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:

- **Sparse Index:** It is an index record that appears for only some of the values in the file.

- **Clustering Index:** records themselves are stored in the Index and not pointers.
- **So**, based on the definition, both of them point to a whole, a entity, which means the sparse index pointing to A is the same as a clustering index pointing to a file record cluster which only contains A.

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

Answer:

- **Sparse Index**
  - **Definition:** It is an index record that appears for only some of the values in the file.
  - **Benefits:** It is faster and needs less space, less maintenance overhead for insertion, and deletions.
- **Multi-level indexes**
  - **Definition:** Index records comprise search-key values and data pointers.
  - **Benefits:** Index helps in breaking down the index into several smaller index in order to make the outermost level so small that it can be saved in a single disk block, which can easily be accommodated anywhere in the main memory.
- Why can the other index be sparse?
  - The index has 3 types, primary, clustering and secondary. Further, the primary index contains sparse and dense index. So sparse index is the smallest unit of index records, or all types of index can be implemented with different numbers and organizational forms of sparse index.

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

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

Answer:

- **Improve performance?**
  - It can efficiently retrieve records from the database files based on some attributes on which the indexing has been done.
- **Disadvantage**
  - Every time data changes in the table, all the indexes need to be updated.
  - Indexes need disk space. The more indexes you have, more disk space is used. It slows down the query performance.
  - Index decreases the performance on inserts, updates and deletes.

Citation:
https://www.tutorialspoint.com/dbms/dbms_indexing.htm#:~:text=Multi%2Dlevel%20Index%20helps%20in,anywhere%20in%20the%20main%20memory.

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

Answer:

- **Hash index**
  - **Pros**
    - It allow for fast access when retrieving the record by the key value. And it require less storage space.
  - **Cons**
    - Databases can degrade if they go through a large number of collisions.
    - It only equivalent search is supported and it can't be used to sort.
    - It can’t deal with range query retrieval.
- **B^+-Tree index**

- Pros:
  - Records can be fetched in equal number of disk accesses.
  - We can access the data stored in a B+ tree sequentially as well as directly.
  - Keys are used for indexing. Faster search queries as the data is stored only on the leaf nodes.
- Cons:
  - Extra insertion and deletion overhead, space overhead.
  - Less efficient for static tables.

Citation: https://dev.mysql.com/doc/refman/8.0/en/index-btree-hash.html

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

Answer:

- **Covering index:** It is a non-clustered index which includes all columns referenced in the query and therefore, the optimizer does not have to perform an additional look up to the table in order to retrieve the data requested.
- **Covered query:** It is a query that can be satisfied entirely using an index and does not have to examine any documents.

Citation: https://www.mongodb.com/docs/manual/core/query-optimization/

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

Answer:

- **Parsing and translation**
  - Translate the query into its internal form. This is then translated into relational algebra.
  - Parser checks syntax, verifies relations.
- **Optimization**

- Among all equivalent evaluation plans choose the one with lowest cost. Cost is estimated using statistical information from the database catalog.

- **Evaluation**
  - The query-execution engine takes a query-evaluation plan, executes that plan, and returns the answers to the query.

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

Answer:

- Explanation:
  - Any two relational expressions are said to be equivalent, if both the expression generate same set of records.
  - Analyze and transform equivalent relational expressions: Try to minimize the tuple and column counts of the intermediate and final query processes.
  - All equivalent queries are measured in query optimization step of its cost(space and time) and then DBMS will execute most efficient query among them.

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:

- **Two properties**
  - MySQL employs a hash join for any query for which each join has an equi-join condition, and in which there are no indexes that can be applied to any join conditions.
  - A hash join can also be used when there are one or more indexes that can be used for single-table predicates.

Citation: https://dev.mysql.com/doc/refman/8.0/en/hash-joins.html

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

Answer:

- **Index selectivity:** Index selectivity is how tightly a database index helps narrow the search for specific values in a table. To be specific, it is defined as the fraction of rows in a table having the same value for the indexed key.
- **How?**
  - The selectivity of an index determines its effectiveness in optimizing performance. Because a highly selective index has few rows for each index entry and an unselective index has many rows for each index entry.
  - So when doing JOIN operation, if 2 tables are both of high index selectivity, they can do less partition of each file record pointed by certain index, which the speeds up JOIN operation's partition process, then further speeds up the query optimization for JOINs.