WHAT IS CASSANDRA? KEY FEATURES AND ADVANTAGES



To fully appreciate Apache Cassandra and what it can do, it's helpful to first understand NoSQL databases and to then look more specifically at Cassandra's architecture and capabilities. Doing so provides a good introduction to the system, so you can determine if it's right for your business.

(This article is part of our Cassandra Guide. Use the right-hand menu to navigate.)

What is Apache Cassandra?

Apache Cassandra is a distributed database management system that is built to handle large amounts of data across multiple data centers and the cloud. Key features include:

- Highly scalable
- Offers high availability
- Has no single point of failure

Written in Java, it's a NoSQL database offering many things that other NoSQL and relational databases cannot.

Cassandra was originally developed at Facebook for their inbox search feature. Facebook open-sourced it in 2008, and Cassandra became part of the Apache Incubator in 2009. Since early 2010, it has been a top-level Apache project. It's currently a key part of the Apache Software Foundation and can be used by anyone wanting to benefit from it.

Cassandra stands out among database systems and offers some advantages over other systems. Its ability to handle high volumes makes it particularly beneficial for major corporations. As a result, it's currently being used by many large businesses including Apple, Facebook, Instagram, Uber, Spotify,

Twitter, Cisco, Rackspace, eBay, and Netflix.

What is a NoSQL Database?

A NoSQL, often referred to as "not only SQL", database is one that stores and retrieves data without requiring data to be stored in tabular format. Unlike relational databases, which require a tabular format, NoSQL databases allow for unstructured data. This type of database offers:

- A simple design
- Horizontal scaling
- Extensive control over availability

NoSQL databases do not require a fixed schema, allowing for easy replication. With its simple API, I like Cassandra for its overall consistency and its ability to handle large amounts of data.

That said, there are pros and cons of using this type of database. While NoSQL databases offer many benefits, they also have drawbacks. Generally, NoSQL databases:

- Only support simply query language (SQL)
- Are just "eventually consistent
- Don't support transactions

Nevertheless, they are effective with huge amounts of data and offer easy, horizontal scaling, making this type of system a good fit for many large businesses. Some of the most popular and effective NoSQL databases include:

- Apache Cassandra
- Apache HBase
- MongoDB

What makes Apache Cassandra database unique?

Cassandra is one of the most efficient and widely-used NoSQL databases. One of the key benefits of this system is that it offers highly-available service and no single point of failure. This is key for businesses that cannot afford to have their system go down or to lose data. With no single point of failure, it offers truly consistent access and availability.

Another key benefit of Cassandra DB is the massive volume of data that the system can handle. It can effectively and efficiently handle huge amounts of data across multiple servers. Plus, it is able to fast write huge amounts of data without affecting the read efficiency. Cassandra offers users "blazingly fast writes," and the speed or accuracy is unaffected by large volumes of data. It is just as fast and as accurate for large volumes of data as it is for smaller volumes.

Another reason that so many enterprises utilize Cassandra DB is its horizontal scalability. Its structure allows users to meet sudden increases in demand, as it allows users to simply add more hardware to accommodate additional customers and data. This makes it easy to scale without shutdowns or major adjustments needed. Additionally, its linear scalability is one of the things that helps to maintain the system's quick response time.

Some other benefits of Cassandra include:

- **Flexible data storage.** Cassandra can handle structured, semi-structured, and unstructured

data, giving users flexibility with data storage.
- **Flexible data distribution.** Cassandra uses multiple data centers, which allows for easy data distribution wherever or whenever needed.
- **Supports ACID.** The properties of ACID (atomicity, consistency, isolation, and durability) are supported by Cassandra.

Clearly, Apache Cassandra offers some discrete benefits that other NoSQL and relational databases cannot. With continuous availability, operational simplicity, easy data distribution across multiple data centers, and an ability to handle massive amounts of volume, it is the database of choice for many enterprises.

How does Cassandra work?

Apache Cassandra is a peer-to-peer system. Its distribution design is modeled on Amazon's DynamoDB, and its data model is based on Google's Big Table.

The basic architecture consists of a cluster of nodes, any and all of which can accept a read or write request. This is a key aspect of its architecture, as there are no master nodes. Instead, all nodes communicate equally.

While nodes are the specific location where data lives on a cluster, the cluster is the complete set of data centers where all data is stored for processing. Related nodes are grouped together in data centers. This type of structure is built for scalability and when additional space is needed, nodes can simply be added. The result is that the system is easy to expand, built for volume, and made to handle concurrent users across an entire system.

Its structure also allows for data protection. To help ensure data integrity, Cassandra has a commit log. This is a backup method and all data is written to the commit log to ensure data is not lost. The data is then indexed and written to a **memtable**. The memtable is simply a data structure in the memory where Cassandra writes. There is one active memtable per table.

When memtables reach their threshold, they are flushed on a disk and become immutable SSTables. More simply, this means that when the commit log is full, it triggers a flush where the contents of memtables are written to SSTables. The commit log is an important aspect of Cassandra's architecture because it offers a failsafe method to protect data and to provide data integrity.

Who should use Cassandra?

If you need to store and manage large amounts of data across many servers, Cassandra could be a good solution for your business. It's ideal for businesses that:

- Can't afford for data to be lost
- Can't have their database down due to the outage of a single server

It's also easy to use and easy to scale, making it ideal for businesses that are consistently growing.

At its core, Apache Cassandra is "built for scale" and can handle large amounts of data and concurrent users across a system. You can store massive amounts of data in a decentralized system, yet it still allows users to have control and access to their data.

Data is always accessible in Cassandra. With no single point of failure, the system offers true

continuous availability, avoiding downtime and data loss. It can be scaled by simply adding new nodes, so there is constant uptime and no need to shut the system down to accommodate more customers or more data. Given these benefits, it's not surprising that so many major companies use Apache Cassandra software.

What do you use Apache Cassandra for?

When you need to handle large amounts of data and must have dependable and fast access to it with a system that can massively scale, Cassandra's fault tolerance and high availability, with global scalability, are the answer. Here are some examples of common use scenarios and applications:

e-Commerce

Cassandra supports vital retail functions, from managing catalogs and shopping carts to inventory management. Customer expectations are high and meeting them is where Cassandra shines. Cassandra ensures zero downtime, fast responsiveness, scalability, and powerful analytics.

Entertainment websites

Websites like Netflix and Spotify are examples of global entertainment sites that use Cassandra. Cassandra empowers sites like these to serve millions of concurrent users with massive amounts of streaming data, user profiles, and viewing history. Cassandra also feeds data into recommendation engines, enhancing user experiences.

Internet of Things (IoT) and edge computing

IoT devices create massive and fast-changing data sets that require a flexible database. Cassandra data tiering can handle "hot" data that is fresh, the summaries and statistics that make up "warm" data, and older "cold" data that might be used for managing maintenance. New nodes can be added without any downtime.

Authentication and fraud detection

To effectively detect security threats, Cassandra makes it possible to analyze large, heterogeneous datasets in real-time to uncover patterns and breaks in patterns that may flag potentially fraudulent behavior. It also supports fast user authentication, without complexity or friction.

Messaging

Cassandra facilitates the sending and receiving of messaging at scale with real-time performance, scaling to handle heavy loads and replicating messages for easier re-routing around outages. It also supports storing conversations, threads, and metadata around messages and conversations.

Logistics and asset management

Whether it is tracking packages, containers, vehicles, or storage locations, Cassandra scales to handle the details of logistics operations without disruptive downtime. Tracking assets, routes, deliveries, inventory levels, and even adding additional fields for scans and sensors, is straightforward.

Limitations of Cassandra

Despite the power and obvious advantages of Cassandra, organizations do run into challenges in implementing and using it.

High maintenance costs

Apache Cassandra is open source, and thus costs nothing to deploy, but ongoing development and maintenance take time, talent, and money. Rather than wait for the development community to extend a feature or fix a bug, you may have to invest in those changes yourself, especially if you have service level agreements to meet.

Risks around security, regulatory compliance, and governance

Cassandra offers some security features, but they may not be adequate for your environment or the global compliance requirements across your markets. You may need to invest in additional capabilities and layers, particularly if you are operating in highly regulated industries and activities.

Patchwork of support and services

Your applications are likely to have been developed by a mix of open source, third-party vendors, and internal resources, each with differing levels of expertise and availability, so implementation and maintenance can feel ad hoc and disjointed.

Finding expertise

Talent with expertise in Cassandra is in high demand, and there's a limited supply of people with sufficient knowledge. People who want to gain that expertise read through open source documentation of varying levels of quality, seek help from community boards, and invest in time-consuming trial and error. Without the right partner, it may be difficult to get full value from a Cassandra deployment.

Related Reading

- BMC Machine Learning & Big Data Blog
- MongoDB Guide, a series of articles and tutorials
- Using Hadoop with Apache Cassandra
- Partition Key vs Composite Key vs Clustering Columns in Cassandra
- MongoDB vs Cassandra: NoSQL Databases Compared
- Data Storage Explained: Data Lake vs Warehouse vs Database
- CAP Theorem for Databases: Consistency, Availability & Partition Tolerance
- Data Ethics & Responsibility