WHITE PAPER

NOSQL TECHNOLOGY FOR TODAY'S DATA

Abstract

Over the past decade, NoSQL has become a mainstream technology that is used by many enterprises to re-imagine how they use data. NoSQL is a broad area encompassing multiple underlying database technologies, each with its specific use case. This white paper provides an overview of NoSQL and examines how different NoSQL databases can help companies meet rapidly changing business needs to stay competitive.

Infosys®
Navigate your next

The evolution of data – the 3 Vs

Data has come a long way. If one compares the volume of data that could be stored on 80-byte punch cards used in the 1950s to that on 1TB microSD cards that were rolled out in February 2019, the increase in storage density is nearly 12 billion times. Even the variety of data that can be stored has changed from highly structured corporate data to videos, chats and pictures. According to a study by Marr in 2018[1], 1.2 trillion photographs were generated in 2017. In recent years, the velocity of data too has exploded with mobile devices generating nearly 12 exabytes of data (Kemp, 2018)[2] at speeds of 350 GB/second. These 3 "Vs" – Volume, Variety and Velocity – have come to define the evolution of data in the last few decades.

Storing, retrieving and managing this staggering volume of data has caused an evolutionary leap in technologies under the 'Big Data' umbrella. This originated within next-gen companies such as Google, Facebook, Twitter, and Yahoo that have been at the center of this data explosion. Data analytics has also become a crucial tool with Hadoop/MapReduce based architecture that effectively crunch large data sets. Today, real-time systems need solutions to handle relatively smaller data volumes even as they grapple with all the challenges presented by the 3 Vs.

How is next-gen data breaking relational database models?

The 1970s saw the growth of relational databases that, until now, have dominated the IT landscape. These databases can store data in structured formats with tight controls for data integrity. They also leverage a common language (SQL) to access and manipulate data. This allows enterprises to build massive systems of record that provide a trusted source of truth. With ACID compliance and distributed transactions, even mission-critical applications can guarantee data integrity. Once an INSERT or UPDATE transaction is committed, you are guaranteed that data will be available for access anytime in the future.

While the capabilities of relational databases are still quite valuable, new trends over the last decade are creating shifts in how companies manage data. For instance, engagement with the end user has become a key business driver. Thus, even while ensuring the integrity of the core financial transactions is critical, there are several use cases where the need for short response times outweigh the need for data integrity. In these cases, a traditional relational database management system (RDBMS) cannot meet expectations.

Relational databases are unable to handle unstructured data like audio/video files and social media posts. Moreover, the storage of large volumes of data in the RDBMS becomes cost prohibitive and performing any aggregation or analytics activities on the data is very difficult. RDBMS are designed to scale up and not scale out. This means the only way for an RDBMS to handle larger workloads is to deploy more CPU, RAM and hard disk on a single machine, which in turn limits scalability and increases cost. However, this is not the way modern cloud native software works.

The rigid data model at the center of any RDBMS guarantees integrity. But change is expensive. As agile development becomes the new paradigm, developers need database solutions that allow the structure to evolve easily.

Another important trend that has emerged over the last two decades is object-oriented programming. The highly normalized structure of RDBMS has resulted in an 'impedance mismatch' between the application layer and the database layer, calling for unusual constructs like object-relational mapping (ORM) frameworks.

What is NoSQL and why it is dominating the database market?

Several technologies have evolved to address the above challenges. As these technologies were compared with the traditional SQL-based RDBMS databases, they were grouped under the umbrella term of 'NoSQL' (which stands for 'Not Only SQL') or 'non-relational'. There are a number of different database types that are clubbed under this term. Columnar, key value store, document, and graph databases are the most popular ones. Each type has its own specific use case and a different underlying technology.

Columnar or Wide Column Stores

This database type is optimized to store large volumes of data in columns that can be queried efficiently. It is best-suited for ingesting large data volumes and creating visualizations and OLAP-like queries.

Some examples of these databases are Cassandra and HBase. Cassandra was used by Facebook to handle their huge data volumes that also needed to be searchable. Some examples of successful implementations of wide-column databases are:

- Spotify uses Apache Cassandra to support 40,000 requests per second for persistent data store[3]
- Ebay uses Cassandra to store 250 TB of data with 6 billion writes per day[4]

Key Value Stores

Here, data is stored as a 'key', which is the primary attribute of the 'value'. These databases are usually implemented as memory-only data stores to cache frequently accessed data. They are effective in scenarios that require the application to maintain a list of items that it refers to frequently such as user session information, master lookup information, currency rates, etc. Some examples of key value databases are:

- Microsoft uses Redis to power the MSN portal that gets 2 billion hits with a latency of less than 10ms on peak days[5]
- LinkedIn uses Couchbase to cache over 8 million real-time metrics (over 12TB of data). Over 16 million entries are loaded into Couchbase every 5 minutes[6]

Graph Databases

These databases store data in graph-like structures and use nodes and edges to represent the data and its connections. These are best-suited for specific use cases that require graph traversal and route optimization-like functionalities. For instance, it can be challenging for companies to determine the inter-relationships between users, transactions, locations, etc. A graph database supports such use cases for activities like fraud detection, compliance with legal restrictions on financial transactions, etc. Neo4J is a prominent example along with OrientDB that has multi-modal capabilities. Some examples of successful implementations of graph databases are:

- ICIJ used Neo4J to uncover relationships between people and their bank accounts, helping them 'follow the money' for investigative journalism related to offshore tax fraud[7]
- Walmart uses Neo4J to improve the speed and effectiveness of their online recommendation platform[8]

Document Databases

This is perhaps the most popular type of NoSQL database. Document databases organize information in JSON or XML documents that can be accessed in multiple ways including SQL-like languages. Modern applications represent business entities as objects that are best represented in a serialized form in documents. Here, the resulting impedance mismatch between the application and the database is much lower, enabling faster access and simpler programming. Document databases can efficiently store all kinds of entities like user profiles, transaction details, product catalogues, asset information, etc., in a highly scalable and performant manner. Some examples of successful document database implementations are:

- Craigslist uses MongoDB to store over 5 billion documents amounting to 10TB

of data with horizontal scaling through commodity hardware[9]

- Amadeus, the world's largest travel booking engine, manages more than 8 million queries per second and stores over 20 billion documents on Couchbase[10]

Since NoSQL is an evolving technology, the differentiation between its databases occasionally blurs. Over time, we may see some databases types being subsumed. For instance, databases like MongoDB can work as key value and graph databases even though the primary database type will offer richer functionalities in that specific area.

NoSQL as an alternative to RDBMS

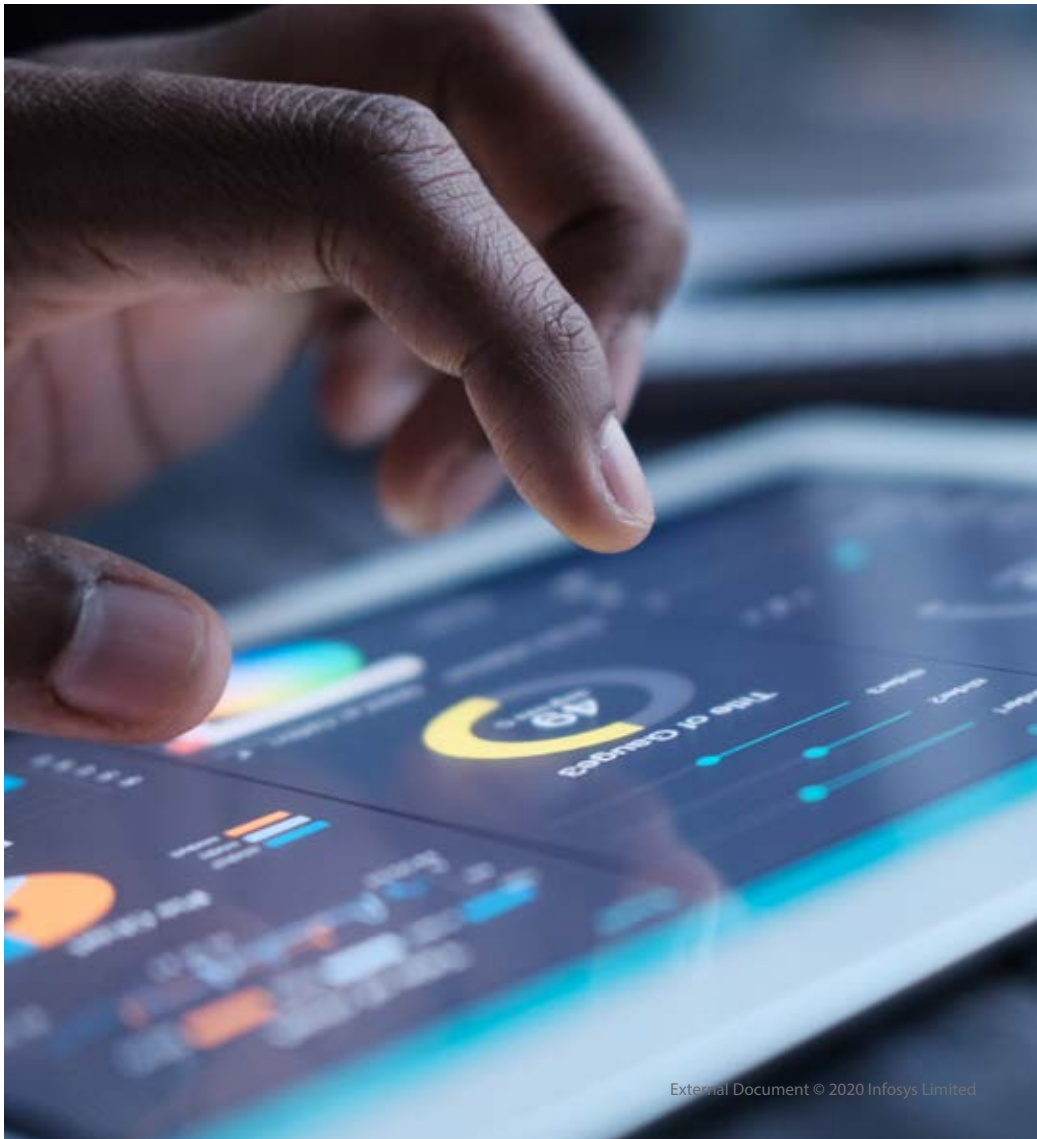
Enterprises should consider using NoSQL as an alternative to their existing databases in the following scenarios:

- New applications that are expected to store an evolving data structure, i.e., the data structure is expected to evolve as new business lines are added or as functionality is expanded. An RDBMS will usually have a pre-defined structure that is difficult to change
- Applications that are expected to be impacted by M&A should consider moving to NoSQL. The flexibility offered by NoSQL databases makes changing data structures relatively easy compared to RDBMS
- Applications that deal with streaming data (social media feeds, IoT) will have no alternative but to employ NoSQL databases because RDBMS technology is not built to handle the volume and speed associated with streaming data
- Applications employing RDBMS databases that are facing significant performance issues due to high user count or large data load can benefit significantly by deploying the right NoSQL database at the back-end
- Databases where data volume is expected to grow exponentially with a starting base of few GBs to multiple terabytes within a few months will need the horizontal scalability offered by NoSQL databases
- Any new agile development using modern languages like Java and JavaScript should consider NoSQL given the ease of development, low 'impedance mismatch' and high developer productivity
- Applications that need high availability, replication, failover, use of commodity hardware, and are polyglot (multi-language) must use NoSQL databases because the corresponding capability in RDBMS is typically offered through expensive add-ons
- Use cases like offline mobile apps, operational data stores which offload read traffic from mainframes, and customer 360 solutions lead to the automatic selection of a NoSQL database

Success stories

Infosys has helped several financial services companies implement NoSQL databases to meet their growing business needs. Some examples are:

- A retirement financial services leader was struggling with an Oracle-based system that ran for 20 hours to perform a monthly batch. Infosys migrated the system to a scoring solution based on Apache Spark and Cassandra that reduced batch processing time to 30 minutes
- Infosys helped a Europe-based credit rating agency implement a graph database that uncovered hidden relationships within complex holding patterns, allowing them to determine potential ownership patterns
- For a financial asset management company, Infosys built an operational data store (ODS) using MongoDB to unify data from multiple legacy platforms. The new database reduced the reporting latency, improved asset class utilization and improved risk monitoring
- Infosys deployed a flexible and scalable MongoDB database to help a global financial services provider improve client satisfaction through a user-friendly web interface for post-trade analysis



Conclusion

As the nature of data evolves, new technologies are emerging to support data and unlock greater value from the business. The three factors that will drive adoption of NoSQL are response time, data volume and overall cost goals. Enterprises looking to develop modern applications should carefully identify the challenges of their existing systems, the needs of the business and how NoSQL databases can bridge this gap. Infosys possesses deep expertise in NoSQL implementation along with a strong partner ecosystem, allowing us to offer end-to-end solutions that meet client expectations.

About the author

Vageesh Patwardhan

Vageesh Patwardhan is a Principal at Infosys. He leads the Open Source database practice and is a hands-on NoSQL Architect/Administrator. Vageesh has 22 years of experience in IT ranging from Mainframes to the latest NoSQL databases across multiple verticals.

References

[1] Forbes. [Online]. Available: https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/#7bb4ede360ba.

[2] Wearesocial. [Online]. Available: https://wearesocial.com/blog/2019/04/the-state-of-digital-in-april-2019-all-the-numbers-you-need-to-know.

[3] DataStax Inc., [Online]. Available: https://www.datastax.com/resources/casestudies/case-study-spotify.

[4] DataStax, "Ebay Case Study," [Online]. Available: https://www.datastax.com/resources/casestudies/ebay.

[5] Redis Labs Inc, [Online]. Available: https://redislabs.com/docs/microsoft-relies-redis-labs/.

[6] Couchbase Inc., "High Performance Applications with Distributed Caching" [Online]. Available: https://resources.couchbase.com/c/high-performance-wp?x=s9hNYZ

[7] Neo4J Inc, "International Consortium of Investigative Journalists Case Study," [Online]. Available: https://neo4j.com/case-studies/icij/.

[8] Neo4J Inc., "Walmart Case Study," [Online]. Available: https://neo4j.com/case-studies/walmart/.

[9] MongoDB inc., "MongoDB Case Study: Craigslist," [Online]. Available: https://www.mongodb.com/post/15781260117/mongodb-case-study-craigslist.

[10] Couchbase Inc., "Amadeus Powering the travel industry," [Online]. Available: https://www.couchbase.com/customers/amadeus.

For more information, contact askus@infosys.com

Infosys®
Navigate your next