A Comparative Study on Query Optimization Techniques in NoSQL

Heena Patel
M. K Institute of Computer studies, Bharuch affiliated to Veer Narmad South Gujarat University –Surat, Gujarat
Email ID:heena2006@gmail.com
Dr. Prashant P. Pittalia
P. G. Department of Computer Science and Technology, Sardar Patel University- Vallabh Vidyanager, Gujarat
Email: prashantppittalia@yahoo.com

ABSTRACT

In today's on-demand business process ingesting a large volume of data that is big data, becoming a very important concept in digitalization word, storage of data is increasing and quick development in cloud computing is on demand. Many growing companies have jumped a variety of data that are structured, unstructured and semi-structured. They do not contain the table and a key model like a traditional relational database which is lack data consistency and can't reach internet scalable and performance. To overcome these challenges new databases are known as NoSQL Databases. Now a day's applications have become more networked and social, and the number of data requests is also increasing, thus NoSQL databases are preferred rather than the traditional database. It is very challenging that which NoSQL model to use and how it is suitable for an organization's needs and how to retrieve a huge number of data in minimum response time. Retrieving quick data in NoSQL that is improve the query optimization techniques is a recent research area. The main idea of this paper is to give a review of the NoSQL database, different models, query processing, and query optimization techniques that can be used for improving performance and reducing response time.

Keywords – **Bloom filter, Hashing, Indexing, NoSQL, Partitioning, Query Processing, Sharding.**

I. INTRODUCTION

Big data is one of the most demanding eras in information technology (IT). In today's technologies such as cloud computing, the main goal is to store and retrieving of data at the minimum response time and maximize throughput. For many decades, Relational Database Management System (RDBMS) is being considered the bests' solution for storing and retrieving data but now a day's continuous development of the Internet and big data, various types of applications have emerged, which made new challenges in the database technology. It requires storing and processing a huge number of data. Especially in large scale and high concurrency applications, such as social applications like Facebook, Twitter, etc. For storing, processing, and working with big data, we need a dynamic distribution storage system. Therefore, to efficiently store and manage the distributed big data system named NoSQL stands for "Not Only SQL" became more popular in today's developing scenario.

NoSQL (Not Only SQL) is a non-relational database that does not require a fixed schema. NoSQL is good at dealing with a huge amount of data involved in big data. The purpose of using a NoSQL database is for distributed storage of data to the handling of parallel processing. NoSQL is mainly based on horizontal scalability and without any failure. NoSQL shifted from ACID property into BASE (Basically Available, Soft state, and Eventually Consistent) along with CAP theorem which stands for consistency, availability and partition tolerance[1]. Which provide two of these three characteristics either availability and partitioning (AP), or consistency and availability (CA) or availability and partitioning (AP). The usage of a NoSQL database is for processing a huge number of data effectively and high performance when reading and writing data. The volume of data is huge so the security and performance issues are major challenges in NoSQL Database.

The paper has been divided in following sections. Section2 describe the various NoSQL data model in detail. Section 3 discuss the query processing and query optimization techniques in NoSQL. Section 4 is analysis section here we compare data model and analyzed various techniques used in literature survey with its advantages and limitation. In finally Section 5 we conclude the work by providing further research objectives and future scope

II. NOSQL DATA MODELS

NoSQL provides flexible schemas and scale-out feature using open source software, easily with large amounts of data with high user loads. Data model of traditional database are mainly relational, specifically to support associated class operations and ACID transactions, but in the NoSQL database support CAP theorem [1], have following types of data model.

2.1 Key-value store
2.2 Column-oriented database
2.3 Document store
2.4 Graph database

2.1 Key-value store data are stored and retrieved using key–value pair that is uniquely identified record, shown in (Fig.1). It is used to quickly find the data within database. It is designed to handle lots of data and heavy load. The value can be a JSON (Java script orientation notation), BLOB (Binary Large Objects), string, etc. It can be used for storing session information, user profile.

Example :Redis, Riak and Amazon's DynamoDB.

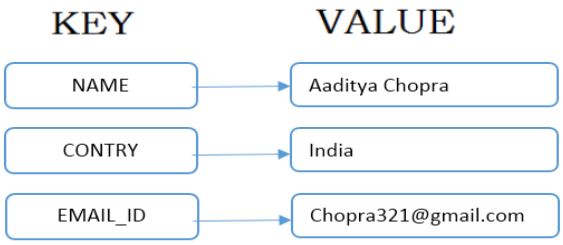

Fig. 1. Key value store

2.2 Column oriented are also known as wide-column store. Data store as column families contains rows and columns shown in (Fig 3). Each row can have many /different columns associated with a row key shown in (Fig. 2). This makes for faster access during read/write operations. High performance on aggregate queries like SUM, COUNT, AVG, MIN. We can use it for content management system, log aggregation, catalog, etc.

Example : Cassandra, Hbase, amazon DynamoDB.

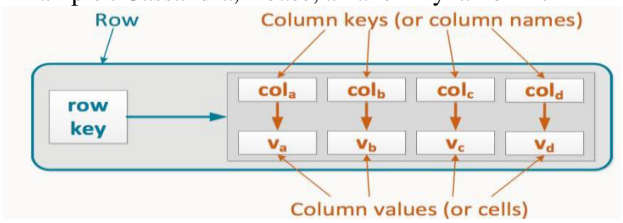

Fig. 2. Column value

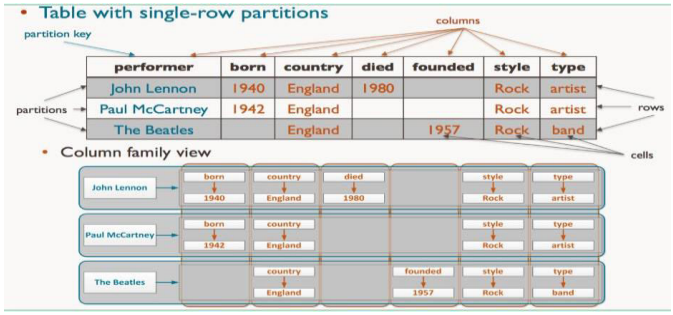

Fig. 3. Data in column family [21]

2.3 Document base is an extension to the key value stores NoSQL databases. Databases are designed to store, retrieve and manage the document-oriented information. Data store in semi-structure format like XML, JSON (Java script orientation notation) shown in [Fig 4]. We use it for real time analysis, Amazon searching

Example : Amazon SimpleDB, CouchDB, MongoDB.

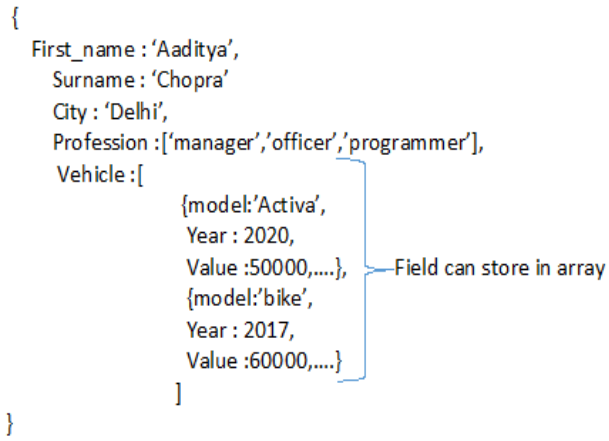

Fig. 4. Document base

2.4 Graph base database is a collection of edges and node as shown in (Fig 5).It stores entities as well the relations amongst those entities. The entity is stored as a node with the relationship as edges. An edge gives a relationship between nodes. Every node and edge have a unique identifier. Graph base databases mostly used for social networks, logistics, whether forecast.

Example : Neo4J, OrientDB

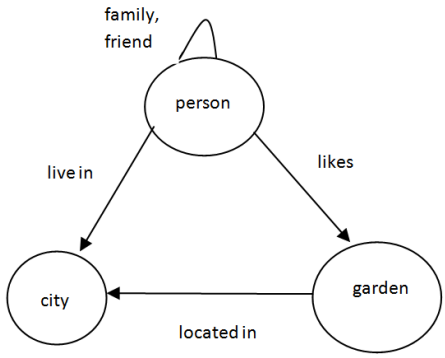

Fig. 5. Graph based

III. QUERY PROCESSING AND QUERY OPTIMIZATION TECHNIQUES

Query processing and optimization are the important factors of the database management system. The main function of query processor is to translate the query in a high level language into low level language. There are various ways to execute the same query, but the main aim is to reduce the execution time when the data is in large volume. The optimizer is require to examine the factor like joining the table, number of rows retrieve using join table and algorithm to be deploy for performing the join.

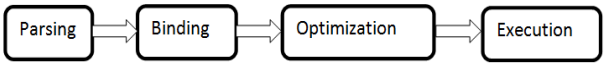

Fig. 6. Process of query optimization

Query Optimization is the process of selecting an efficient execution plan for evaluating the query. After parsing query is passed to the query optimizer. The query optimizer parses the query binding the other tables, join the table, applies the optimization techniques, and finally gets desired result with minimized response time[10].

In this paper, we discussed the various techniques used to optimize the query for searching data to minimize the response time.

3.1 Query Optimization Techniques.

- Bloom filter

A Bloom filter is a probabilistic data structure that allow to quick filtering of rows that cannot be joined before they would reach the join operator. Cassandra merges data on SSTable(disk) with data in mamtable (RAM). It improves the performance because irrelevant block do not have to be read.it is useful for queries and applied for numeric and non- numeric data.it is used for primary key and indexing. Bloom filter can only support lookup queries, not for range queries [13][14]. Google Chrome web browser and Gmail use bloom filter for identify the malicious URL and check whether the user name is already taken or not for billions of username respectively.

•Hashing

Hashing is the techniques that allows to search location of data without using index structure. It is used to search a specific data faster using special hash key. There are two type of hashing that is static and dynamic. Hashing is best for large database. Hash function is used one way hashing algorithm and hash cannot be converted into original key.

- Indexing

Indexing is the most powerful feature to speed up the execution of query. Query optimizer uses indexes to generate an execution plan when we preparing a DML statement. The value in index store in rowid of the physical data location in the database which is help for fast data retrieval. The following point to use the index in table for improve the retrieval of data is

– Joining columns of the tables.
– Columns uses ORDER BY clause.
– Columns uses WHERE clause.

Following Index are used for query optimization are bit-map index, hash index, function based index, LSM index,B-tree index[13-15].

– Bit-map index : Bit-map index are basically used for low cardinality columns, which have distinct value. This techniques is used for huge database and most frequently used in query with WHERE clause[14].

– Hash Index : It is commonly used in data management.

Hash index are used in primary key or unique identifier like email address, empno etc.The benefit of hash index is for fast searching and insertion of data. They work by splitting data in small chunk and then storing them. When you search data within database, it is much faster than other.

– Function based Index: function based index is used to create when we use functions or expression. Function-based indexes can involve multiple columns, arithmetic expressions. In that case function based indexing are used where index maintain the data values after using the function over the column[14].
Example: create index emp_idx on employee(UPPER(ename));

– LSM Tree Index : A Log Structured Merge Tree (LSM- tree) index is used for high insert volume such as write heavy workload. It is optimized to perform the sequential write.it is used for Cassandra, BigTable and RocksDB[13].

– B-Tree : B-tree is a data structure that provide to store and searching of data. It is highly capable of storage huge number of data. The B-tree simplify the binary serach tree by allowing node with more than two children. It provide an efficient way to insert and searching of data. Database use both B-tree and B+-tree together to store the data. B- tree is used for indexing and B+-tree is used to store the actual record and searching for sequential searching[13][14].

- Sharding

Sharding is the process of distributing data in different server. MongoDB support sharding for horizontal scalability. Types of sharding are range sharding, hash

sharding and group sharding[16].sharding is method of storing data across many server. Many Nosql database offer auto sharding. Each shard read and write its own data.

- Partitioning

  Partitioning is the concept of distributing data into different node instead of a single node. The purpose of partitioning is to reduce the query load, improve scalability and performance. Partitioning is to spread data so that execution can be distributed across the various nodes and improve the read performance. The different types of partitioning are horizontal partitioning (sharding), vertical partitioning, and functional partitioning shown in fig 7. [12][14].

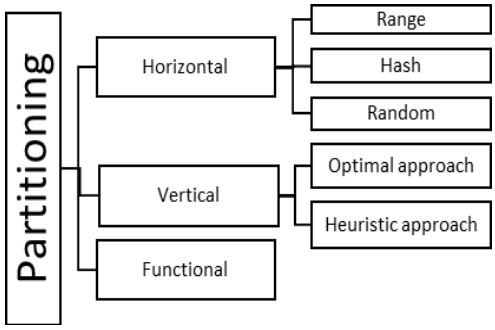

Fig. 7. Methods of partitioning

– Horizontal Partition : Horizontal Partition is also called sharding. Each partition is a separate data store but same schema. Each partition is also call as shard and it holds a specific data such as all order for specific set of customer. For example customer having name [A-H] have one shard and [I-Z] have another shard. It is use to reduce the traffic and improve the performance. Horizontal partition have three different types that are Range, Hash and Random.

– Vertical Partition: In this partition, each partition holds subset of fields for item in the data store. For doing partition ,fetching item that are frequently used in one partition and less frequently access in another partition. For example in product table, product id, product name, description and price that are frequently used is in one partition. Stock and order date are separated by other partition which are not used frequently. The advantages of vertical partition are that reduces the concurrent access. There are two type of vertical partitioning shown in fig. 7 that is Optimal approach and Heuristic approach

– Functional Partition: Functional partitioning is used to improve the isolation and data access performance. Another use is to separate read and write data from read only database. For example Inventory data is separated by customer data.

Table 1.Comparison between Cassandra, MongoDB

IV. ANALYSIS

The presented survey paper is analysed NoSQL data model and various optimization techniques. According to the table
1 about the comparison between Cassandra and MongoDB and table 2 is listed various query optimization techniques used in various literature survey.

Query	CQL	JSON
Performance	Best	Better
CAP Theorem	AP	CP
Consistency	Eventual consistency	Eventual consistency
Atomicity	Single Column (multicolumn updates may cause dirty writes)	Single Document
Key customer	Twitter, Instagram, Netflix	Catalogue performance, eBay, Google, Cisco, SAP
Technique[26]	Partitioning	Sharding
Uses	Use for concurrent read and write operation	Use for XML data

PROPERTY	Cassandra	MongoDB
Architecture	Wide Column Store	Document Store

Table 2.Various Techniques for query optimization

Techniques	Advantages	Limitation
Bloom Filter [9-11] [13][15]	High performance and Accuracy. Low complexity and low cost. Searching of data with less amount of	It does not support deletion.
Hashing[2][9][11][13][16][18]	Best use for large dataset. Use hash key for searching data in dataset. It provides better synchronization	It does not allowed for null value.
Index[4][6][7][9][11] [13-15][17][18]	Use to increase the efficiency for searching mechanism. Use primary index	It required extra space for indexing.
Partition[4][5][9] [11-	Improve the scalability and optimize	Scan all tables when find the data
Sharding [15][16][17]	Use to apply horizontal scalability and high availability.	Increase the power consumption and complexity of

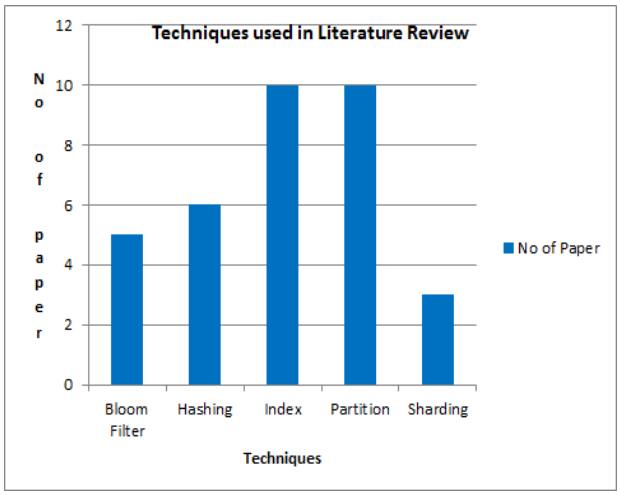

Fig. 8. Graph analysis according to Table [2]

According to the table [1] Cassandra perform better than MongoDB [15][16][19] and table[2] summarized that partition and indexing techniques are good for query optimization[16-20] as shown in above graph analysis Fig 8.

V. CONCLUSION

This paper presents a systematic review of query optimization in NoSQL and basic information about the query optimization process. This paper introduces the requirement of the NoSQL database, their different data model, and techniques like indexing, partitioning, replication, and query processing were discussed. This type of survey understands how big data were stored in a NoSQL data model as per customer needs. Discuss the comparison between MongoDB and Cassandra databases using various properties. Cassandra performs a better result than MongoDB databases for the query optimization process. Discuss some optimization techniques were list for quick response time with the benefits and weaknesses regarding the selected query optimization techniques have been investigated. According to these techniques indexing and partitioning is best for quick response time. Both approaches improve the retrieval performance. Therefore, future research may consider to develop an ensemble model for swarm optimization algorithm using optimization techniques in the Cassandra dataset and improve the concurrency control mechanisms.

REFERENCES

[1] A. H. Abed, "Recovery and Concurrency Challenging in Big Data and NoSQL Database Systems," Int. J. Adv. Netw. Appl., vol. 11, no. 04, pp. 4321–4329, 2020, doi: 10.35444/ijana.2020.11041.

[2] D. Kumar and V. K. Jha, "An efficient query optimization technique in big data using σ -ANFIS load balancer and CaM-BW optimizer," J. Supercomput., vol. 77, no. 11, pp. 13018–13045, 2021, doi: 10.1007/s11227-021-03793-6.

[3] D. Kumar and V. K. Jha, "An improved query optimization process in big data using ACO-GA algorithm and HDFS map reduce technique," Distrib. Parallel Databases, vol. 39, no. 1, pp. 79–96, 2021, doi: 10.1007/s10619-020-07285-z.

[4] E. Azhir, N. J. Navimipour, M. Hosseinzadeh, A. Sharifi, and A. Darwesh, "Query optimization mechanisms in the cloud environments: A systematic study," Int. J. Commun. Syst., vol. 32, no. 8, 2019, doi: 10.1002/dac.3940.

[5] M. H. Abdalla and M. Karabatak, "To Review and Compare Evolutionary Algorithms in Optimization

of Distributed Database Query," 8th Int. Symp. Digit. Forensics Secur. ISDFS 2020, pp. 2–6, 2020, doi: 10.1109/ISDFS49300.2020.9116418.

[6] F. Cores, F. Guirado, and J. L. Lerida, "High throughput BLAST algorithm using spark and cassandra," J. Supercomput., vol. 77, no. 2, pp. 1879–1896, 2021, doi: 10.1007/s11227-020-03338-3.

[7] S. Ravikumar and D. Kavitha, "A New Adaptive Hybrid Mutation Black Widow Clustering Based Data Partitioning for Big Data Analysis," Wirel. Pers. Commun., vol. 120, no. 2, pp. 1313–1339, 2021, doi: 10.1007/s11277-021-08516-x. [8] B. Jena, M. K. Gourisaria, S. S. Rautaray, and M. Pandey, "A survey work on optimization techniques utilizing map reduce framework in HADOOP cluster," Int. J. Intell. Syst. Appl.,vol. 9, no. 4, pp. 61–68, 2017,doi: 10.5815/ijisa.2017.04.07.

[9] T. Chawla, G. Singh, E. S. Pilli, and M. C. Govil, "Storage, partitioning, indexing and retrieval in big rdf frameworks: A survey," Comput. Sci. Rev., vol38, 2020, doi: 10.1016/j.cosrev.2020.100309.

[10] J. Song, Y. Bi, G. Han, and T. Li, "FacetsBase: A Key-Value Store Optimized for Querying on Scholarly Data," IEEE Trans. Emerg. Top. Comput., vol. 9, no. 1, pp. 302–315, 2021, doi: 10.1109/TETC.2018.2844313.

[11] M. S. Mahmud, J. Z. Huang, S. Salloum, T. Z. Emara, and K. Sadatdiynov, "A survey of data partitioning and sampling methods to support big data analysis," Big Data Min. Anal., vol. 3, no. 2, pp. 85–101, 2020, doi: 10.26599/BDMA.2019.9020015.

[12] C. Lehmann, L. Goren Huber, T. Horisberger, G. Scheiba, A. C. Sima, and K. Stockinger, "Big Data architecture for intelligent maintenance: a focus on query processing and machine learning algorithms," J. Big Data, vol. 7, no. 1, 2020, doi: 10.1186/s40537-020-00340-7.

[13] M. A. Qader, S. Cheng, and V. Hristidis, "A comparative study of secondary indexing techniques in LSM-based NoSQL databases," Proc. ACM SIGMOD Int. Conf. Manag. Data, pp. 551–566, 2018, doi: 10.1145/3183713.3196900.

[14] V. Neri and N. Sarajli, "a Review on Big Data Optimization Techniques," vol. 14, no. August, 2020.

[15] R. Dharavath, A. Kumar, and V. K. Dharavath, "Capturing Anomalies of Cassandra Performance with Increase in Data Volume: A NoSQL Analytical Approach," Lect. Notes Data Eng. Commun. Technol., vol. 37, pp. 3–20, 2020, doi: 10.1007/978-981-15-0978-0_1.

[16] E. A. Khashan, A. I. El Desouky, and S. M. Elghamrawy, "A Framework for Executing Complex Querying for Relational and NoSQL Databases (CQNS)," Eur. J. Electr. Eng. Comput. Sci., vol. 4, no. 5, pp. 1–11, 2020, doi: 10.24018/ejece.2020.4.5.195.

[17] D. Mahajan, C. Blakeney, and Z. Zong, "Improving the energy efficiency of relational and NoSQL databases via query optimizations," Sustain. Comput. Informatics Syst., vol.22, pp. 120–133, 2019, doi: 10.1016/j.suscom.2019.01.017.

[18] D. Preuveneers and W. Joosen, "Automated configuration of NoSQL performance and scalability tactics for data- intensive applications," Informatics, vol. 7, no. 3, 2020, doi: 10.3390/INFORMATICS7030029.

[19] S. Gupta and G. Narsimha, "Efficient query analysis and performance evaluation of the Nosql data store for bigdata," Adv. Intell. Syst. Comput., vol. 507, pp. 549–558, 2017, doi: 10.1007/978-981-10-2471-9_53.

[20] "Data partitioning guidance",https://learn.microsoft.com/en-us/azure/architecture/best-practices/data-partitioning.

[21] "The main NoSQL Database type",https://studio3t.com/knowledge-base/articles/nosql- database-types, last accessed 2021/08/04.

BIOGRAPHIES AND PHOTOGRAPHS

Heena Patel is an Assistant Professor at M.K Institute of computer Studies affiliated to Veer Narmad South Gujarat University, Bharuch. Currently pursuing Ph. D in computer science at Sardar Patel University. She has obtain Master of Computer application (MCA) from Veer Narmad South Gujarat University, Surat.
Her area of interest is data Analytics, data mining and Artificial Intelligence (AI).

Dr. Prashant P. Pittalia is Associate Professor at department of Computer Science, Sardar Patel University. He is member of Board of Studies in Computer Science & Technology at Gujarat Forensic Sciences University and International Association of Engineers (IAENG).
He has more than 53 research publications including International books, International journal papers, International Conferences, National Conferences / Seminars. He has guided Ph D students, M.Tech students for dissertation, MCA students for projects. His area of interest is computer network and cyber security.