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

Homework 4a – Written Assignment

Instructions

- The homework submission date/time is 2022-MAY-01 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_HW4a_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: Explain why a sparse index must also be a clustering index.

Answer:
A sparse index contains pointers to every block in the database. So, it has entries for only some rows in the database. Since a clustering index has one entry for every distinct value in the clustering field and a pointer to the first value with this field, it must be sparse.

Question 2: Briefly explain sparse, multi-level indexes and their benefits. Why can the outer index be sparse?

Answer:
Even with a sparse index, index size may still grow too large. For example, if we have 1Billion records, with 1000 records per block, that is still 1 Million index records. That is still a 1000 index blocks. This might be too large to fit in the main memory, so a search will result in several disk reads.
The solution is to build another index on the index records.
These help reduce the disk reads for index lookups.

Moreover, since the main index is always sorted, the outer index can be sparse.

Ref: https://www.guru99.com/indexing-in-database.html

Question 3: Indexes can significantly improve performance? What are some disadvantages of having many indexes?

Answer:

Indexing let's us quickly retrieve records from a database by storing pointers to memory locations. Hence, they can reduce the number of disk reads to search, thereby significantly improving the performance.

Some disadvantages of indexing:

1. Need a unique non-null primary key on the data
2. Cannot partition an index-organized table
3. Decreases performance of Insert, Delete and Update query.

Ref: https://www.guru99.com/indexing-in-database.html

Question 4: Briefly compare the pros and cons of B^+-Tree versus a Hash Index.

Answer:

B-Tree vs Hash Index

Pros:

Can perform range selection faster O(log(n)) for B-Tree and Hash Index might take O(n)

B-Tree easier to maintain, grow etc. Hash algorithms do not scale, leading to bucketing and longer search times

B-Trees are more memory compact

Cons:

Searching takes O(log(n)) for B Tree but Hash Index takes O(1)

B-Trees might require balancing on insertion/deletion, leading to overheads.

Ref: https://stackoverflow.com/questions/7306316/b-tree-vs-hash-table

Question 5: Briefly explain the concepts of covering index/covered query.

Answer:

A covering index is a special case of an index where all required fields for a query are included in the index; in other words, the index itself contains the required data to execute the queries without having to execute additional reads.

Question 6: Briefly explain the three main steps/stages in query processing.

Answer:

Parser/Translator
– Verifies syntax correctness and generates a parse tree.
– Converts to logical plan tree that defines how to execute the query.
    Tree nodes are operator(tables, parameters)
    Edges are the flow of data “up the tree” from node to node.
Optimizer
– Modifies the logical plan to define an improved execution.
– Query rewrite/transformation.
– Determines how to choose among multiple implementations of operators.

Engine
– Executes the plan
– May modify the plan to optimize execution, e.g. using indexes.

Ref: Lecture notes

Question 8: Explain the role of equivalent queries in query optimization.

Answer:
For query optimization, the optimizer picks the equivalent query with the lowest cost, leading to the fastest execution time.

Question 9: Assume that the ON clause in a JOIN on tables A and B compares columns a1 with b1 and column a2 with b2. What two properties must the ON condition have for the optimizer to be able to use a Hash Join?

Answer:
The hash join can only be used to equi-join tables. So the ON condition must have a1=b1 and a2=b2. Moreover, between the two conditions we must use an AND operation.

Question 10: What is index selectivity? How does it factor into query optimization for JOINs?

Answer:

Index selectivity is how tightly a database index helps narrow the search for specific values in a table. So for JOIN operations, an index with a high selectivity will lead to fewer searches for the specific records, leading to faster join operations.