The Rising NoSql Technology

Sumati Baral

Assistant Professor, Department of CSE,Trident Academy Of Creative Technology, BBSR,Odisha,India

Abstract*—The rising interest in NoSQL technology over the last few years resulted in an increasing number of evaluations and comparisons among competing NoSQL technologies From survey we create a concise and up-to-date comparison of NoSQL engines, identifying their most beneficial use from the software engineer point of view.*

Keywords—Agile, ACID, BASE, CAP Theorem , Schema*.*

I. INTRODUCTION

Databases can be divided in 3 types:

- RDBMS (Relational Database Management System)
- OLAP (Online Analytical Processing)
- NoSQL (recently developed database)

NoSQL stands for **not-only-SQL**. The idea here is not to oppose SQL, but instead provide an alternative in terms of storage of data. As most users are well versed with SQL, many NoSQL databases strive to provide an SQL like query interface.

What is NoSQL?

It is designed for distributed data stores where very large scale of data storing needs (for example Google or Facebook which collects terabits of data every day for their users). These type of data storing may not require fixed schema, avoid join operations and typically scale horizontally.

Characterstics "Not only SQL"

- Does not use SQL as querying language
- Distributed, fault-tolerant architecture
- No fixed schema (formally described structure)
- No joins (typical in databases operated with SQL)
- RDBMS are "scaled up" by adding hardware processing power
- NoSQL is "scaled out" by spreading the load
  - Partitioning (sharding) / replication

Why NoSQL?

Well, here are the reasons:

- **Managing Large Chunks of Data:** NoSQL databases can easily handle numerous read/write cycles, several users and amounts of data ranging in petabytes.
- **Schema not needed:** Most NoSQL databases are devoid of schema and therefore very flexible. They provide great choices when it comes to constructing a schema and foster easy mapping of objects into them. Terms such as normalization and complex joins are, well, not needed!
- **Programmer-friendly:** NoSQL databases provide simple APIs in every major programming language and therefore there is no need for complex ORM frameworks. And just incase APIs are not available for a particular programming language, data can still be accessed over HTTP via a simple API, using XML and/or JSON.
- **Availability**: Most distributed NoSQL databases provide easy replication of data and failure of one node does not affect the availability of data in a major way.
- **Scalability**: NoSQL databases do not require a dedicated high performance server. Actually, they can easily be run on a cluster of commodity hardware and scaling out is just as simple as adding a new node.
- **Low Latency:** Unless you are running a cluster of a trillion data servers (or something like that, give or take a few million of them), NoSQL can help you achieve extremely low latency. Of course, latency in itself depends on the amount of data that can be successfully loaded into memory.

RDBMS vs NoSQL

RDBMS

- Structured and organized data
- Structured query language (SQL)
- Data and its relationships are stored in separate tables.
- Data Manipulation Language, Data Definition Language
- Tight Consistency

NoSQL

- Stands for Not Only SQL
- No declarative query language
- No predefined schema
- Key-Value pair storage, Column Store, Document Store, Graph databases
- Eventual consistency rather ACID property
- Unstructured and unpredictable data
- CAP Theorem
- Prioritizes high performance, high availability and scalability
- BASE Transaction

BENEFITS OF NOSQL OVER RDBMS

Schema Less:

NoSQL databases being schema-less do not define any strict data structure.

Dynamic and Agile:

NoSQL databases have good tendency to grow dynamically with changing requirements. It can handle structured, semi-structured and unstructured data.

Scales Horizontally:

In contrast to SQL databases which scale vertically, NoSQL scales horizontally by adding more servers and using concepts of sharding and replication. This behavior of NoSQL fits with the cloud computing services such as Amazon Web Services (AWS) which allows you to handle virtual servers which can be expanded horizontally on demand.

Better Performance

All the NoSQL databases claim to deliver better and faster performance as compared to traditional RDBMS implementations.

Talking about the limitations, since NoSQL is an entire set of databases (and not a single database), the limitations differ from database to database. Some of these databases do not support ACID transactions while some of them might be lacking in reliability. But each one of them has their own strengths due to which they are well suited for specific requirements.

II. NoSQL DATA MODEL

Some of the major and most prominent differentiations among NoSQL databases are as follows:

1. Document Stores
2. Hierarchical
3. Network
4. Column-oriented
5. Object-oriented
6. Key-value Stores
7. Triple Stores

Document stores

Gone are the days when data organization used to be as minimal as simple rows and columns.The reason for favoring XML or JSON is because both of them are extremely portable, c ompact and standardized. Again, simply because NoSQL databases are schema-less, and

there exists no predefined for an XML or JSON document and as a result, each document is independent of the other. The database can be employed in CRM, web-related data, real-time data, etc. Some of the most well known implementation models are MongoDB, CouchDB and RavenDB.

Hierarchical Databases

These databases store data in the form of hierarchical relevance, that is, tree or parent-child relationship. In terms of relational models, this can be termed as 1:N relationship. Basically, geospatial databases can be used in a hierarchical form to store location information which is essentially hierarchical, though algorithms may vary. Major examples of the same include PostGIS, Oracle Spatial, etc. Also, some of the most well known implementations of hierarchical databases are the Windows Registry by Microsoft and the IMS Database by IBM.

Graph Network Databases

Graph databases are the most popular form of network database that are used to store data that can be represented in the form of a Graph. Basically, data stored by graph databases can grow exponentially and thus, graph databases are ideal for storing data that changes frequently. A general technique to query a graph is to begin from an arbitrary or specified start node and follow it by traversing the graph in a depth-first or breadth-first fashion, as per the relationships that obey the given criterion. Most graph databases allow the developer to use simple APIs for accomplishing the task. For instance, you can make queries such as: “Does Jonny Nitro read Data Center Magazine?” Some of the most popular graph databases include FlockDB, HyperGraphDB and Neo4j.

Column-oriented Databases

Column-oriented databases came into existence after Google’s research paper on its BigTable distributed storage system, which is used internally along with the Google file system. Some of the popular implementations are Hadoop Hbase, Apache Cassandra, HyperTable, etc.

Such databases are implemented more like three-dimensional arrays, the first dimension being the row identifier, the second being a combination of column family plus column identifier and the third being the timestamp. Column-oriented databases are employed by Facebook, Reddit, Digg, etc.

Object-oriented Databases

Such databases allow the storage of data in the form of objects, thereby making it highly transparent. Some of the most popular ones include db4o, NEO, Versant, etc. Object-oriented databases are generally used in research purposes or web-scale production.

Key-value stores

Key-value stores are based on Amazon’s Dynamo Research Paper and Distributed hash Tables. Such data models

are extremely simplified and generally contain only one s et of global key value pairs with each value having a unique key associated to it. The database, therefore, is highly scalable and does not store data relationally. Some popular implementations include Project Voldemort (open-sourced by LinkedIn), Redis, Tokyo Cabinet, etc.

Advantages of NoSQL

- **Dynamic and Agile:**
  NoSQL databases have good tendency to grow dynamically with changing requirements. It can handle structured, semi-structured and unstructured data.
- **Scales Horizontally:**
  In contrast to SQL databases which scale vertically, NoSQL scales horizontally by adding more servers and using concepts of sharding and replication. This behavior of NoSQL fits with the cloud computing services such as Amazon Web Services (AWS) which allows you to handle virtual servers which can be expanded horizontally on demand.
- **Better Performance:**
  All the NoSQL databases claim to deliver better and faster performance as compared to traditional RDBMS implementations.
- **NoSQL database examples**

CAP Theorem (Brewer's Theorem)

CAP theorem states that there are three basic requirements which exist in a special relation when designing applications for a distributed architecture.

Consistency - This means that the data in the database remains consistent after the execution of an operation. For example after an update operation all clients see the same data.

Availability - This means that the system is always on (service guarantee availability), no downtime.

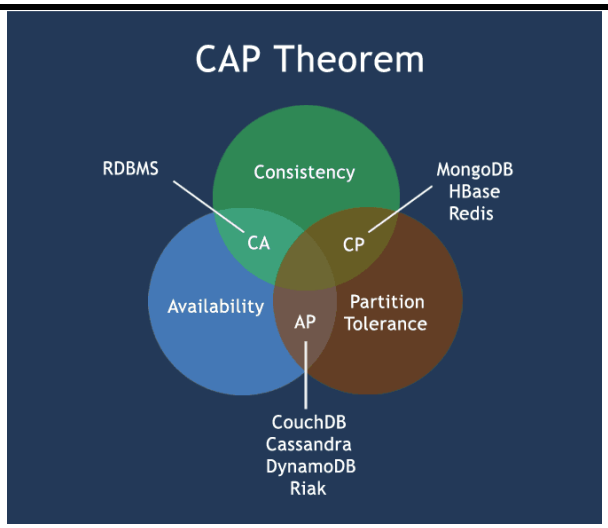
Partition Tolerance - This means that the system continues to function even the communication among the servers is unreliable, i.e. the servers may be partitioned into multiple groups that cannot communicate with one another.

In theoretically it is impossible to fulfill all 3 requirements. CAP provides the basic requirements for a distributed system to follow 2 of the 3 requirements. Therefore all the current NoSQL database follow the different combinations of the C, A, P from the CAP theorem. Here is the brief description of three combinations CA, CP, AP :

CA - Single site cluster, therefore all nodes are always in contact. When a partition occurs, the system blocks.

CP - Some data may not be accessible, but the rest is still consistent/accurate.

AP - System is still available under partitioning, but some of the data returned may be inaccurate.



NoSQL pros/cons

Advantages :

- High scalability
- Distributed Computing
- Lower cost
- Schema flexibility, semi-structure data
- No complicated Relationships

Disadvantages

- No standardization
- Limited query capabilities (so far)
- Eventual consistent is not intuitive to program for

The BASE

The BASE acronym was defined by Eric Brewer, who is also known for formulating the CAP theorem.

The CAP theorem states that a distributed computer system cannot guarantee all of the following three properties at the same time:

- Consistency
- Availability
- Partition tolerance

A BASE system gives up on consistency.

- **B**asically **A**vailable indicates that the system *does* guarantee availability, in terms of the CAP theorem.
- **S**oft state indicates that the state of the system may change over time, even without input. This is because of the eventual consistency model.
- **E**ventual consistency indicates that the system will become consistent over time, given that the system doesn't receive input during that time.

ACID vs BASE

ACID	BASE
Atomic	**B**asically **A**vailable
Consistency	**S**oft state

Isolation	**E**ventual consistency
Durable	

POPULAR NOSQL DATABASES

- **Document Oriented Databases** – MongoDB, HBase, Cassandra, Amazon SimpleDB, Hypertable, etc.
- **Graph Based Databases** – Neo4j, OrientDB, Facebook Open Graph, FlockDB, etc.
- **Column Based Databases** – CouchDB, OrientDB, etc.
- **Key Value Databases** – Membase, Redis, MemcacheDB, etc.

III. CONCLUSION

Application developers have been frustrated with the impedance mismatch between the relational data structures and the in-memory data structures of the application. Using NoSQL databases allows developers to develop without having to convert in-memory structures to relational structures.

REFERENCES

[1] www.ibm.com/analytics/**nosq**
[2] Best-MongoDB-**Books**.html
[3] https://**books**.google.co.in/**books**?isbn=331925264X.