As per UGC guidelines an electronic bar code is provided to seure your paper

International Journal for Modern Trends in Science and Technology, 8(05):464-468, 2022

ISSN: 2455-3778 online
DOI: https://doi.org/10.46501/IJMTST0805069
Available online at: http://www.ijmtst.com/vol8issue05.html

In-Memory Encrypted Database: Analysis on B and B+ Tree Query Optimization

Gaurav Bisht | Dr. Sunil Maggu

[1]Department of Information Technology, Maharaja Agrasen Institute of Technology, Rohini, Delhi
[2]Assistant Professor, Department of Information Technology, Maharaja Agrasen Institute of Technology, Rohini, Delhi
Corresponding author Email ID: gaurav13bisht@gmail.com

To Cite this Article
Gaurav Bisht and Dr. Sunil Maggu. In-Memory Encrypted Database: Analysis on B and B+ Tree Query Optimization. International Journal for Modern Trends in Science and Technology 2022, 8(05), pp. 464-468. https://doi.org/10.46501/IJMTST0805069

Article Info
Received: 22 April 2022; Accepted: 16 May 2022; Published: 22 May 2022.

ABSTRACT

This research paper focuses on a JavaScript and Node.js database project. "IN-MEMORY ENCRYPTED DATABASE: ANALYSIS ON B AND B+ TREE QUERY OPTIMIZATION" is a CLI (Command Line Interface) app that is used to store data into the system's main memory, allowing for faster access to data that would otherwise take too long due to the use of secondary memory to store the data in secondary memory.Encryption is improved further by using the AES algorithm, as well as optimizing our database and analyzing query access speeds by using the B and B+ Tree for faster query processing on the Database management system, as opposed to the trivial method of using a regular file system, which was much slower.

KEYWORDS:Database, Database Management System, Advanced Encryption Standard, B Tree, B+ Tree, Optimizer, Encryption, JavaScript.

1. INTRODUCTION

People's use of the internet has reached new heights in the current context. The use of the internet will rise soon as well, because a big portion of the public is still unaware of its existence. Data is an essential component of the internet, and it must be stored and retrieved in the most secure and efficient manner possible. Any normal internet user's privacy, data security, and speedier access to that data are all high priorities. This study focuses on the same problems that an individual, business, or institution could have. It provides a management system of data which stores and accesses data in the system's main memory, resulting in speedier data access. When storing data, AES algorithm is used which is a symmetric key cipher i.e. same secret key is used for both encryption and decryption which increases security. Using B and B+ Tree indexing has improved query processing speed even further, and a comparison of the two indexing approaches shows a distinct difference between them.

2. OBJECTIVES

The main objectives of the project are:

- Examining query execution times when employing B and B+ Trees indexing.
- Create a private and immediate database utilizing the user's system's main RAM.
- Encrypt data using the Advanced Encryption Standard (AES) algorithm to improve data security and give multi-layered protection.

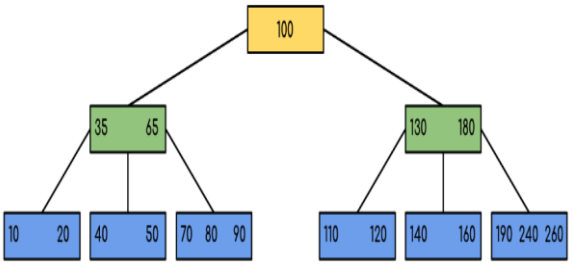
- To give faster data access by delivering a speed of approximately 10^6 times that of a secondary memory-based Database system.
- To provide a platform for reducing the risk of a data breach and its implications.
- To eliminate reliance on external database systems where data privacy is a concern.

3. LITERATURE SURVEY

Optimizing database access speed and data security are two significant areas of enhancing database performance and reliability.

1. The read/write speed of magnetic discs is the primary concern in terms of access speed. Moore's law improves CPU and memory performance, but disc performance continues to lag. Because of the speed gap between CPUs and drives, encrypted databases with main memories that directly insert data into the main memory have been developed to perform manipulation. Memory read/write speeds are approximately 10^6 times faster than disc read/write speeds.

2. The B Tree is a particular m-way tree that can be used to access discs in a variety of ways. At most m-1 keys and m children can be found in a B-Tree of order m. One of the main advantages of the B tree is its capacity to store many keys in a single node and huge key values while keeping the tree's height low.



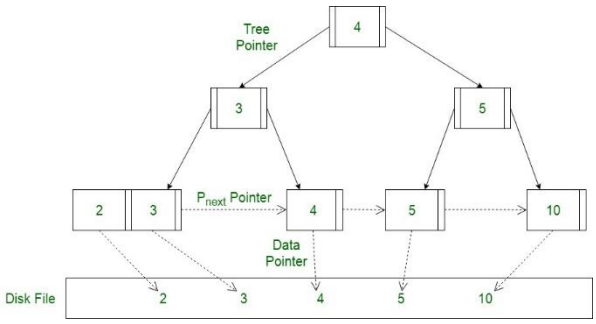
All the features of M way tree are present in a B tree of order m. Additionally, it has the following features.

- Each node in a B-Tree has a maximum of m children.
- Except for the root and leaf nodes, every node in a B-Tree has at least m/2 children.
- At least two root nodes are required.
- All leaf nodes must be at the same level.

It is not necessary that, all the nodes contain the same number of children but, each node must have m/2 number of nodes.

3. B+ Tree is a B Tree enhancement that enables for faster insertion, deletion, and search operations. Both keys and records can be stored in the internal and leaf nodes of a B Tree. In a B+ tree, records (data) can only be kept on the leaf nodes, whereas key values can only be placed on the interior nodes. To make search queries more efficient, the leaf nodes of a B+ tree are linked together in the form of singly linked lists.

B+ Trees are used to store vast amounts of data that are too large to fit in the main memory. The internal nodes (keys to access records) of the B+ tree are stored in the main memory, whereas leaf nodes are stored in the secondary memory, due to the restricted amount of main memory.



4. The Advanced Encryption Standard (AES) was created by the US National Institute of Standards and Technology (NIST) in 2001 as a specification for the encryption of electronic data. Despite being more difficult to implement than DES and triple DES, AES is commonly used today.

Points to remember:

- AES is a block cypher.
- The key size might be 128/192/256 bits.
- Data is encrypted in 128-bit blocks.

That is, it accepts 128 bits as input and outputs 128 bits of encrypted cypher text. AES is based on the substitution-permutation network principle, which entails replacing and shuffling the input data through a series of connected processes.

4. IMPLEMENTATION

The project has been implemented in the following aspects, which include employing B and B+ Tree indexing to improve query processing time and using AES to secure the stored data:

- Increased processing speed: The use of B and B+ tree indexing, in which nodes with data pointers are stored in tree data structures for faster retrieval and to reduce query processing time, has increased the speed with which data can be accessed. The primary memory of the system is where data is stored. The main memory is approximately 10^6 times faster than the disc memory. As a result, data retrieval speed is improved. While employing the system's main memory, JavaScript and its beneficial principles were used to store data and offer a database management system. The data is then saved in secondary memory, and anytime the user requires access to that database, it is loaded into the main memory for quick access.

- B and B+ Tree Analysis: The same queries were performed on both a B and a B+ tree indexing-based system, and the time taken to process each query was recorded, and graphs were constructed based on the results, demonstrating the difference in processing time between the two indexing approaches.

- Enhanced Security: The encryption in this project was accomplished utilizing the AES algorithm, which is a data encryption technology. When the user provides the data, the encryption procedure begins. After encryption, the data is saved in the main memory and remains unchanged even after being saved to the secondary memory. When a user requests data using the "search" query, it is decrypted, and the original data is displayed to the user.

5. TECHNOLOGY USED AND REQUIREMENT ANALYSIS

JavaScript: JS is a lightweight, interpreted, or just-in-time compiled programming language with first-class functions. While it is most well known as the scripting language for Web pages, many non-browser environments also use it, such as Node.js

Node.JS: Node.JS is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser.

Software Requirements:

Visual Studio Code: We will be using VS Code to run NODE.JS, which is a server side runtime environment, based on JavaScript. It simply means that you can decide on backend development being written in JavaScript.

Git: Git is software for tracking changes in any set of files, usually used for coordinating work among programmers collaboratively developing source code during software development.

Hardware Requirements:

- 1.6 GHz or faster processor
- 2-4 GB or more RAM
- Nvidia's GPUs (Preferred)
- Intel's CPU (preferred)

6. RESULTS

This database allows the user to enter data, but any changes to the database can only be made when a Transaction has been started, and it follows Serial Scheduling, so in order to start a new Transaction, the user must first close the current Transaction, and after each Transaction is completed, the database is updated, and if the user needs to Roll back to the previous state of the database for any reason, the database will revert. Secondary memory can be used to store databases.

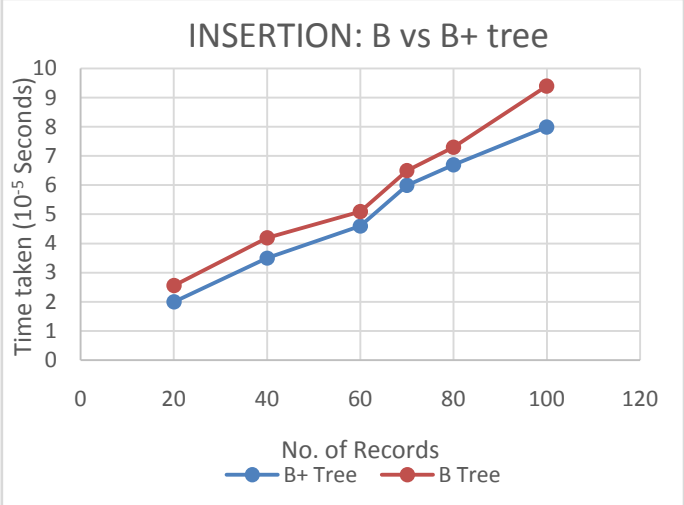
The use of B and B+ tree indexing further reduced the time taken for inserting, searching, and deleting records, and the analysis on each technique is further highlighted by showing a graph with the time spent and the number of records on the Y and X axes, respectively.

In case of Insertion of data, insertion takes more time in B tree and it is not predictable sometimes whereas in case of B+ Tree, insertion is faster (Fig. A) and the results are always the same.

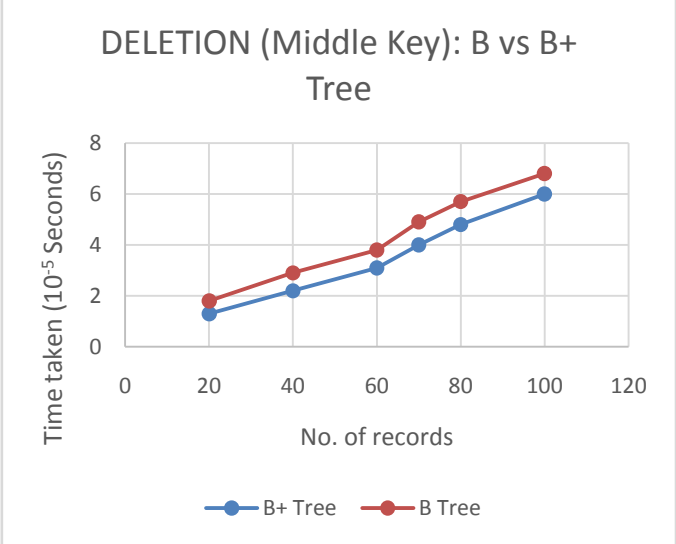
In case of deletion of data, deletion of internal node is very complex and B tree has to undergo lot of transformations whereas in case of B+ tree, deletion of any node is easy and faster (Fig. B) because all node are found at leaf.

In case of searching of data, since all keys are not available at leaf in B tree, search often takes more time as compared to the B+ tree indexing (Fig. C) where all keys are at leaf nodes, hence search is faster and accurate.

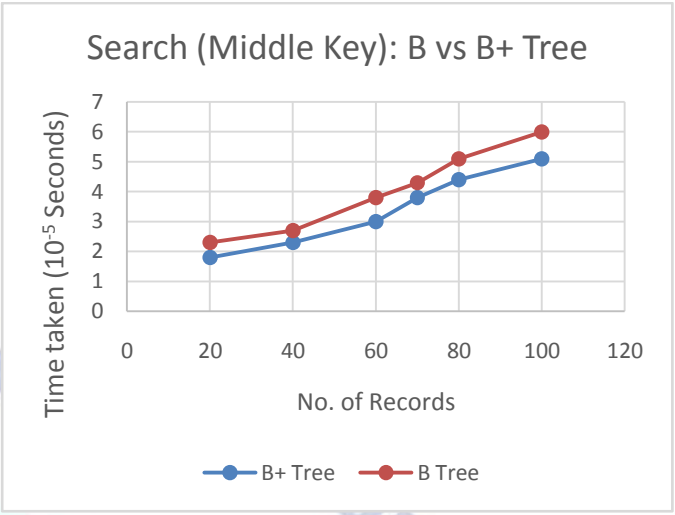
The AES Algorithm was implemented to make our database more encrypted and safer. It works when new data is entered and when it is retrieved using INSERT and SEARCH queries, respectively. When data is entered into the database, the encryption process begins, and the database is stored in the main memory in encrypted form. When the user retrieves data, the decryption process occurs, and the data is displayed to the user in its original form.



(Fig. A)



(Fig. B)



(Fig. C)

7. CONCLUSION

This project can encrypt data, hence increasing security. It provides the user with multi-layered security. It is highly adaptable and can be further changed to meet future needs. The data access time has been raised by a factor of nearly 106 times. A good front-end improves the user experience and makes it easier to select different database processes. This project has the potential to be used for both personal and commercial purposes.

Because of the quick transactions on data while stored on their machine only, moderate-scale products will benefit the most from our project, and even if they decide to store it on cloud databases, the encryption using the AES algorithm will ensure that it cannot be used by others in the event of data breaches. Analysis of B and B+ tree indexing revealed a clear picture of query processing times, with B+ Tree indexing processing queries in significantly less time than B Tree indexing.

ACKNOWLEDGMENT

Foremost, I would like to express my sincere gratitude to my mentor Dr. Sunil Maggu for the continuous support during the project and research paper, for patience, motivation, enthusiasm and immense knowledge. His guidance helped in all the time of research and writing of this paper. I could not have imagined having a better advisor and mentor for this project.

Conflict of interest statement

Authors declare that they do not have any conflict of interest.

REFERENCES

[1] H. Garcia-Molina and K. Salem, "Main memory database systems: an overview," in IEEE Transactions on Knowledge and Data Engineering, vol. 4, no. 6, pp. 509-516, Dec. 1992, doi: 10.1109/69.180602.

[2] Saringat, Mohd & Mostafa, Salama & Mustapha, Aida & Hassan, Mustafa. (2020). A Case Study on B-Tree Database Indexing Technique. 10.30880/jscdm.2020.01.01.004.

[3] https://www.oracle.com/in/database/what-is-database/

[4] https://www2.cs.duke.edu/csed/poop/huff/info/

[5] https://developer.mozilla.org/en-US/docs/Web/Javascript

[6] https://www.javatpoint.com/b-tree

[7] https://www.javatpoint.com/b-plus-tree

[8] https://www.geeksforgeeks.org/advanced-encryption-standard-aes/