristics (these are not parameters of any particular disk unit): block size B = 512 bytes; interblock gap size G = 128 bytes; number of blocks per track = 20; number of tracks per surface = 400. A disk pack consists of 15 double-sided disks.

a. What is the total capacity of a track, and what is its useful capacity (excluding interblock gaps)?

b. How many cylinders are there?

c. What are the total capacity and the useful capacity of a cylinder?

d. What are the total capacity and the useful capacity of a disk pack?

e. Suppose that the disk drive rotates the disk pack at a speed of 2,400 rpm (revolutions per minute); what are the transfer rate (tr) in bytes/msec and the block transfer time (btt) in msec? What is the average rotational delay (rd) in msec? What is the bulk transfer rate?

f. Suppose that the average seek time is 30 msec. How much time does it take (on the average) in msec to locate and transfer a single block, given its block address?

g. Calculate the average time it would take to transfer 20 random blocks, and compare this with the time it would take to transfer 20 consecutive blocks using double buffering to save seek time and rotational delay.

Question 1.17. A file has r = 20,000 STUDENT records of fixed length. Each record has the following fields: Name (30 bytes), Ssn (9 bytes), Address (40 bytes), PHONE (10 bytes), Birth_date (8 bytes), Sex (1 byte), Major_dept_code (4 bytes), Minor_dept_code (4 bytes), Class_code (4 bytes, integer), and Degree_program (3 bytes). An additional byte is used as a deletion marker. The file is stored on the disk whose parameters are given in Exercise 1.16.

a. Calculate the record size R in bytes.

b. Calculate the blocking factor bfr and the number of file blocks b, assuming an unspanned organization.

c. Calculate the average time it takes to find a record by doing a linear search on the file if (i) the file blocks are stored contiguously, and double buffering is used; (ii) the file blocks are not stored contiguously.

d. Assume that the file is ordered by Ssn; by doing a binary search, calculate the time it takes to search for a record given its Ssn value.

Question 1.18. Suppose that only 80% of the STUDENT records from Exercise 1.17 have a value for Phone, 85% for Major_dept_code, 15% for Minor_dept_code, and 90% for Degree_program; and suppose that we use a variable-length record file. Each record has a 1-byte field type for each field in the record, plus the 1-byte deletion marker and a 1-byte end-of-record marker. Suppose that we use a spanned record organization, where each block has a 5-byte pointer to the next block (this space is not used for record storage).

a. Calculate the average record length R in bytes.

b. Calculate the number of blocks needed for the file.

Question 1.19. Suppose that a disk unit has the following parameters: seek time s = 20 msec; rotational delay rd = 10 msec; block transfer time btt = 1 msec; block size B = 2400 bytes; interblock gap size G = 600 bytes. An EMPLOYEE file has the following fields: Ssn, 9 bytes; Last_name, 20 bytes; First_name, 20 bytes; Middle_init, 1 byte; Birth_date, 10 bytes; Address, 35 bytes; Phone, 12 bytes; Supervisor_ssn, 9 bytes; Department, 4 bytes; Job_code, 4 bytes; deletion marker, 1 byte. The EMPLOYEE file has r =

30,000 records, fixed-length format, and unspanned blocking. Write appropriate formulas and calculate the following values for the above EMPLOYEE file:

a. Calculate the record size R (including the deletion marker), the blocking factor bfr, and the number of disk blocks b.

b. Calculate the wasted space in each disk block because of the unspanned organization.

c. Calculate the transfer rate tr and the bulk transfer rate btr for this disk unit.

d. Calculate the average number of block accesses needed to search for an arbitrary record in the file, using linear search.

e. Calculate in msec the average time needed to search for an arbitrary record in the file, using linear search, if the file blocks are stored on consecutive disk blocks and double buffering is used.

f. Calculate in msec the average time needed to search for an arbitrary record in the file, using linear search, if the file blocks are not stored on consecutive disk blocks.

g. Assume that the records are ordered via some key field. Calculate the average number of block accesses and the average time needed to search for an arbitrary record in the file, using binary search.

Question 1.20. A PARTS file with Part# as the hash key includes records with the following Part# values: 2369, 3760, 4692, 4871, 5659, 1821, 1074, 7115, 1620, 2428, 3943, 4750, 6975, 4981, and 9208. The file uses eight buckets, numbered 0 to 7. Each bucket is one disk block and holds two records. Load these records into the file in the given order, using the hash function h(K) = K mod 8. Calculate the average number of block accesses for a random retrieval on Part#.

Question 1.21. Load the records of Exercise 1.20 into expandable hash files based on extendible hashing. Show the structure of the directory at each step, and the global and local depths. Use the hash function h(K) = K mod 128.

Question 1.22. Load the records of Exercise 1.20 into an expandable hash file, using linear hashing. Start with a single disk block, using the hash function h0 = K mod 20, and show how the file grows and how the hash functions change as the records are inserted. Assume that blocks are split whenever an overflow occurs, and show the value of n at each stage.

Question 1.23. Suppose that a file initially contains r = 120,000 records of R = 200 bytes each in an unsorted (heap) file. The block size B = 2,400 bytes, the average seek time s = 16 ms, the average rotational latency rd = 8.3 ms, and the block transfer time btt = 0.8 ms. Assume that 1 record is deleted for every 2 records added until the total number of active records is 240,000.

a. How many block transfers are needed to reorganize the file?

b. How long does it take to find a record right before reorganization?

c. How long does it take to find a record right after reorganization?

Question 1.24. Suppose we have a sequential (ordered) file of 100,000 records where each record is 240 bytes. Assume that B = 2,400 bytes, s = 16 ms, rd = 8.3 ms, and btt = 0.8 ms. Suppose we want to make X independent random record reads from the file. We could make X random block reads or we could perform one exhaustive read of the entire file looking for those X records. The question is to decide when it would be more efficient to perform one exhaustive read of the entire file than to perform X individual random reads. That is, what is the value for X when an exhaustive read of the file is more efficient than random X reads? Develop this as a function of X.

Question 1.25. Suppose that a static hash file initially has 600 buckets in the primary area and that records are inserted that create an overflow area of 600 buckets. If we reorganize the hash file, we can assume that most of the overflow is eliminated. If the cost of reorganizing the file is the cost of the bucket transfers (reading and writing all of the buckets) and the only periodic file operation is the fetch operation, then how many times would we have to perform a fetch (successfully) to make the reorganization cost effective? That is, the reorganization cost and subsequent search cost are less than the search cost before reorganization. Support your answer. Assume s = 16 msec, rd = 8.3 msec, and btt = 1 msec.