COMPARATIVE STUDY OF DISTRIBUTED, SCALABLE, & HIGH PERFORMANCE NOSQL DATABASES

#1.Ms. Neetu Sharma, Student of M.Tech (CSE), AITM Engg College, Palwal
#2.Ms. Charu Jain, Student of M.Tech (CSE), AITM Engg College, Palwal
#3. Mr. Mahesh Chauhan, Asstt. Prof., AITM Engg College, Palwal

Abstract: SQL and NoSQL have been great inventions over time in order to keep data storage and retrieval optimized and smooth. Relational database is widely used in most of the application to store and retrieve data. They work best when they handle a limited set of data. Handling real time huge volume of data like internet was inefficient in relation database systems. To overcome this problem the "NO-SQL" or "Not Only SQL" Database came into existence [1]. As the popularity of data virtualization continues to rise, companies are increasingly relying on data storage and retrieval mechanisms like NoSQL to extract tangible value out of the voluminous amounts of data available today. In this paper we aim to independently investigate the performance of some NoSQL and SQL databases.
This paper discusses about problems with relation databases, differences between SQL and NOSQL, and how different types of NOSQL Databases are used to efficiently handle the huge amount of data.

Keyword: ***BigData, MongoDB, CouchDB, Key-value pair, column oriented, and document based etc.***

I Introduction: The necessitate for data management has increased over the last few years. Emerging technologies like Cloud Computing & Big Data have driven the acceptance of NoSQL technology. Relational database management system often use a schema-based approach which turns out to be a mismatch when it comes to processing unstructured formats of data types. NoSQL comes in handy as it can use varying data types with ease. Advantage of NoSQL, works well with distributed data stores such as Google and Facebook where large data types are handled and need to be stored. Such kind of huge data types may not require having join operations, fixed schema and horizontal scaling.

Examples of SQL database are as: MySql, Oracle, Sqlite, Postgres and MS-SQL[10].

Examples of NoSQL database are as: MongoDB, BigTable, Redis, RavenDb, Cassandra, Hbase, Neo4j and CouchDb[10].

1. MySQL Community Edition

MySQL database is very popular open-source database. It is generally been stacked with apache and PHP[6]. The following are some of MySQL benefits and strengths:

(i) Replication: The work load can be reduced by heavily increasing the scalability and accessibility of business application.

(ii)Sharding: By sharding MySQL servers, the application is partitioned into multiple servers and dividing the database into small chunks[12]. As low cost servers can be deployed for this purpose, this is cost effective. It is useful when there is large no of write operations in a high traffic website.

Memcached as a NoSQL API to MySQL is used to increase the performance of the data retrieval operations giving an advantage of NoSQL API to MySQL server.

2. MS-SQL Server Express Edition

It is a powerful and user friendly database which has good stability, reliability and scalability with support from Microsoft. The following are some of MS-SQL benefits and strengths:

- Integrated Development Environment
- Disaster Recovery
- Cloud back-up

3. Oracle Express Edition

This database is a limited edition of Oracle Enterprise Edition server with certain limitations and it is also free for development and deployment [6]. The following are some of Oracle benefits and strengths:

- Easy to Upgrade
- Wide platform support
- Scalability

NoSQL Database Examples

1. MongoDB

Mongodb has been developed by the founders of DoubleClick, written in C++ and is currently being used by some big companies like The New York Times, Craigslist, MTV Networks. This database is one of the most popular document based NoSQL database as it stores data in JSON like documents. It is non-relational database with dynamic schema[5,10]. The following are some of MongoDB benefits and strengths[11]:

(i)Speed: It gives good performance, as all the related data are in single document which eliminates the join operations.

(ii)Scalability: It reduces the workload by increasing the number of servers in your resource pool instead of relying on a individual resource.

(iii)Manageable: It is easy to use for both developers and administrators.

(iv)Dynamic Schema: Its gives the flexibility to evolve data schema without modifying the existing data.

2. CouchDB

It is also a document based NoSQL database. It stores data in form of JSON documents. The following are some of CouchDB benefits and strengths:

(i)Schema-less: As a member of NoSQL family, it also have dynamic schema which makes it more flexible, having a form of JSON documents for storing data.

(ii)HTTP query: Can access database documents using your web browser.

(iii)Conflict Resolution: It has automatic conflict detection which is useful while in a distributed database.

(iv)Easy Replication: Implementing replication is fairly straight forward

3. Redis

This database is another Open Source NoSQL database which is mainly used because of its lightening speed. It is written in ANSI C language. The following are some of benefits and strengths:

(i)Data structure: The keys stored in database can be hashes, lists, strings, sorted or unsorted sets.

(ii)Redis as Cache: Using Redis as a cache by implementing keys with limited time to live to improve the performance.

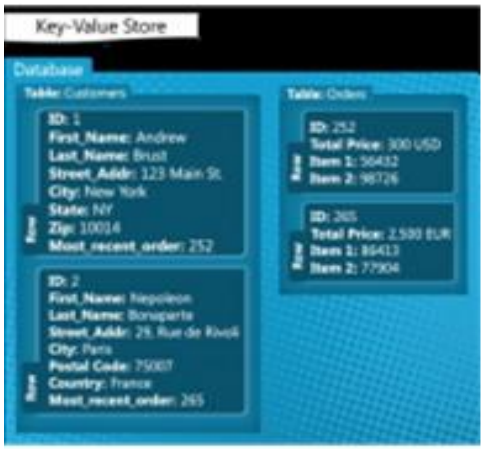
(iii)Very fast: It is considered as one of the fastest NoSQL server as it works with the in-memory dataset.

II Types of NOSQL Databases:

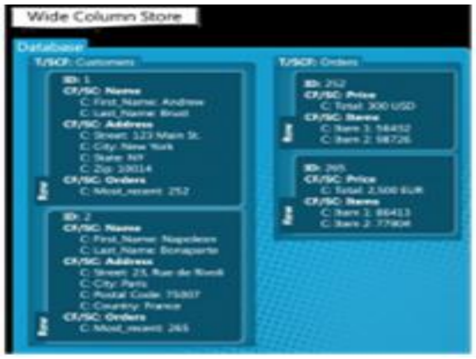
There are three types of popular NoSQL databases. Each database is individually good at dealing with size and complexities.

1. Key-value stores: As the name implies, it is a system that stores values indexed for retrieval by keys. These systems can hold structured or unstructured data. Key value data model means that a value corresponds to a Key. In this structure is simple, the query speed is higher than relational database, it supports mass storage and high concurrency, etc., It provided support for query and modify operations for data through the primary key [3].

Key values represent bucket of data. For example, in case of a shopping cart mentioned in Figure 1, each shopping cart are represented in individual buckets and represented using a key value which could be user id. The key values can be serialized using either java serialization or XML. This way is very fast to store as it just writes bits to the discs. Some of key value stores available in market are Berkeley DB, Tokyo Tyrant, Voldemart, Crassandra.

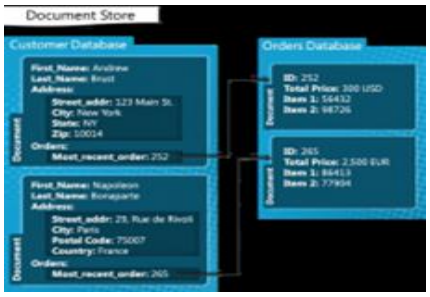


2. Column- oriented databases: Rather than store sets of information in a heavily structured table of rows and columns with uniform sized fields for each record, as is the case with relational databases, column-oriented databases contain one extendable column of closely related data.



3. Document-based stores: This database store and organize data as collections of documents, rather than as structured tables with uniform sized fields for each record. With these databases, users can add any number of fields of any length to a document. Document database is not concerned about high performance read and write concurrent, but rather to ensure that big data storage and good query performance [3].

Typical document database are Mongo DB, Couch DB. Couch DB is one most popular document database which is flexible, fault-tolerant database, which supports data format called JSON. It provides REST-style API to ensure data consistency. In addition, Mongo DB is non-relational database, which features the richest and most like the relational database. It supports complex data types which uses BJSON data structures to store complex data type [3].



Example: MongoDB was designed to store a huge data (BigData), to be easy-scalable, high-available and fast. JOINs and global transactions were a stumbling-block for designing such system and the only way to get rid of them is to use another data storage model. It means to avoid data splitting between tables during normalization process. Key-value pairs combined into the document were chosen as data storage model. The data are stored mainly in one collection (set of documents) without necessity to JOIN and watch over data consistency. MongoDB is called document-oriented database.

4. Graph Database: Graph Database works with 3 central part like abstraction, Node, relationships between nodes and key value pairs which can be attached to nodes and relationships.Neo is a graph database which is supported by neo technologies, it is a high-performance, NOSQL graph database with all the features of a mature and robust database [7].

III Key Point For SQL and NOSQL:

(i)SQL databases are mainly called as Relational Databases (RDBMS); whereas NoSQL database are mainly called as non-relational or distributed database.

(ii)SQL databases are table based databases whereas NoSQL databases are document based, key-value pairs, graph databases or wide-column stores. This means that SQL databases represent data in form of tables which consists of n number of rows of data whereas NoSQL databases are the collection of key-value pair, documents, graph databases etc.

(iii)SQL databases have predefined schema whereas NoSQL databases have dynamic schema for unstructured data.

(iv)SQL databases are vertically scalable whereas the NoSQL databases are horizontally scalable.

(v) SQL databases uses SQL (structured query language) for defining and manipulating the data, which is very powerful. In NoSQL database, queries are focused on collection of documents. Sometimes it is also called as UnQL (Unstructured Query Language).

(vi)For complex queries: SQL databases are good fit for the complex query intensive environment whereas NoSQL databases are not good fit for complex queries

(vii)For the type of data to be stored: SQL databases are not best fit for hierarchical data storage. But, NoSQL database fits better for the hierarchical data storage as it follows the key-value pair way of storing. NoSQL database are highly preferred for large data set (i.e for big data).

(viii)For high transactional based application: SQL databases are best fit for heavy duty transactional type applications, as it is more stable and promises the atomicity as well as integrity of the data. While NoSQL uses for transactions purpose, it is still not comparable and stable enough in high load and for complex transactional applications.

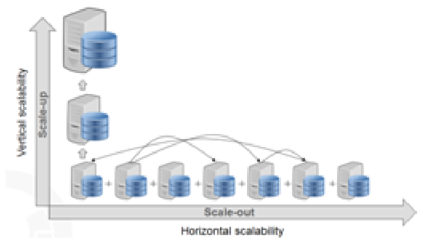
(ix)For properties: SQL databases emphasizes on ACID properties (Atomicity, Consistency, Isolation and Durability) whereas the NoSQL database follows the Brewers CAP theorem (Consistency, Availability and Partition tolerance)

(x)For DB types: On a high-level, we can classify SQL databases as either open-source or close-sourced from commercial vendors. NoSQL databases can be classified on the basis of way of storing data as graph databases, key-value store databases, document store databases, column store database and XML databases [9].

(xi)Agile practices: Relational databases were not designed to support Agile software development and be easy scalable, but NoSQL databases on the other hand support them. The modern world uses Agile practices to develop products, agile sprints with quick iteration is the one of the main goals. Dynamic schema is the key of Agile support in MongoDB.

(xii)Database Schema: Relational databases (SQL) require established data schemas for data storage, before data will be added. Schema necessity doesn't fit with agile development approaches. It's hard to predict an exact db schema at the beginning of the feature development. If schema is changed in relational database you must think over data migration to the new schema. If data is large it's a very slow process that involves significant downtime. If data is changed regularly the downtime may also be frequent. NoSQL database in contrast to relational databases doesn't require a predefined schema and what is more it doesn't require schema at all. We can call it schema-less database. This feature is fully fit into the Agile approaches. The same collection can contain data with a significantly differing structure.

(xiv)Scalability: Relational database can be scaled just vertically, because entire database has to be hosted in a single server. This is necessary in order to ensure reliability and continuous availability of data. NoSQL databases were designed to scale horizontally. Instead of increasing power of one single server we just need to add more server instances to get expected power.



(xv)Query Language: Relational databases use SQL as query language to retrieve data. The scope of SQL includes data insert, query, update, delete, schema creation and modification and data access control.

IV Key Characteristics of NoSQL Databases:

1. Distributed Computing (Scalability, Reliability, Sharing of Resources, Performance) **-** NoSQL databases are distributed, can scale horizontally. These databases handles large amount of data like in terabytes or petabytes, with low latency.

The increase in parallel users and the volume of data requires web and mobile applications and supporting databases to scale accordingly using, which can be achieved using by following two methods:

a. Vertical Scaling (or scale up) - This implies adding resources to a single node by deploying additional CPU or increasing the memory storage [2].

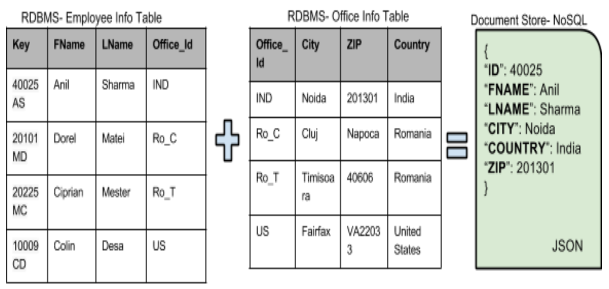
Scaling Up with Relational Databases: To support a large number of concurrent users and/or store more data, we need big servers with additional CPUs to handle the workload, more memory, and more disk storage to keep all the tables. Installing and maintaining big servers is a complex process that also adds to capital expenditure, unlike the low-cost, commodity hardware typically used at the web/application server tier of RDBMS.

b. Horizontal Scaling (or scale out) - Scale out is related to adding on to the nodes on a system. E.g. – Adding a new computer to a distributed software application.

Scale Out with NoSQL Database: Scaling out takes recourse to a cluster of servers for storing data and supporting database operations. They use a cluster of servers to store data and support database operations. Since commodity servers are susceptible to failure, NoSQL databases are designed to withstand and recover from such recurring failures, making them highly flexible.

NoSQL databases provide a much easier, linear approach to database scaling.

2. **More Flexible Data Model -** NoSQL databases follow one of the following data models or data stores[2]:
a. Key Value Stores
b. Document Stores
c. Column Based Stores
d. Graph Databases
e. XML Databases


RDBMS- Employee Info Table

Key	FName	LName	Office_Id
40025 AS	Anil	Sharma	IND
20101 MD	Dorel	Matei	Ro_C
20225 MC	Ciprian	Mester	Ro_T
10009 CD	Colin	Desa	US

RDBMS- Office Info Table

Office_Id	City	ZIP	Country
IND	Noida	201301	India
Ro_C	Cluj	Napoca	Romania
Ro_T	Timisoara	40606	Romania
US	Fairfax	VA22033	United States

Document Store- NoSQL

```
{
"ID": 40025
"FNAME": Anil
"LNAME": Sharma
"CITY": Noida
"COUNTRY": India
"ZIP": 201301
}
```

JSON

3. **Asynchronous Inserts & Updates/Weak Transactional –** Full transactional guarantees and simultaneous completion of transactions at all nodes in the distributed environment are not provided by NoSQL databases. Instead it guarantees the availability of the data at the distributed level (by some internal process of synchronization). This is the reason why NoSQL is a perfect model to apply for instances like social media applications where simultaneous transactions are not a constraint.

4. **Follows BASE/CAP instead of ACID** - Instead of ACID, NoSQL databases more or less follow called "BASE" (Basically Available, SoftState, Eventual Consistency). All NoSQL databases relax one or more of the ACID properties (CAP theorem). For example, when no updates occur for a certain period of time(could be few seconds), eventually all updates may propagate through the system depending on load, cluster size and network traffic which will make all the nodes consistent.

5. **Query Language** - These databases do not support SQL unlike in relational databases. However, few NoSQL databases support some other form of query language, like CouchDB uses JSON to store data and JavaScript as its query language.

6. **No Joins -** NoSQL databases don't use the concept of joins.

7. **Easy Implementation** –It provides schema flexibility and less complicated relationships unlike in RDBMS.

CAP Theorem

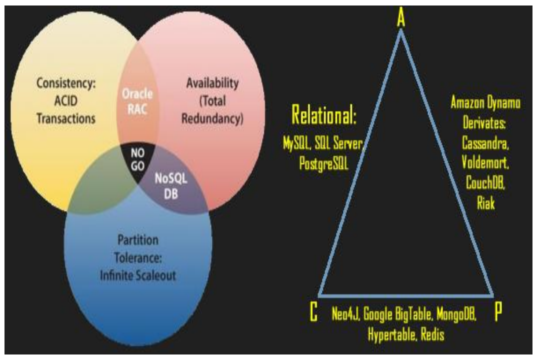
As per the CAP theorem [5]; the following guarantees are not available in a distributed system:

(i)Consistency - All nodes view the same data at the same time. Data in the database remains consistent after the execution of an operation.

(ii)Availability - A guarantee that every request receives a response about whether it was successful or failed.

(iii)Partition Tolerance - The system continues to operate despite failure of part of the system. The servers may be partitioned into multiple groups that cannot communicate with every other group. The network can break into two or more parts, each with active systems that cannot influence other parts.

NoSQL database follows different combinations of the CAP theorem [3]. Here is an detailed description of three such combinations:


Consistency: ACID Transactions
Oracle RAC
Availability (Total Redundancy)
NO GO
NoSQL DB
Partition Tolerance: Infinite Scaleout
A
Relational: MySQL, SQL Server, PostgreSQL
Amazon Dynamo Derivates: Cassandra, Voldemort, CouchDB, Riak
C
Neo4J, Google BigTable, MongoDB, Hypertable, Redis
P

CA - All nodes will remain in contact as a result of the single site cluster and any partition will block the system.

CP – In this, some data might not be accessible but consistency and accuracy are not compromised. There is no need for distributed concurrency control as well.

AP - With the AP approach, the returned data might not be accurate but system will be available in spite of any partitioning. This is best suited for replication needs and fault tolerance.

V Difference Between SQL & NoSQL[7]:

	MS SQL	MongoDB
Data Storage Model	Relational DBMS	Document-oriented
JOINs	Yes	No
Transaction	ACID	No
Support agile practices	No	Yes
Data schema	Fixed	Dynamic
Scalability	Vertical	Horizontal
Replication	Yes (depending on software edition)	Primary-Secondary
MapReduce	No	Yes
Query Language	SQL query language	JSON query language
Secondary Indexes	Yes	Yes
Triggers	Yes	No
Foreign keys	Yes	No
Concurrency	Yes	Yes
Official Website	http://www.microsoft.com/en-us/sqlserver/default.aspx	http://www.mongodb.org
Company	Microsoft	MongoDB, Inc
Licence	Commercial	Open Source
Implementation language	C++	C++
OS support	Windows	Windows, Linux, OS X, Solaris
Drivers for programming languages	.NET, Java, PHP, Python, Ruby, Visual Basic	Actionscript, C, C#, C++, Clojure, ColdFusion, D, Dart, Delphi, Erlang, Go, Groovy, Haskell, Java, JavaScript, Lisp, Lua, MatLab, Perl, PHP, PowerShell, Prolog, Python, R, Ruby, Scala, Smalltalk

VI Advantages

1. **Non-Relational means table-less:** NoSQL databases are non-relational. This means they are easier to manage and they provide a higher level of flexibility with newer data models.

2**. Mostly Open Source**: The open source nature of NoSQL databases means it is now possible to use large data hungry applications at a lower price point. This is because open source technologies are free of cost.

3.**Easier scalability through support for Map Reduce:** NoSQL database experts often use elastic scalability as a major selling point of NoSQL. NoSQL databases are designed to function on full throttle even with low-cost hardware.

4. No need to develop a detailed database model: The non-relational nature of a NoSQL database allows database architects to quickly create a database without needing to develop a detailed (fine-grained) database model. This saves a lot of development time.

VII Disadvantages

1. **Lack of reporting tools:** A major problem with NoSQL databases is the lack of reporting tools for analysis and performance testing. However, with MySQL, we can find a wide array of reporting tools to help for application's validity.

2**. Doesn't conform to ACID properties:** Relational database systems function on the ACID paradigm (Atomicity, Consistency, Isolation, Durability) but NoSQL databases don't.

3.**Lack of standardization: I**n order for NoSQL to grow, it needs a standard query language like SQL. This is a major issue highlighted by researchers at Microsoft, who claim that NoSQL's lack of standardization can cause a problem during migration.

VIII Conclusion: With the current emphasis on "Big Data", NoSQL databases have surged in popularity. These databases are claimed to perform better than SQL databases[4]. SQL and NoSQL have been great inventions over time in order to keep data storage and retrieval optimized and smooth. Criticizing any one of them will not help the cause. If there is a buzz of NoSQL these days, it doesn't mean it is a silver bullet to all our needs. Both technologies are best in what they do. It is up to a developer to make a better use of them depending on the situations and needs. NoSQL databases are becoming a major part of the database landscape today, and with their handful of advantages, they can be a real game changer in the enterprise arena. If our application doesn't fall into the category of the likes of Google, Yahoo, Facebook or

Wikipedia, we should reconsider our options for using NoSQL and stick with MySQL instead.

IX Reference:

[1] Clarence J M Tauro et.al."Comparative Study of the New Generation, Agile, Scalable, High Performance NOSQL Databases", IJCA June 2012

[2] http://www.3pillarglobal.com/insights/just-say-yes-to-nosql

[3] Haihong, E. ; Guan Le ; Jian Du"Survey on NoSQL database", IEEE Conference, Oct 2011.

[4] Yishan Li , Manoharan, S. "A performance comparison of SQL and NoSQL databases ", IEEE conference, Aug 2013.

[5] Radulescu, F. ; Agapin, L.I. et.al. "MongoDB vs Oracle -- Database Comparison " IEEE Conference, Sept 2012

[6]http://www.thegeekstuff.com/2014/01/sql-vs-nosql-db/

[7]https://www.digitalocean.com/community/tutorials/understanding-sql-and-nosql-databases-and-different-database-models

[8]http://www.networkworld.com/article/2226514/tech-debateshat-s-better-for-your-big-data-applica/tech-debates/what-s-better-for-your-big-data-application--sql-or-nosql-.html

[9]https://www.udemy.com/blog/nosql-vs-sql/

[10]http://blog.monitor.us/2013/05/cc-in-review-the-key-differences-between-mysql-and-nosql-dbs/

[11] http://sql-vs-nosql.blogspot.in/

[12] http://techledger.wordpress.com/2011/07/15/problems-of-rdbms/