W4111 – Introduction to Databases Sections 002, V002; spring 2022

Homework 3a – Written Assignment

Instructions

- The homework submission date/time is 2022-APR-17 at 11:59 PM.
- Submission format is a PDF version of this document with your answers. Place your answers in the document after the questions.
- The name of your PDF must be <UNI>_S22_W4111_HW3a_Written.pdf. For example, mine would be dff9_S22_W4111_HW3a_Written.pdf
- You must use the Gradescope functions to mark the location of your questions/answers in the submitted PDF. Failure to mark pages will cause point deductions. **Please, please read the countless Ed posts, TA produced instructions and videos, etc. to prepare your submission.**
- You can use online sources but you must cite your sources. You may not cut and paste text.
- Questions typically require less than five sentences for an answer. You will lose points if your answer runs on and wanders.

  "Verbosity wastes a portion of the reader's or listener's life."

Questions

Question 1: Relative to disk drives, briefly define the following terms:

- Seek time
- Rotational latency time
- Transfer time/data transfer rate

Answer:

- **Seek time**:
  - Time it takes to re-position the arm over the correct track.
- **Rotational latency time**:
  - Time it takes for the sector to be accessed to appear under the head.
- **Transfer time/data transfer rate**:
  - Time required to transfer the required amount of data in disk to memory.

Question 2: Briefly explain logical block addressing and cylinder-sector-head addressing.

Answer:

- **Logical block addressing**:
  - It is a common scheme used for specifying the location of blocks of data stored on computer storage devices, generally secondary storage systems such as hard disk drives. LBA is a particularly simple linear addressing scheme; blocks are located by an integer index.
- **Cylinder-sector-head addressing**:
  - It is the process of identifying individual sectors on a disk by their position in a track, where the track is determined by the head and cylinder numbers. The terms are explained bottom up, for disk addressing the sector is the smallest unit.

Citation: https://en.wikipedia.org/wiki/Cylinder-head-sector#:~:text=CHS%20addressing%20is%20the%20process,sector%20is%20the%20smallest%20unit

Question 3: Some databases only map data to/use outer sectors/cylinders of hard disk drive. Why?

Answer:
The main reason is to optimize performance. Using outer sectors allows the DBMS to read or write more data with one rotation which could significantly reduce the rotational latency.

Question 4: Briefly explain the elevator algorithm for disk I/O scheduling and how it may improve performance?

Answer:

- **Definition**:
  - The algorithm determines the motion of the disk's arm and head in servicing read and write requests. When a new request arrives, the initial arm/head movement will be in one direction of the cylinder. As more requests arrive, they are serviced only in the current direction until the arm reaches the edge of the disk. Then direction reverses, and the requests that were remaining in the opposite direction are serviced.
- **Improvement**:
  - One way is to ensure all requests are serviced in only one direction, that is, once the head has arrived at the outer edge of the disk, it returns to the beginning and services the new requests in the same direction. This results in more equal performance for all head positions, as the expected distance from the head is always half the maximum distance.

Citation: https://en.wikipedia.org/wiki/Elevator_algorithm

Question 5: Relative to database buffer management, briefly explain the following concepts:

- Cache hit/cache miss
- Pinned page/block
- Least recently used replacement algorithm

Answer:

- **Cache hit/cache miss**
  - A cache miss is when something is looked up in the cache and is not found – cache did not contain the item being looked up.
  - The cache hit is when you look something up in a cache and it was storing the item and is able to satisfy the query.
- **Pinned page/block**
  - Pinned page: the pin count of its frame is incremented. It guarantees that the page will not be removed from the buffer pool by the buffer manager.
  - Pinned block: memory block that is not allowed to be written back to disk.
- **Least recently used replacement algorithm**
  - It is to use past pattern of block references as a predictor of future references. The page which is least recently used is replaced.

Citation: https://stackoverflow.com/questions/18559342/what-is-a-cache-hit-and-a-cache-miss-why-would-context-switching-cause-cache-mi#:~:text=A%20cache%20miss%2C%20generally%2C%20is,able%20to%20satisfy%20the%20query.

Question 6: Briefly explain the concepts of and differences between row-oriented storage and column-oriented storage. For each model, give an example of row access/query patterns that would benefit from the model.

Answer:

- **Difference**:

Table 1 Difference

Row oriented data stores	Column oriented data stores
Data is stored and retrieved one row at a time and hence could read unnecessary data if some of the data in a row are required	Data are stored and retrieve in columns and hence it can only able to read only the relevant data if required
Records are easy to read and write.	Read and write operations are slower as compared to row-oriented.
Best suited for online transaction system	Best suited for online analytic processing
These are not efficient in performing operations applicable to the entire data-sets and hence aggregation in row-oriented is an expensive job or operations	These are efficient in performing operations applicable to the entire data-set and hence enables aggregation over many rows and columns
Typical compression mechanisms which provide less efficient result	High compression rates due to little distinct or unique values in columns

- **Example**:
  - **Row oriented**: The query of retrieving a row or a set of rows, which is powerful for OLTP. Transaction typically create and retrieve.
  - **Column oriented**: The query of making aggregation, which is extremely powerful for BI scenarios such as aggregation ops, e.g. SUM, AVG and PROJECT to get a few columns

Citation: https://www.geeksforgeeks.org/difference-between-row-oriented-and-column-oriented-data-stores-in-dbms/

Question 7: Provide one benefit and one disadvantage of variable length record management/organization relative to fixed length record management/organization.

Answer:

- **One advantage**:
  - Use only as much storage as is needed, so it can accommodate unusual data not originally planned.
- **One disadvantage**:
  - Difficult to insert or delete and hard to search through, so it cannot simply map a C structure to a record.

Citation:
http://www.eli.sdsu.edu/courses/spring95/cs596_3/notes/databases/lect10.html

Question 8: The sample database associated with the book has a *takes* relation/table – *takes(ID, course_id, section_id, semester, year, grade)* that associated students and courses/sections.

- How might table partitioning (storage) be beneficial for storing the rows in the table?
- What columns would the database use to partition the data?
- Why might the database partition the *takes* table but not partition the *student* and *section* tables?

Answer:

- The database have the ability to read or write data very quickly(faster query processing); Easier administration of large tables; Flexible index placement.
- Takes.year could be the column to be partitioned into the takes information in different years.
- Every column in a table could be partitioned but would the partition improve the performance is significant. If we partition table *takes* by

the year column, we can visit the current year records faster. But the *student* and *section* table aggregate some of the records into the *takes* table, as well, *student* and *section* table themselves is ID specific, it's hard to make efficient partition on this 2 tables.

Question 10: Briefly explain and give 2 examples of how the algorithm for mapping records/rows to file/disks blocks can significantly affect performance.

Answer:

- **Example**:
  - Comparison of using Heap, Sequential and B+ tree organization
    - **Sequential organization** is fast and efficient method for huge amount of data, it's simple and cheaper. But time wastage is more than Heap organization as it takes time and space for sorting records.
    - **Heap organization** is faster at fetching and retrieving records than sequential record but only in case of small databases. But it waste many unused memory blocks and is very inefficient for larger databases.
    - **B+ Tree** traversal is easier and faster, so searching becomes easy as all records are stored only in leaf nodes and are sorted sequential linked list and the size is adjustable. But it is inefficient for static tables.

Citation: https://www.geeksforgeeks.org/file-organization-in-dbms-set-1/?ref=lbp