Introducing and Comparing NoSQL Database Management Systems

Jiarong Ye
The Pennsylvania State University
jxy225@psu.edu

ABSTRACT

This paper gives a precise introduction of the NoSQL DBMS and provides comparisons among some of the most widely used NoSQL databases in the industries nowadays, Cassandra, Redis, MongoDB and Neo4j, mainly from their architectures, data models, benefits and drawbacks.

KEYWORDS

NoSQL, Architecture, Data Model

1 BACKGROUND

With the exponentially generated and accumulated data from varied sources like mobile devices, mechanical sensors, financial transactions, satellite imaging, the task of data managing gets much more challenging day after day. And these collected raw data are in a wide range of forms, most are unstructured, like images, videos, audios, etc. In the whole data processing pipeline, starting from data integrating and storing, it's already a quite essential part. Since the consistency, accessibility, velocity of data inserting and retrieving might make a huge difference to the following data manipulation tasks, especially in fields like finance, social networks, etc. Thus making an optimal choice of the right database for different data format, different data volume, different business focus like whether velocity or consistency is more important for the client's perspective.

2 INTRODUCTION OF NOSQL DATABASES

Widely used Database Management Systems are Relational DBMS such as SQL (MySQL, SQL Server), and NoSQL(Not Only SQL). NoSQL is not built for the entire replacement of SQL, but as a complementary option for different tasks. The architecture of NoSQL databases are distributed and fault-tolerant by utilizing replica sets among multiple machine to guarantee data availability in the case of network disruption or when certain servers accidentally go down. Why the relational DBMS is limited is that due to its strong focus on consistency and reliability and ACID guaranteed (ACID: Atomicity, Consistency, Isolation and Durability), thus also making it difficult for scaling, which is one of the many reasons why NoSQL is considered better than SQL. To reach a balance between availability and consistency, NoSQL achieve more optimal level of consistency by slightly sacrificing its availability.

NoSQL also can reduce the level of availability to achieve higher consistency. However there are still many remaining problems which NoSQL databases cannot solve. As a result, providers produced distinct data model CDBMSs(Cloud Database Management System) to satisfy different requirements, whereas each of them are present an imperfection.

By design, NoSQL databases and management systems are relationless (or schema-less). They are not based on a single model (e.g. relational model of RDBMSs) and each database, depending on their target-functionality, adopt a different one.

In a way, RDBMS is still the mainstay in the industry, however, scalability is the major limitation of RDBMS since it is not built to support distributed system. That's when the NoSQL comes into play. Compared with the conventional relational data storage system, one of the main reasons that NoSQL stands out is that it is easy to scale horizontally. With sharding, which would be explored in more details later, it partitions data onto a cluster of remote machines, boosting the overall performance while controlling cost with cheap commodity hardware. And the ability to process massive amount of data with a schema-less structure is also what makes NoSQL more popular in data mining industries where the data that most engineers and data scientists have to work with are unstructured with flexible schema.

3 CHARACTERISTICS OF NOSQL DATABASES

3.1 Availability

3.1.1 Replication.

To guarantee high availability, the mechanism of failover is utilized by creating replica sets as remedy for server malfunctions, network disruptions, system maintenance and upgrade.

3.2 Scalability

3.2.1 Master-slave.

For databases with master-slave mechanism, all writes into the database are written to the master node, and all reads are processed through the replication set of slave nodes. This centralization architecture could encounter significant inconvenience when the incoming dataset is large, the step when the master node duplicates the data and then passes them down to slave nodes will slow down the entire processing speed. Moreover, if the data written in is somehow incorrect, the mistake would be propogated down to all the slave nodes as well, hence providing the wrong version of data to the users for that the master node is the only source that receives the input raw data.

3.2.2 Sharding.

In the partitioning step, carefully designed hashing functions are applied to distribute the data as balanced as possible to multiple workers on a cluster to process in parallel. However the improvement of performance comes with sacrifice as well, that across partitions, operations such as joins since the lack of consistency and flexible schema all violate the prerequisite of executing joins. And complex relation features are not supported and referential integrity constraints can also no longer be maintained.

3.3 Consistency

As a distributed system, since the writes of data into storage is asynchronous, thus with different architecture, the real-time consistency is not guaranteed. But the eventual consistency is absolute after all updates are propagated throughout the system

4 DATA MODELS OF NOSQL DATABASES

Although various NoSQL databases are currently used in industries depending on the different products and services each company offers to their target clients, there are several major NoSQL in each of its four categories that are most widely used: Redis as key-value store database, Cassandra as column-oriented database, MongoDB as document-oriented database and Neo4j as graph database.

4.1 Key-Value Databases

Key-value database storage systems store data in a distributed system

4.1.1 Redis.

As a NoSQL database written in C++, the excel performance and high I/O speed of Redis makes it an optimal choice for caching. Redis supports varied data structures such as list, queue, set (basic operations like union, intersection, difference), hash table, stc.

Redis is preferred for applications with real-time data collecting and processing, which specifically requires high I/O speed because the incoming data is updating rapidly.

4.2 Column-oriented Databases

4.2.1 Cassandra.

Cassandra shines in the applications with large clusters. The replication mechanism guarantees high availability of each data center. And the querying language (CQL) used in Cassandra is quite similar to SQL, hence the learning curve is not as steep as databases like HBase with the entire ecosystem including zookeeper, pig, hive, etc.

4.3 Document-oriented Databases

4.3.1 MongoDB.

Among many NoSQL databases, MongoDB is considered the most popular with the largest number of users. Because not only does it has features of NoSQL database system like flexible schema, easy to scale out and also MongoDB is quite SQL-friendly, for instance, MongoDB allows users to create indexes to speed up the searching process. And the support of writing quries in Javascript and inserting data with a semi-structured format like json, bson, etc. makes MongoDB a user-friendly choice because unlike the other NoSQL databases, it does not require users to learn new syntax specific to the database in order to start using it.

4.4 Graph Databases

4.4.1 Neo4J.

5 BENEFITS AND DRAWBACKS

5.1 Benefits

5.1.1 Scaling Capability.

5.1.2 Process Big Data.

5.1.3 Flexible schema.

5.1.4 Cost.

5.2 Drawbacks

5.2.1 Not Support Complicated transactions.

- Join
- Aggregate

5.2.2 Stability.

6 CONCLUSIONS

REFERENCES

[1] A. B. M. Moniruzzaman and Syed Akhter Hossain. 2013. NoSQL Database: New Era of Databases for Big data Analytics - Classification, Characteristics and Comparison. *CoRR* abs/1307.0191 (2013). arXiv:1307.0191 http://arxiv.org/abs/1307.0191