International Journal of Multidisciplinary Research and Development
Online ISSN: 2349-4182, Print ISSN: 2349-5979; Impact Factor: RJIF 5.72
Received: 21-05-2019; Accepted: 24-06-2019
www.allsubjectjournal.com
Volume 6; Issue 7; July 2019; Page No. 73-78

The battle between NoSQL Databases and RDBMS

Sourav Mukherjee
Senior Database Administrator & PhD Student at University of the Cumberland's Chicago, United States

Abstract
NoSQL is a free and open-source, scattered, extensive column store database management system intended to handle large amounts of data across many product servers, providing high obtainability and accessibility with no single point of failure. It is the easiest truly big-data database that can scale and replicate data globally in a master-less configuration. A NoSQL database delivers a mechanism for storage and recovery of data that is demonstrated in means other than the tabular relations used in relational databases. NoSQL databases usually understood by engineers as 'not only SQL databases' neither 'no SQL', it is an alternative to the most widely used relational databases. As the given name proposed, it is a substitute for SQL that uses in such a way that the SQL is co-existed. A relational database management system (RDBMS) is a database management system based on the relational model of data and it is a completely structured way of storing data. But NoSQL is an unstructured way of storing data. Is NoSQL an RDBMS? This is an article we will discuss various features of NoSQL, various advantages over RDBMS and the future of NoSQL. This article also reviews the basics of NoSQL and its usages.

Keywords: apache, Cassandra, SQL, NoSQL, RDBMS, API

Introduction

Is NoSQL an RDBMS?

NoSQL is not a relational database management system. NoSQL or 'Not only SQL' is a non-relational database which supports a very simple query language with no fixed schema. NoSQL is the easiest truly big-data database that can scale and replicate data globally in a master-less configuration. This database shelters the storage of non-related data. NoSQL databases have a distributed structure. It can grip of the data in a very high volume an also at high speed. NoSQL database is organized in a horizontal manner. Few examples of NoSQL database are Cassandra, HBase, Couchbase, Cosmos DB, etc.

RDBMS essentially uses SQL or Structured Query Language. This database supports a controlling query language with a fixed schema which covers the storage of related data. This database has a compacted and unified structure and is organized vertically. RDBMS handles a restrained volume of data at low speed. Few examples are RDBMS, MongoDB, etc.

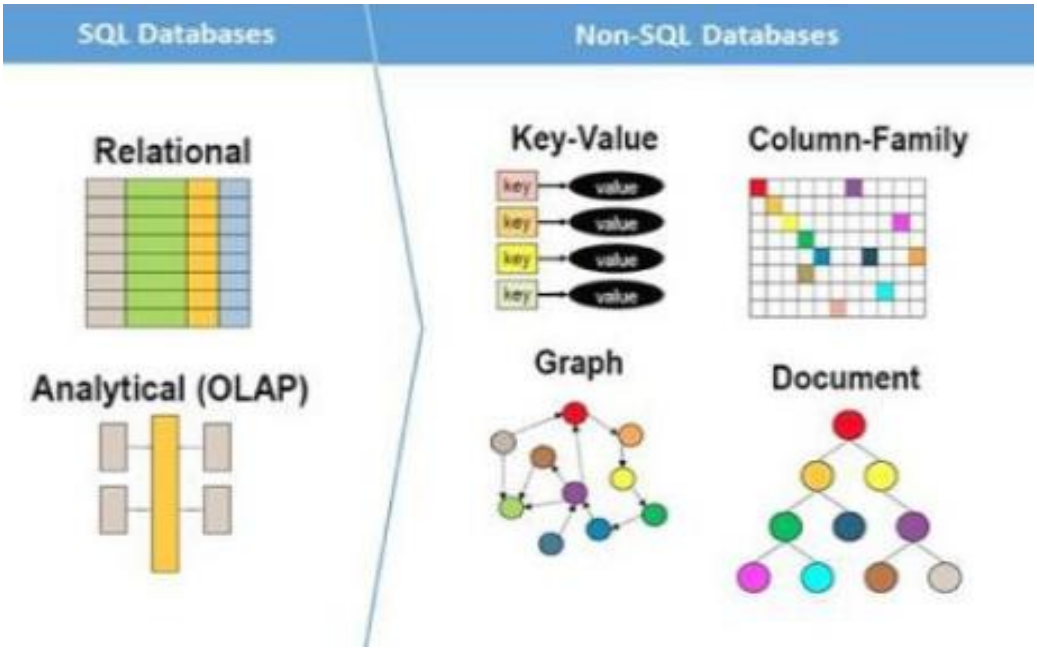

Fig 1: SQL vs NoSQL database architectures

Table 1

NoSQL	RDBMS
NoSQL is used to deal with unstructured data.	RDBMS is used to deal with structured data.
NoSQL has no fixed schema.	RDBMS has a fixed schema.
In NoSQL, the row is a unit of replication.	In RDBMS, the row is a specific record.
In NoSQL, relationships are described using collections.	In RDBMS, there are concepts of primary and foreign keys, joins etc.
In NoSQL, a table is a list of "nested key-value pairs". (Row x Column Key X Column value)	In RDBMS, a table is an array of arrays. (Row X Column)
In NoSQL, a column is a unit of storage.	In RDBMS, the column represents the attributes of a relation.
In NoSQL, tables or column families are the entity of key-space.	In RDBMS, tables are the entities of a database.
In NoSQL, key space is the furthermost container which includes data corresponding to an application.	In RDBMS, the database is the furthermost container which contains data corresponding to an application.

Literature Review or Background

NoSQL databases have developed in recent years as a response to the limitations of relational databases and to deliver the performance, scalability, and flexibility essential to the modern applications. Most features of these NoSQL skills differ significantly and have slight in mutual except for the fact that they do not use a relational data model.

NoSQL can be categorized into five types mentioned below.

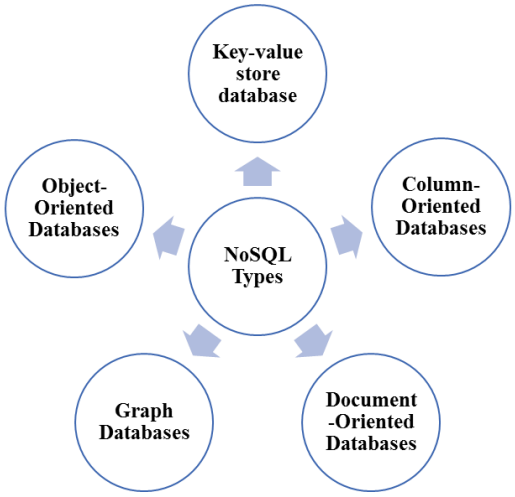

Fig 2: Types of NoSQL databases

1. Key-value store database

The key-values store database is very well-organized and devastating model. It can easily communicate to API (Application Programming Interface). API is an application programming interface is a set of procedure definitions communication protocols and tools for building software. The key-value data can be stored in an appearance that schema may not be needed and data can be stored in tables based on their data types of the programming language or an object. The data has as its main feature and divided into two parts, a string which signifies the key and the actual data which is to be associated as value thus generating a key-value pair. The values are stored in hash tables where the keys are the indexes which makes it faster than RDBMS. Hence the data model is very unexacting and easily manageable. The data is stored with high extensibility over reliability and so the querying features like joins and collective processes have been excluded. One of the drawbacks for key-value store database is since there is no schema, it is difficult to create custom views. Key-values data storage mainly used in user's session or shopping cart, to get the list of favorite products saved, websites, forums, online shopping, etc. Some of the examples of a key-value store database are Amazon Dynamo DB, Riak, etc.

Amazon Dynamo DB: It is a NoSQL database offered by Amazon which is highly predictable and scalable.

- Fast and flexible NoSQL database service for all applications that need consistent, single-digit millisecond latency at any scale.
- Supports both document and key-value store models.
- Flexible data model, reliable performance, and automatic scaling of throughput capacity make it a great fit for mobile, web, gaming, ad tech, IoT, and many other applications.
- Fully Managed. No longer required to have any concern about database management tasks.
- Fine-grained Access Control. It integrates with the AWS Identity and Access Management for fine-grained access control.
- Event Driven Programming. Integrates with AWS Lambda to provide Triggers which enables to architect applications that automatically react to data changes.
- Highly Scalable. Automatically scales capacity up or down, an application request volume increases or decrease.
- Dynamo DB Accelerator (DAX) is a fully managed, highly available, in-memory cache. Can reduce response times from milliseconds to microseconds, even at millions of requests per second.

Riak: Riak is a scattered NoSQL key-value data store that advances high accessibility, liability tolerance, effective effortlessness, flexibility, and adjustability. In addition to the open-source version, it originates in a reinforced enterprise version and a cloud storage version.

- Unmatched flexibility beyond typical 'high availability' contributions.
- Advanced technology to ensure data accuracy and never lose data.
- The immense scale on product hardware.
- Common code basis with true multi-model support.
- Allocates data across the cluster to ensure fast performance and fault-tolerance.
- Multi-cluster replication ensuring low-latency and robust business steadiness.
- Faster reads and writes making it easier to store, query, and analyze time and location data.

It can be used for the following purposes:

- Handling individual information of the user for social networking websites or MMORPGs (Massively Multiplayer Online Role-Playing Games).
- To collect checkout or POS (Point of sales) data.
- Managing Factory control and Information systems
- Building Mobile Applications on the cloud etc.

Riak should be evaded for highly centralized data storage projects with secure, fixed data structures. Riak is used by Mozilla, AOL, and Comcast.

2. Column-Oriented Database

Column-oriented database stores data in a table by column instead of rows. Column stores in NoSQL are composite row-column storage, unlike pure relational column databases. Although column-oriented database concept of column-by-column storage of columnar databases and columnar extensions to row-based databases, in NoSQL column-oriented stores, do not store data in tables. It stores data in enormously scattered designs and architectures. In a column-oriented database, each key is associated with one or more attributes (columns). The column-oriented database provides high flexibility and scalability in data storage. These types of database are suitable for data mining and analytical applications. Some of the distinguished DBaaS providers using column-oriented Databases are mentioned below.

Cassandra: Apache Cassandra is the easiest truly big-data database that can scale and replicate data globally in a master-less configuration. What used to be in the hands of only the biggest in Silicon Valley is now available as a mature database to the masses. Originally created at Facebook after they studied Amazon's Dynamo DB and Google's Big Table whitepapers, the Cassandra we know today is very different and has far surpassed its ancestors in feature set and has now become a popular wire-protocol for other databases such as Scylla DB, Yuga Byte, and Azure's Cosmos DB.

Apache Cassandra is written in Java language. Why it is chosen to be written in Java may be because the security is a prime concern it is developed in Java rather than in C++. Another key reason could be Performance. It might be slower at the startup, but once the code is ready and in running state it is way faster as compared to C++. Java code is continuously optimized by the JVM and in that consideration, it appears faster to C++. It may have other reasons as well such as advanced memory optimization or efficient garbage collection.

There are different flavors of Apache Cassandra available in the market,

- *Scylla DB* is an open-source distributed NoSQL data store which was intended and designed with Apache Cassandra while achieving expressively higher throughput and lower latencies. It's written in C++. *YugaByte* DB is a transactional and high-performance distributed database for building large-scale scattered cloud services. It also supports APIs which are Cassandra compatible and Redis compatible, with PostgreSQL in the Beta stage. YugaByte DB core is written in C++, but the repository contains a Java-based code that is needed to run sample applications.
- *Data Stax* Enterprise offers Apache Cassandra flavor in a database platform which is built knowingly for providing performance and availability demands of IOT, Web and Mobile applications. It gives organizations a safe always-on database that effects operationally simple when scaled in a single or across multiple data centers and in the clouds. Cassandra and Data Stax Enterprise have helped the customers supporting multi-datacenter and hybrid cloud deployments since the beginning. It is written in Java.

Let's understand the merits of using Apache Cassandra!

Cassandra

is a unique platform to handle huge amount of unstructured data at scale. If you're trying to make your relational database faster and reliable, Cassandra may be your ultimate companion. It combines the Amazon's Dynamo storage system along with Google's big table model, offering the near-constant availability required to support real-time querying for web and mobile apps.

- Cassandra can handle even the most massive datasets.
- It can work as amazing, record-setting reliability at scale.
- Eventual consistency yields high availability.
- It offers Wide-column flexibility.
- It also offers minimal administrative tasks at scale.
- It offers easy setup and maintenance (does not matter how big the dataset that you are setting)
- Flexible parsing and wide column requirements.
- Not with multiple secondary indexes.
- It allows applications to write into any node anywhere and anytime.
- Automatic workload management and data balancing across the nodes
- Linearly scalable by just adding more nodes to the cluster.

On the other hand, it may not be a proven benefit if,

- Your app requires transactional operations,
- It requires dealing with financial data,
- Need dynamic queries against column data,
- Low latency requirement,
- Read exceeds write by a large margin
- On-the-fly aggregations & joins and so on...

It is hard to find which large-scale organization does not use Cassandra nowadays. When dealing with distributed databases, it is always the key requirement to identify how the data and the workload will be distributed. Correspondingly, the data model must be correctly designed. For example,

- Not letting the partition key too large,
- A specific size of the tables,
- Keeping an ideal same partition size, etc.

The most important point to highlight is even though distributed databases falls under the category of the database, however, treating this application to behave like a traditional relational database may incur excessive performance degradation and it may break the application as well. So, we must have to be careful while designing the application.

Bigtable: Goggle's big table is high performance and compressed data storage system which is built on Google file system, Chubby Lock Service, SS Table and few other Google technologies. Big table also inspires Google Cloud Data store, which is accessible as a part of the Google Cloud Platform. Big table is used by many Google applications such as web indexing, Google Maps, Google Book search Google Earth, Google Code, YouTube, Gmail, etc. Google has developed its own database to increase scalability and

performance. Google's Spanner RDBMS is covered on an application of big table with a Paxos group for two-phase commits to each table. Google F1 was built using Spanner to substitute an operation based on MySQL.

Big table is one of the perfect examples of a wide column store. Big table is designed to add thousands of machines and it is easy to add more machines.

- Three main mechanisms: the library, the master server, and many tablet servers. The library is connected to many clients, the master server manages the schema changes, tablet servers manages the tables.
- It is not a relational database and can be well defined as a light, scattered multi-dimensional organized map.
- Big table is used to scale into petabyte mode.
- When table size portends to raise beyond a definite limit, the tablets may be compressed using the algorithm BMDiff and the Zippy compression algorithm.
- It provides steadiness, reliability, fault tolerance and tenacity.

3. Document-Oriented Databases

Document-oriented databases also called document store databases is a subset of a type of NoSQL database. It is designed for storing, retrieving and handling document-oriented information. Document-oriented databases are characteristically a subclass of the key-value store, another NoSQL database concept. This offers great performance and horizontal scalability choices. The data stores in a form of documents and the storage are like records but the data are more flexible as there are no uses of schemas. Documents can be stored in the form of PDF. JSON, XML etc. Document storage in the document-oriented database is complex because it stored in key-value pairs also known as key-document pairs. Documents may have special characters in it. It's easy to fetch the data if documents are portioned across some documents. Some of the document-oriented databases provide are MongoDB, CouchDB, etc. MongoDB: an open source distributed document database, highly optimized for JSON. Apache CouchDB: an open source, Erlang based, database with a REST full HTTP API.

MongoDB

MongoDB is a document-oriented database program that can be implemented on multiple computer platforms. This is classified as a NoSQL database program. 10gen Software Company started developing MongoDB in 2007 as a planned platform as a service product and during 2009 it was initially released as an open source development model. During 2013 the 10Gen changed the name to MongoDB. C++ was used to develop MongoDB as a high performance and effective database.

- MongoDB is highly optimized for JSON, it stores data in flexible JSON-documents that means the columns may vary document to document and the data structure may be reformed over time.
- Easy to work with as the object mapping is done by the document model in the application code.
- The real-time aggregation, the indexing, and queries give significant ways to access and examine the data.
- MongoDB is a scattered database at its fundamental, so high obtainability, horizontal scaling, and topographical circulation are built in and can be used easily.
- Absolutely free to use, the versions are released before October 16, 2018, are available under the AGPL. All versions are released after October 16, 2018, including patch fixes for prior versions, are published under the Server-Side Public.
- MongoDB provides end-to-end security.
- Management tools are available for automation, monitoring, and backup.
- Fully elastic database as a service with built-in best practices.
- High accessibility through built-in replication and failover.

CouchDB

Apache CouchDB is open-source database software that emphases on ease of use and having a scalable architecture. This also has document-oriented NoSQL database architecture. CouchDB was first introduced in 2005. Later it became the Apache foundation project in 2008. Each CouchDB is a group of independent documents. Unlike a relational database, CouchDB never stores data and relationships in tables. Each document preserves its own data and uses its own schema. Like MongoDB, CouchDB is developed using C++. The data storing can be done in JSON format and database with a REST ful HTTP API.

- http-based REST interface using which documents can be easily created and managed.
- Easy to set up with multiple nodes. Easy repetition of a database across multiple server instances.
- Gets data in the form of JSON format and to store the data it takes only a few minutes.
- The data stores in a format that space is not wasted leaving empty fields in the documents.
- When the frontend editing option is available, it is possible to set up an application very fast for loading and managing data.
- CouchDB has flexible schema designs, fast indexing, and retrieval of data.

4. Graph Databases

Graph databases are NoSQL databases which use to store data as a graph. This data model encompassed of vertices, which is an object such as a person, place, pertinent section of data and edges, which signify the connection between two nodes. The graph also entails of possessions related to nodes. The associations permit data in the store to be connected straightly and mostly recovered with one operation. Graph databases hold the relations between data as a priority. Querying relations within a graph database is fast because they are eternally stored within the database itself. Relations can be instinctively imagined using graph databases, making it beneficial for deeply inter-connected data. In the graph database, every node consists of a direct pointer which points to the adjacent node. Millions of records can be traveled using this method. Graph databases offer schema-less and effective storage of semi-organized data. The queries are articulated as traversals, therefore creating graph databases are quicker than relational databases. While the graph model clearly lays out the addictions between nodes of data, the relational model and other NoSQL database models connect the data by implicit ways.

Some of the Graph databases and the language it supports are listed below –

- Allegro Graph (Language: C#, C, Common Lisp, Java, Python)

- Amazon Neptune
- Anzo Graph (Language: C#, C)
- Arango DB (Language: C++, JavaScript.NET, Java, Python, Node.js, PHP, Scala, Go, Ruby, Elixir)
- Data Stax Enterprise Graph (Language: Java)
- Infinite Graph (Language: Java)
- Janus Graph (Language: Java)
- Mark Logic (Language: Java)
- Microsoft SQL Server 2017 (Language: SQL/T-SQL, R, Python)
- Neo4j (Language: Java, NET, JavaScript, Python, Ruby)
- Open Link Virtuoso (Language: C, C++)
- Oracle Spatial and Graph; part of Oracle Database (Language: Java, PL/SQL)
- Orient DB (Language: Java)
- SAP HANA (Language: C, C++, Java, JavaScript & SQL-like language)
- Spark see (Language: C++)
- Sqrrl Enterprise (Language: Java)
- Teradata Aster (Language: Java, SQL, Python, C++, R)

Amazon Neptune: Fast, Reliable, Fully-managed graph database service that makes it easy to build and run applications that work with highly connected datasets.

- Fast, reliable, fully-managed graph database service makes it easy to build and run applications with highly connected datasets.
- The core of Amazon Neptune is a purpose-built. High-performance graph database engine that is optimized for storing billions of relationships and also to query the graph with milliseconds latency.
- Highly available, with read replicas, point-in-time recovery, continuous backup to Amazon S3, and replication across Availability Zones.
- Secure, with support for encryption at rest and in transit.
- Supports open graph APIs for both Gremlin and SPARQL and provides high performance for both graph models and their query languages.
- Highly available, durable, and ACID (Atomicity, Consistency, Isolation, Durability) compliant.
- Fully Managed. User needn't worry about database management tasks such as hardware provisioning, software patching, setup, configuration, or backups.

5. Object-Oriented Databases

In the object-oriented database, the data or information is stored in the form of objects. It is different from the Relational database as this is not table-oriented. Object-oriented database management systems are also known as object database management systems syndicate database competences with object-oriented programming language competences. Object-oriented database management systems let the object-oriented programmers build the product, load them as objects and modify the existing objects to make a new object. Accessing data in the object-oriented database are comparatively faster as an object can directly be retrieved from its pointer. Some object-oriented databases are intended to work fine with object-oriented programming languages such as C++, C#, Python, Ruby, Delphi, Perl, JavaScript, Java, Visual Basic. NET, Objective-C, and Smalltalk, etc. This database can be used in the applications relating to complex object relationships, modifying object structures when the application describes collections. Scientific research, telecommunication, software that analyze, optimize and design are some places where object-oriented databases are mainly used. The scalability becomes problematic when the physical memory size exceeded in Object-oriented database. Some of the Object-Oriented Databases are listed below –

- db4o
- Gem Stone/S
- Inter Systems Caché
- JADE
- Object Database++
- Object DB
- Objectivity/DB
- Object Store
- ODABA
- Perst
- Open Link Virtuoso
- Versant Object Database
- ZODB

Advantages of Object-Oriented Databases

- The object-oriented database lets the data, information, components, Products to distribute effortlessly.
- This allows integration easily the databases, operating systems, spreadsheets, other systems like AI, objects, and applications, etc.

Advantages of NoSQL

- NoSQL databases are more scalable than SQL databases.
- NoSQL is also very flexible. The scalability and flexibility of NoSQL combined with the rich functionality.
- NoSQL is a fully automated and managed the database.
- It has a flexible data model.
- Fast performance: Quick retrieval of files and fast disk-based storage.
- Flexible database service for all applications that require consistent, single-digit millisecond latency at any scale.
- It supports horizontal scalability for read and writes.
- Non-relational and distributed database system.
- NoSQL has dynamic schemas.
- NoSQL is appropriate for hierarchical data storage.
- Beneficial for complex queries.
- NoSQL is eventually consistent as there is no need to worry about deadlock.
- If index data is required, the view is the best option as the view can automatically index data.

Disadvantages of NoSQL

- Some of the NoSQL databases are not ACID (Atomicity, Consistency, Isolation, Durability) competent.
- NoSQL database maintenance if difficult.
- No standard query language.

RDBMS vs NoSQL key differences

- NoSQL supports a column-oriented database where RDBMS is a row-oriented database.
- NoSQL works faster for unstructured and unrelated data.
- RDBMS uses a schema that means the structure of the data is known in advance to certify that the data follows to the schema. NoSQL is a schema-less database.
- NoSQL is horizontally scalable where RDBMS is vertically scalable. Processing many records in vertically scalable databases are very expensive.

- NoSQL is faster while fetching a greater number of records.
- Mostly banking sector, retail, payroll uses RDBMS as easy to query the tables. Though it has its own limits usually Engineers easily understand it and data validation becomes simple.

Future of Database with NoSQL

Nowadays every organization deals with a massive amount of records from a variety of sources at revolutionary speeds. Relational databases are sometimes ineffective for businesses processing and investigating the vast amount of multifaceted and unstructured data. As NoSQL is schema-less or fixed schema model databases, it is very efficient to handle a large amount of data and it is set to real-time data accessibility data model. Mostly all organizations are swamped with loads of data every second from a variety of sources retrieving from the internet. With this data, validation can be done for making the best or most effective use it and for making future predictions. Using NoSQL real-time analytics are much faster, response times can be under a minute for all complex queries. In SQL databases tables are tied with the primary, foreign keys whereas in NoSQL different data model can be used to deal with the gigantic amount of data. When a user wants to use key-value pairs, the key-value databases can be used, for data pointers graph databases can be used, more nodes can be added to the cluster which is easily scalable instead of using big machines. NoSQL can be used by many advanced applications. NoSQL makes machine learning must faster. Thousands of transactions every second can easily be monitored using NoSQL so it's useful when working with fraud detections of banking transactions. Also, tentatively 2.5 quintillion bytes of data that generate from social media, climate data, innovative pictures and footages, the conversation of data, and that's just the starting. For these scenarios, numerous kinds of elements e.g. Pictures, Video, and Audio are integrated and stored in the database. Various NoSQL databases are scalable to handle the information.

With the growth of Big Data, the operation of NoSQL invention is growing faster among all web organizations and creativities. Assistance includes elastic design, scaling and well switch over convenience. NoSQL databases are serving well web organizations to influence their analytic goals in the fast-paced world. Activists of NoSQL preparations are fast that it delivers faultless handling and boosted implementation comparative to traditional relational databases. NoSQL address the problems of overwhelmingly focused organizations viewing to regulate to varying client requirements and varied growing markets. This technology beats outlooks at processing major unstructured data and the organization joins most popular open source products like Cassandra, MongoDB, Redis, etc. NoSQL also provides a less-expensive substitute for data load and retrieval. If we consider the pros and cons for both NoSQL and SQL, the best approach will be to combine both for additional impulsion research horizons and make it more productive in the future.

Conclusion

Technology is moving faster, and real-time analytics will help organizations to keep up to data. NoSQL permits reliable deployment, distributed database swiftly that can gage with organization's requirements. The article explains various types of NoSQL, its pros and cons, the uses of NoSQL databases over RDBMS.

References

1. Fig:1 Retrieve from https://www.kdnuggets.com/2016/07/seven-steps understanding-nosql-databases.html
2. Cassandra Vs RDBMS, retrieved from https://www.javatpoint.com/rdbms-vs-cassandra
3. Retrieve from https://www.mongodb.com/what-is-mongodb
4. Mukherje S. Popular SQL Server Database Encryption Choices. ArXiv preprint, 2019. ArXiv, 1901, 03179.
5. Mukherjee S. Benefits of AWS in Modern Cloud. ArXiv preprint arXiv, 1903, 03219.
6. Mukherjee S. How IT allows E-Participation in Policy-Making Process. ArXiv preprint arXiv: 2019.00831.
7. Chakraborty, Moonmoon & Excellence, Operations. Supply Chain & Inventory Management. 10.6084/m9.figshare.7824107, 2019.
8. Yoon, Byoung-Ha; Kim, Seon-Kyu; Kim, Seon-Young. Use of Graph Database for the Integration of Heterogeneous Biological Data". Genomics & Informatics. doi:10.5808/GI.2017.15.1.19. ISSN 1598-866X. PMC 5389944. PMID 28416946. 2017; 15(1):19-27.
9. Author Craig Kerstiens, retrieve from Postgres and superuser access *https://www.citusdata.com/blog/*, 2019.