International Journal on Future Revolution in Computer Science & Communication Engineering ISSN: 2454-4248
Volume: 7 Issue: 2 09 – 12

Medical Image Management by NoSQL Database

Arti Sharma[1], Dr. Rajesh Soni[2]
[1]Research Scholar, Bhupal Nobel's University, Udaipur, Rajasthan
[2]Associate Professor, Bhupal Nobel's University, Udaipur, Rajasthan

Abstract
Over the past three decades, medical imaging technology has revolutionized the healthcare sector. It enables medical professionals to detect disease early and improve patient outcomes. Large amounts of semi-structured and unstructured picture data that lack a set schema are produced as health records. Better data models for storage and retrieval are required. Big Data may be stored and retrieved much more easily with NoSQL (Not merely SQL) databases. Hence, the suggestion is to store medical health records in a NoSQL database. This essay aims to contrast several NoSQL databases from the perspective of image technology.

Keywords: NoSQL databases, Medical Imaging Technology, HBase, MongoDB, Cassandra.

*****

Introduction

The ability to store and retrieve medical images is crucial in the field of medical informatics, and data storage techniques have seen a significant transformation during the past ten years. A medical image will be used as part of every curative operation because it is a component of medical informatics.

Medical diagnostics and therapies rely heavily on medical imaging technologies. Therefore, the amount of saved medical images will significantly expand in the coming years. Experts predicted that in the future, medical picture data, particularly, will account for 30% of all global storage in the context of health informatics. According to research, the market for medical imaging equipment will reach $49 billion in 2020[1, 2]. Basic and advanced medical imaging systems are divided into two categories. Examples of fundamental modalities include ultrasound, general X-ray, and mammographic X-ray. Advanced imaging modalities include molecular imaging, magnetic resonance imaging (MR), and computed tomography (CT). Finding a better method that facilitates effective medical picture archiving is essential to lowering the complexity and expense of medical image storage [3]. The volume of medical image data now kept has crossed the 1 Exabyte threshold, bringing medical imaging into the Big Data universe [4].

Several big data technologies have been created in recent years to handle large amounts of data. Due to their volume and variety, medical images fall under the big data category [5, 6]. Because different types of data require different methods of storage and retrieval, NoSQL was developed. Examples of free and open-source NoSQL databases include Terrastore, Neo4j, Couch DB, Cassandra, and Couch DB. These databases can manage vast amounts of semi-structured and unstructured data in addition to structured data. They offer fast data read and write speeds. The scholarly community and the IT sector are utilizing these databases. Unstructured data such as medical images can be maintained using NoSQL databases without a doubt. In this study, we analyze and evaluate the storage of medical images using the three NoSQL databases HBase, MongoDB, and Cassandra.

The format of this essay is as follows: In Section II, we covered the current system for picture storage and retrieval that uses No SQL. Section III discusses the need for and benefits of using NoSQL to store photos. Additionally, we contrasted the features of several NoSQL database types. Then, in section V, we discussed the system setup details and experiment findings. In section IV, we also projected the implementation details of image storage and retrieval linked to HBase, MongoDB, and Cassandra. Section VI provides our conclusions at the end.

Existing Systems Survey

NoSQL is adopted by the industry so rapidly. Under several scientific and research contexts [7,8], It had been compared with the relational property. Rascovsky et al. proposed and implemented a CouchDB-based medical archiving system. The authors recommend that document databases are extremely significant to store and retrieve DICOM files. Also suggested using Document databases to store the metadata of DICOM images [8,10], In

[8] Luís A. Bastião Silva et. al. compared MongoDB and CouchDB in medical images storage and retrieval and concluded that Mongo DB performance was better than CouchDB. D.R.Rebecca and Dr.I.E. Shanthi [9] compared the storage and retrieval of Medical images in MYSQL with Mongo DB. The results show that the Mongo DB performance was better than MySQL. Also proved that NoSQL is well suitable for storing unstructured data. Generally, Medical images are used to store by stored using an RDBMS-based solution called. Various disadvantages of storing medical images in an RDBMS-based archive are

discussed in [9, 10]. In [10], it is proved that Mongo DB performed better. In the context of medical image storage and retrieval, we definitely to compare the different NoSQL databases

Handling Medical Images

The NoSQL databases are ideal for storing photos of greater size. The providers of medical images must be transferred to the cloud It is now necessary for medical imaging providers to move their storage needs to the cloud [13]. RDBMS-based storage solutions are a poor fit for the entire medical imaging situation [9, 10], as opposed to cloud-based storage systems. [11, 12]. Finding an improved NoSQL database that saves medical images efficiently is necessary. Three NoSQL databases' performances are compared in this research.

Data for medical images are gathered from several heterogeneous systems. It is massive in volume and comes in the forms of structured, semi-structured, and unstructured data. This type of data cannot be handled by any SQL database. Since they are not relational, they do not adhere to a specific schema. It uses a more adaptable data model.

NoSQL databases have several advantages over RDBMS, including:

I. High availability with redundancy in various locations.
II. It runs across many data centers and is cloud-enabled.
III. It has a high write speed and low latency read.
IV. It is extremely expandable and simple to add more storage and processing power.

The amount of data is challenging to keep and handle on a single system or cluster. The following design objectives were set to successfully and efficiently process large images using a cluster of inexpensive hardware:

- Rather than using a centralized system, data storage can be dispersed among several machines. Large files can be divided up into smaller ones and stored over several nodes.
- Data should be stored in a flexible schema structure that can be quickly modified when necessary.
- To create the results for efficient bandwidth use, data processing must be performed on isolated subsets and joined with them after processing.

Key-Value stores, column-oriented data stores, document-oriented data stores, and graph databases are the different types of NoSQL databases. Table 1 shows the comparison between different types of NoSQL databases in terms of performance, scalability, flexibility, and complexity.

Table 1 Feature comparison of No SQL databases

Data model	Performance	Scalability	Flexibility	Complexity
Key–value store	high	high	high	none
Column-oriented store	high	high	moderate	low
Document-oriented store	high	variable(high)	high	low
Graph database	variable	variable	high	high

Column-oriented store varies from row-oriented databases concerning a) performance b) storage and c) ease of schema modification

Implementation:

The following is the code implementation for storing and retrieving images from various databases:

I. Apache Cassandra: This distributed database has no single point of failure and is extremely scalable and available [14]. It is built to handle big amounts of data across numerous servers and offers high performance. If huge objects are stored in Cassandra improperly, it may result in high heap pressure and hot spots. To simultaneously store the image in the Cassandra database by breaking it into portions, Netflix offers the Astyanax API. By breaking up huge objects into several keys and handling fetching them in random order to reduce hot spots, it provides utility classes that address these difficulties. The sample code presented in Fig. should be used to store the image.

```
ObjectMetadata imgMd = ChunkedStorage.newWriter(
          cassandraChunkedStorageProvider,
          objName, imageInputStream)
          .withChunkSize(0x1000)
          .withConcurrencyLevel(8)
          .withTtl(60).call();
```

```
ObjectMetadata imgMd =ChunkedStorage.newInfoReader( cassandraChunkedStorageProvider,
objName).call();
ByteArrayOutputStream os = new ByteArrayOutputStream( imgMd.getObjectSize()
.intValue());
imgMd = ChunkedStorage.newReader(cassandraChunkedStorageProvider, objName,
byteArrayOutputStream).withBatchSize(11)
.withRetryPolicy(new ExponentialBackoffWithRetry(250,20))
.withConcurrencyLevel(2).call();
```

Sample code to retrieve the image in Cassandra

II. **Apache HBase:** Open source, non-relational, distributed database modeled database which is developed based on Google's Big Table Concept. It fits key-value workloads with high-volume random reads and writes access patterns used for basic use cases. It serves the image files by either storing them in itself or storing the image. Handling more images is complex and it depends on the name node memory size [17]. The column qualifiers of the single column are data and type where the data column could store either the path or the actual image bytes and type would store the image type (png, jpg, tiff, etc.). It helps for sending the correct mime type over the wire when returning the image. Cloudera's Kestelyn mentioned that "HBase provides a record-based storage layer that enables fast,

 ISSN: 2454-4248
09 – 12

random reads and writes to data, complementing Hadoop by emphasizing high throughput at the expense of low-latency I/O.

Sample code to store the image in HBase Configuration conf = HBaseConfiguration.create();HTable table = new HTable(conf, "test".getBytes());Put put = new Put("row".getBytes());
put.add("C".getBytes(),
"img".getBytes(),extractBytes("/tmp/sample/input.jpg"));
table.put(put);

Sample code to retrieve image from HBase
Get get = new Get("row".getBytes());Result result = table.get(get);
byte[] arr = result.getValue("C".getBytes(),
"img".getBytes());

OutputStream out = new BufferedOutputStream(new FileOutputStream("/tmp/sample/output.jpg"));
out.write(arr);

III. **MongoDB**: It is a cross-platform document-oriented database system that deals with JSON-like documents with dynamic schemas. It integrates data as BSON (Binary Simple Object Notation) in certain types of applications simpler and faster.

There are many ways to store images in the database. one is saving them in a database as blob type and anotheris with a folder structure that can retrieve them in a fast and efficient way. MongoDB uses a GridFS file system [16] to store and efficiently retrieve images or files. Below some code depicts the way to store and retrieve images using GridFS

GridFS gfsImg = new GridFS(mongoTemplate.getDb(),
"img");GridFSInputFile gFile = gfsImg.createFile(content);
gFile.setFilename(fileName);
gFile.setContentType(contentType);
gFile.save(); return gFile.getId()
Reading the image from the database is as easy as saving it. We require the identity of the image. Inthis case, we assume that we have the ID.
GridFS gfsPhoto = new GridFS(mongoTemplate.getDb(),
"img");

GridFSDBFile img = gfsPhoto.findOne(new ObjectId(id));
InputStream stream = img.getInputStream();

Experiments and Discussion:
The Experiments are tested on Ubuntu distribution with 8GB RAM and 1TB hard disk runs on an intel core i5 processor. The experimental code is implemented in JAVA. The databases used for the analysis are HBase 1.2.6, MongoDB 3.4.9, and Cassandra 3.11.1. We have tested the setup by storing and retrieving the images varies in size from 1 MegaByte to 100 Mega Bytes. The mechanism to store and retrieve images related to three databases is mentioned below with code.
Endpoint has performed various experiments [17] on three different NoSQL Databases which are running on Amazon Web Services EC2 instances. For each experiment, a new EC2 instance is used to reduce the impact of any “lame instance” or “noisy neighbor” in cloud environments. In the performance view, there is no single winner among the NoSQL databases or processing engines. It is completely depending on deployment and use cases. Fig 1 and 2 show the comparison between NoSQL databases with analogous and diversified operations.

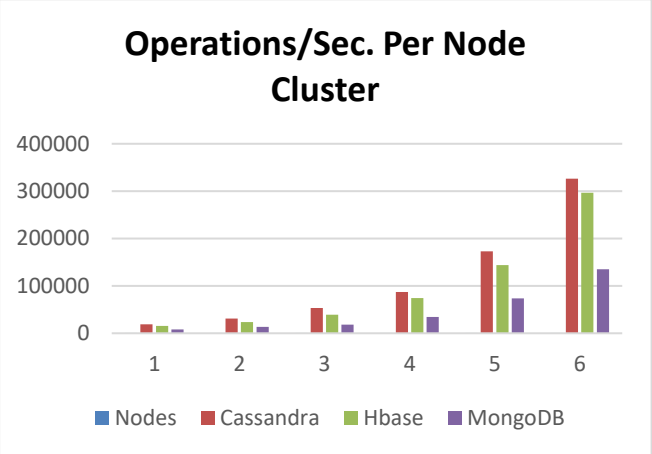

Fig 1- No SQL databases with analogous operations

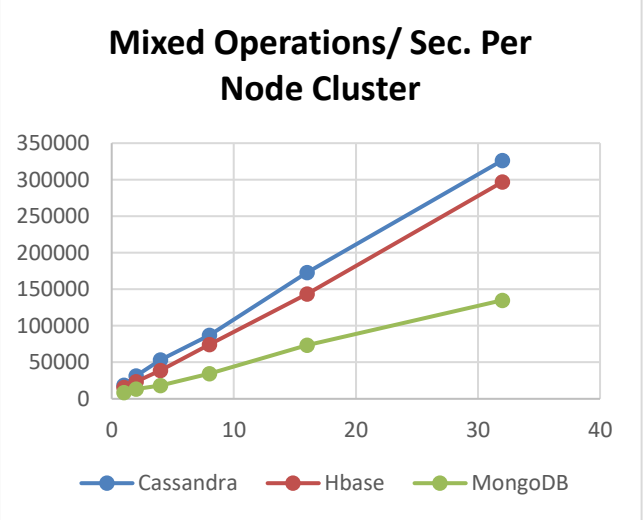

Fig 2- No SQL databases with diversified operations

It shows workload with the throughput/operations-per-second in Y- the axis and the number of nodes used is presented in X-axis. Tables 1 and 2 show the results of each graph. Cassandra is the only database performing durable write operations in HBase, and MongoDB performs non-durable write operations. The below results show that Cassandra performed 195 times faster than Mongo DB and six times faster than HBase for varied operational and

analytic workloads.

Table 1 – Comparison between NoSQL databases with the same operations

Nodes	Cassandra	Hbase	MongoDB
1	18683.43	15617.98	8368.44
2	31144.24	23373.93	13462.51
4	53067.62	38991.82	18038.49
8	86924.94	74405.64	34305.3
16	173001.2	143553.4	73335.62
32	326427.07	296857.4	134968.9

Table 2 – Comparison between NoSQL databases with diversified operations

Nodes	Cassandra	Hbase	MongoDB
1	4690.41	269.3	939.01
2	10386.08	333.12	30.96
4	18720.5	1228.61	10.55
8	36773.58	2151.74	39.28
16	78894.24	5986.65	377.04
32	128994.91	8936.18	227.8

Conclusion:

The strategies that are currently used to manage medical photographs have been discussed in this paper. HBase, MongoDB, and Cassandra are three different types of databases with which we have tested. And contrast them in terms of read and write operation times. The findings of this investigation indicated that Cassandra, among the top NoSQL databases, has quick write and read performance [14] and good linear scale performance.

References

[1]. A DBA's Guide to NoSQL, Apache Cassandra, Datastax-2014.
[2]. Hari Krishna Timmana and Dr C Rajabhushanam," Mininet Implementation of SDN towards Network Softwarization", International Journal Of Innovative Research In Management, Engineering And Technology,Vol. 2, Issue 5, May 2017
[3]. Shashank tiwari, professional Nosql, John Wiley & Sons, Inc. 2011 Available at http://www.frost.com/prod/servlet/press-release.pag?docid=268728701
[4]. Frost & Sullivan: U.S. Medical Imaging Informatics Industry Reconnects with Growth in the Enterprise Image Archiving Market
[5]. Dan McCreary,Ann Kelly, Making Sense of NoSQL Databases, Manning Publications,2014.
[6]. N. V. Chawla and D. A. Davis, "Bringing big data to personalized healthcare: A patient-centered framework," Journal of general internal medicine, vol. 28, pp. 660-665, 2013
[7]. Rajat Aghi, Sumeet Mehta, Rahul Chauhan, Siddhant Chaudhary and Navdeep Bohra,A comprehensive comparison of SQL and MongoDB databases,International Journal of Scientific and Research Publications, Volume 5, Issue 2, February 2015,ISSN 2250-3153
[8]. Luís A. Bastião Silva, Louis Beroud, Carlos Costa and José Luis Oliveira,Medical imaging archiving: a comparison between several NoSQL,978-1-4799-2131-7/14/$31.00 ©2014 IEEE
[9]. D.Revina Rebecca, I.Elizabeth Shanthi,A NoSQL Solution to efficient storage and retrieval of Medical Images,International Journal of Scientific & Engineering Research, Volume 7, Issue 2, February 2016,ISSN 2229-5518
[10]. Simón J. Rascovsky, MD, and et.al,Use of CouchDB for Document-based Storage of DICOM Objects
[11]. Katarina Grolinger1, Wilson A Higashino, Abhinav Tiwari and Miriam AM Capretz,Data management in cloud environments: NoSQL and NewSQL data stores,journal of cloud Computing, Springer Open journal, 2013
[12]. J.Antony John Prabu, Dr.S Britto Ramesh Kumar,Issues and Challenges of Data Transaction Management in Cloud Environment
[13]. http://docs.datastax.com/en/cassandra/2.1/cassandra/gettingStartedCassandraIntro.html
[14]. https://github.com/Netflix/astyanax/wiki/Chunked-Object-Store
[15]. "Cassandra vs. MongoDB vs. Couchbase vs. HBase" available at https://www.datastax.com/nosql-databases/benchmarks-cassandra-vs-mongodb-vs-hbase
[16]. Mongo DB api site Available at http://api.mongodb.com/
[17]. Apache HBase Documentation Available at http://hbase.apache.org/
[18]. Sivaraman, K., Dr. K.P. Kaliyamurthie, Cloud Computing in Mobile Technology, Journal of Chemical and Pharmaceutical Sciences, 2016
[19]. Priya. N, Sridhar. J, Sriram M, Mobile large data storage security in cloud computing environment- a new approach, Journal of Chemical and Pharmaceutical Sciences, Vol. 9, Issue 2, April-June 2016