Comparison of NoSQL Databases

***Abstract* - As the need to handle the large amount of data is increasing, the number of NoSQL databases are increasing too. There are four basic types of NoSQL databases. Key-Value stores are best for persistent sharing and some of famous databases are Apache Cassandra, CouchDB, Redis. Graph databases are useful for relational data and famous databases are Neo4j and Allegro graph. Likewise, social and mobile applications can use document based data store like couchDB.**

I. INTRODUCTION

In the past relational databases were considered the only solution to data handling. NoSQL was first introduced as a movement and not technology by group of developers and companies. The purpose of this movement was awareness that there is another simpler approach to look into the data. The use of NoSQL databases is increasing in field of big data and real-time applications that are generation huge amounts of data and need higher throughput. As the data is now generating in very higher scale than ever these databases are making their way into the technology sector. These databases have much simpler design, horizontal scalable and highly available. There are several flavors of NoSQL databases are available in the market. One can choose the database according to the need of business. For example: key- value store is useful in micro services application, graph databases are useful where the data has deep relation and wide column store like cassandra can be used where data is coming at higher rates

II. BODY OF THE PAPER

The term NoSQL was first used in 1998 for a relational database which was not using SQL. And then second time this term camps up in 2009 in conference related to non relational databases. A Rackspace employee Eric Evans who who used this term for the conference says that whole point of seeking an alternate is to resolve the issues that were not addressed by relation databases.

An article written in computer world magazine describes the main issues occurred in relational databases that were addressed by non relational systems, some are discussed below:

Less Complex and higher throughout: Complexity and the level of strictness in implementing ACID in relational databases are somewhat unnecessary in some of the applications.

Also there is a significant higher rate of throughout in non relational databases.

Commodity hardware and Horizontal Scaling: NoSQL databases are designed to run on commodity hardware. In contrast to the sharding in relational databases cluster solution, NoSQL are highly scalable. Horizontal scaling allows to to add or remove hardware without any performance issues.

Object-Relational Mapping: Data structures that have low complexity do not need expensive object relational mapping which is done in relational databases. NoSQL databases mostly stores simple structures are compared to relational databases.

Cheap Clustering: NoSQL databases are designed to use commodity hardware which allow to expand the cluster in cheaply and cost effective manner. Also this helps in applications where the future size of data cannot be guessed.

Cloud Computing: Cloud computing boosted the use of NoSQL databases. Two main requirements of data stores in cloud are high until ultimate scalability and low administration overhead.

NoSQL databases can be divided into four categories. Each of them are
discussed below:

A. Key Value Store:

There is no defined schema in key value store. Unlike relational databases that specify their schema with all the data types and columns, key value store simply considers the data as one collection with a key which is a random string that could be used to access that piece of data.

Each value in key value store corresponds to a Key. As the structure is simple it increases the speed of query as compare to relational database. It allows query and modify operations using the primary key. Key values can be serialized using either java serialization or XML.

Originally key value stores were not designed to support filtering, even they do not have a query language. Simple commands are used to perform basic operations like save or delete or retrieve the data.

Searching is performed by manual indexing in key value store. This approach is not so popular because it needs the user to have good knowledge of indexes.

Scaling is considered as one of the biggest challenges in data stores. There are three most common techniques for scaling in key value store master slave, sharding and dynamo model. In master slave architecture all the write

operations are done to one master machine. Afterwards data is transmitted to stores on other machines in the cluster called slaves. Master could be a bottle neck if the data coming into the system exceeds, as the number of masters are limited. There comes a second approach called sharing. Completely isolated structures are put into one machine, this approach allows to handle the double workload. Adding new server to the cluster could be problem in such system, as it requires to shuffle the data every time a machine is added.
This simplicity makes the key value store most scalable of the NoSQL database types, which is capable of storing the huge amounts of data.

Three of the most common key value databases are discussed below:

Apache Cassandra:

Apache cassandra was developed by Facebook which was later released as open source in 2008. It is now managed by apache foundation. Cassandra is document based distributed NoSQL database. Apache cassandra is famous for its high performance, scalability, and high availability. Query language used by Apache Cassandra is Cassandra Query Language (CQL) which is similar to SQL.
Every node in Cassandra can perform read and write operations and data is replicated across the cluster which make Cassandra fault tolerance and highly available. When a node gets down, user is routed to the nearest available node. Also Cassandra allows linear scaling and high performance with master less architecture.

Redis:

Redis is another famous in-memory key value store. There are two main components in redis architecture redis client and redis server. Server stores the data in memory and perform the management operations. As compared to Cassandra, Redis is faster in fetching the live data as it performs in memory operations and it supports large number of data types. Redis supports master replica replication where server can replicate to any number of replicas.
Typical use case of redis is caching due to its in-memory operations. Other uses could be message queuing engine and leaderboards. Redis is available on amazon web services, and Microsoft azure.

B. Big Table:

Big table is a multidimensional sorted map. Row key, column key and timestamps are used to do indexing. Each value in the big table is array in uninterrupted array of bytes. According to the definition in google white paper it is a distributed storage that is designed to manage large scale structured data
The uninterrupted byte array that stores data can have a wide variety of structured and unstructured data. Memory or disk based storage can be selected using parameters for the coming data. A string represents a key for row. These keys allow data to be stored in order like alphabetically, which enhances the performance overall. This feature is very useful for web based applications as it keeps the data closer to each other which are from same domain.
Column family which is a group of column keys usually have the same data type. Web table implementation using column families to stores web references makes distributed environment more efficient in terms of read and write.
Multiple versions of data are maintained using time stamp indexing. Time stamps are stored in descending order to keep most recent data on top.
Several other big table NoSQL databases were developed based on Google's big table.

Apache Accumulo:

Apache Accumulo written in java uses three other Apache technologies underneath. It uses Apache Hadoop for distributed processing of large data sets and thrift for API functionality, and Zookeeper to serve as the role of traffic cop for configuration and synchronization. To add a layer of security to the data stored Accumulo added the concept of column visibility which is not in the Google's Big table. This feature allows the users to access the only information they are authorized for. Accumulo uses hadoop file system in its cluster and can be scaled horizontally as more data added to the database.

Apache Hbase:

Apache Hbase is the open source version of Big Table. Hbase has most of the functionalities that are in Big Table. Apache Hbase is written in Java. Hbase is an Apache project which aims to provide a storage system similar to Big Table in the Hadoop environment. Hadoop Distributed File System (HDFS) is a distributed file system structure for operating on commodity hardware. Hbase has strict consistency on reads and writes and offer linear scalability. Using Map Reduce Hbase offers parallel processing. Hbase is not eventual consistent but strong consistent in reads and writes. It also supports java API programmatic access and thrift and REST for non java programs.

C. Document Database:

Document databases are built to perform better in terms of storage and query processing. These databases are less concerned about fast reads and writes. It stores data in json like format called document. Json is commonly used

on internet to exchange data. This format is human readable which gives the ability to find meaning in it. No predefined schema is required and data fields can vary in each document. As the schema is flexible, documents can have different values and different attributes. Documents are like objects in programming languages. These documents are independent, so distribution of data across the servers is simple, also replication helps recovery faster and increases availability too. Ad hoc queries and indexing helps to perform analysis and transform data easily. Data from the documents can also be accessed using its content.

Document databases are horizontal scalable which meaning capacity and performance can be increased by adding more servers into the system. Horizontal scaling is very common in the cloud architectures; it decreases the chance of system failure by offering redundancy in data. These databases are fast and are useful for analysis, blogging, management system and e commerce tools. Document databases are not very useful in applications where complex queries or multiple transactions are required.

MongoDB:

Mongo DB is NoSQL database which has features most like the relational database and supports complex data types which uses BJSON data structures to store complex data. Powerful query language of mongoDB allows most of functions like in single-table of relational databases, and it also support indexing. All ACID transactions can be performed and in also supports joins in queries. It also offers reference and embedded relationships. MondoDB speed is 10X faster than relational databases with large datasets. MongoDB is becoming better option than relational databases in most of the projects because of these characteristics. MongoDB is also horizontal scalable using sharding.

Couch DB:

Documents which are primary unit of data in CouchDB consists of any number of fields and attachments which are uniquely named in the database, and CouchDB provides restful and http API for performing add, edit, delete operations on database documents. Documents also have metadata which is maintained by the system. Document fields are uniquely named and contain values of different data types and there is no set limit to text size or element count. The CouchDB features gives all ACID properties. Database file is always in a consistent state because CouchDB never overwrites committed data on disk. Documents are indexed in B-tree structure by their ID

D. Graph Database:

Graph databases use graph structure. They have nodes and edges. Nodes represent entities whereas edges represent relationship between these entities. Graph databases express the relationships in data in simpler and efficient way than relational databases.

Graph database can represent all of the relations in large data set and is helpful in managing deeply linked data. Now graph databases are widely used in number of fields, including computer science and social network analyses. Researchers in the biological community begun to adopt the graph database for biological network analyses.

Graph databases are useful for graph like queries and is flexible enough to allow inserting new data in existing graphs. They have fast retrieval rates because index-free adjacency forces nodes to have physical RAM addresses. There are several types a=of graph databases. Social graph is related to connection between different people. Consumption graph is widely used in retail industry and e-commerce platform. Interest graph is like a social graph; it maps person's interests. Mobile graph deals with mobile data.

For associative data sets graph databases are faster than relational databases. Graph databases do not need join so they can scale more easily for large datasets. Also due to flexible schema approach graph databases are more suitable for the changing data.

Neo4j:

Neo4j is a high performance, NoSQL graph database with all the features of a mature and robust database. It is a graph database developed in Java and it supports master-slave replication. It can be accessed using command line tool, Neo4j browser, Neo4j Desktop and Neo4j Bloom. It can also be accessed through REST interface. It uses cypher query language to perform queries and also supports SPARQL and RDF query languages.

It can be used to represent connected and semi structured data easily. Neo4j has simple and powerful data model and it is highly available for large enterprise applications. Neo4j can easily retrieve relationship details without complex joins.

Allegro graph:

Allegro graph is a database with application framework for building semantic web applications. SPARQL and Prolog query languages can be used to query the triples. Triple consists of subject, predicate and an object. Each triple has a unique identifier. Allegro Graph can be accessed through Allegro graph web view browser or Gruff. Allegro Gruff is tool used to navigate and create new triple or query it. Gruff has two forms, first is standalone and the other one is server edition which is needed to work on hundred of million or billions of triples. Data can be inserted using SPARQL commands on Allegro Graph web view interface. It also provides interface for java, python, ruby etc. It is most commonly used in bioinformatics, healthcare, defense and intelligence departments, telecom etc. It

allows ACID transactions, backup, replication, clustering etc.

General comparison of NoSQL Database types:

	Performance	Scalability	Complexity
Key value store	High	High	None
Document store	High	Variable	Low
Graph database	Variable	Variable	High
Big table	High	High	Moderate

III. CONCLUSION AND FUTURE DIRECTIONS

Moving from one data stack to other is maybe the one of hardest decisions a company has to made, especially when they have invested a good amount of money in schemas and expensive servers. As NoSQL databases are proving to be more efficient than the legacy model but there is always a room for improvements. For example, mongoDB user need to code around slower queries when index is not present. Similarly, some other key value stores require programmer to write index and join login which is built-in in sql. Famous NoSQL databases have made it into the top 10 databases but relational databases are still on top which shows that still there is a long way to go.

REFERENCES

NoSQL Databases: https://christof-strauch.de/nosqldbs.pdf

Comparative Study of the New Generation, Agile, Scalable, High Performance NOSQL Databases : https://www.researchgate.net/publication/235719671_Article_Comparative_Study_of_the_New_Generation_Agile_Scalable_High_Performance_NOSQL_Databases

Understanding Key-Value Databases By Keith D. Foote : https://www.dataversity.net/understanding-key-value-databases/

The NoSQL Movement – Big Table Databases By Shannon Kempe

https://www.dataversity.net/the-nosql-movement-big-table-databases/

Fundamentals of Document Databases **By Keith D. Foote**

https://www.dataversity.net/fundamentals-of-document-databases/