Mission Modernization with Active Everywhere Architectures

DATASTAX
Active Everywhere—Every Cloud

CONTENTS

INTRODUCTION

For more than 40 years, the relational database management system (RDBMS) has been the dominant technology to power agency systems—including most of those in use today. It provides powerful mechanisms to store and query structured data under strong consistency and transaction guarantees and has demonstrated a high level of reliability, which was ideal for yesterday's workloads.

Today's mission demands are forcing agencies to rethink their approach to data management, as new use cases and operational needs are presenting new challenges. Modern agencies *must* support constantly connected users via modern applications that require access from anywhere in the world. Whether agencies are delivering citizen services to external users, supporting employees with mission-critical applications, or powering a Common Operational Picture (COP) for real-time mission visibility, this ability is essential.

In many modern application areas, relevant data has become so vastly distributed in such a variety of formats and moves at such high velocity that it cannot be properly handled by traditional database solutions. Additionally, data is frequently stovepiped, held in different systems, and often not defined in a uniform manner across the organization. Fusing user-generated information (structured and unstructured data) with data from large sensor networks to provide mission insight is one common example of where relational databases are being pushed past their intended limits. These were designed for fixed schemas and centralized workloads—not for today's high-velocity, high-volume, and high-variety data needs.

NOSQL DATABASES

More than ever, agencies today are relying on data to power their day-to-day operations. Their IT departments must support global access by millions of users on connected devices and by smart machines of all types. This not only requires geographic data distribution, but also the ability to handle massive amounts of reads and writes on the database in real time, without worry of downtime. The rapid rise of these new kind of data requirements led to a new class of data management systems called NoSQL databases which are being used in conjunction with traditional relational databases. To varying degrees, NoSQL databases offer the promise of horizontal scalability and higher availability, and they approach data management in a way that's useful for very large sets of distributed data.

A subset of NoSQL databases are designed for high-volume, web-scale applications in clustered environments, delivering the infinite scale and continuous uptime required to meet the evolving needs of government today.

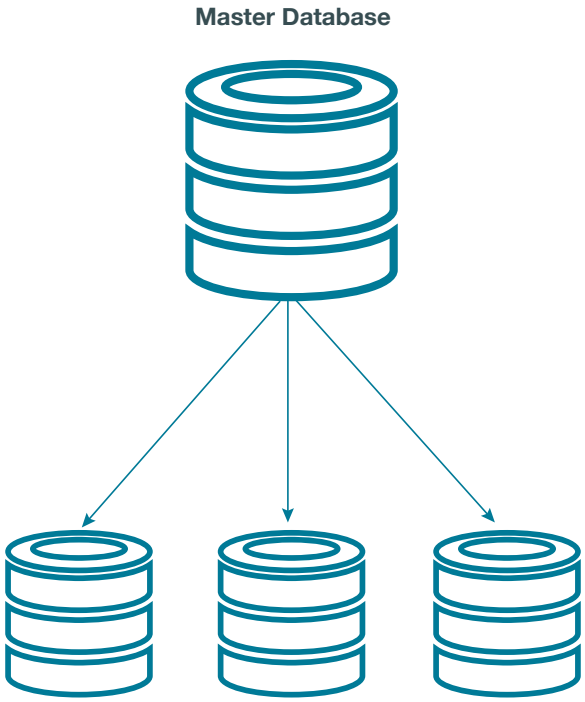
"HIGH AVAILABILITY" VS. "ALWAYS-ON" DATABASES

Many databases are designed to deliver "high availability." But what does that mean?

Conceptually, a highly available database uses an architecture designed to recover from hardware or network failures and resume normal operations. Often, this involves a significant footprint of redundant hardware, software, and other resources that sit idle a high percentage of the time. This model has existed for years; it was conceived to minimize the impact of failure across systems that were limited by technology constraints at the time.

Fundamentally, this architecture is designed for *disaster recovery*. Yet, how certain *are* agencies that all disasters can be recovered from? How often are these recovery systems tested? Are they tested using appropriate scenarios?

In contrast, a system architected to be "always on" is designed for *disaster avoidance*. It is purpose-built to eliminate failure altogether. This is why, intrinsically, a system designed for "high availability" is not necessarily a system designed to be "always on."



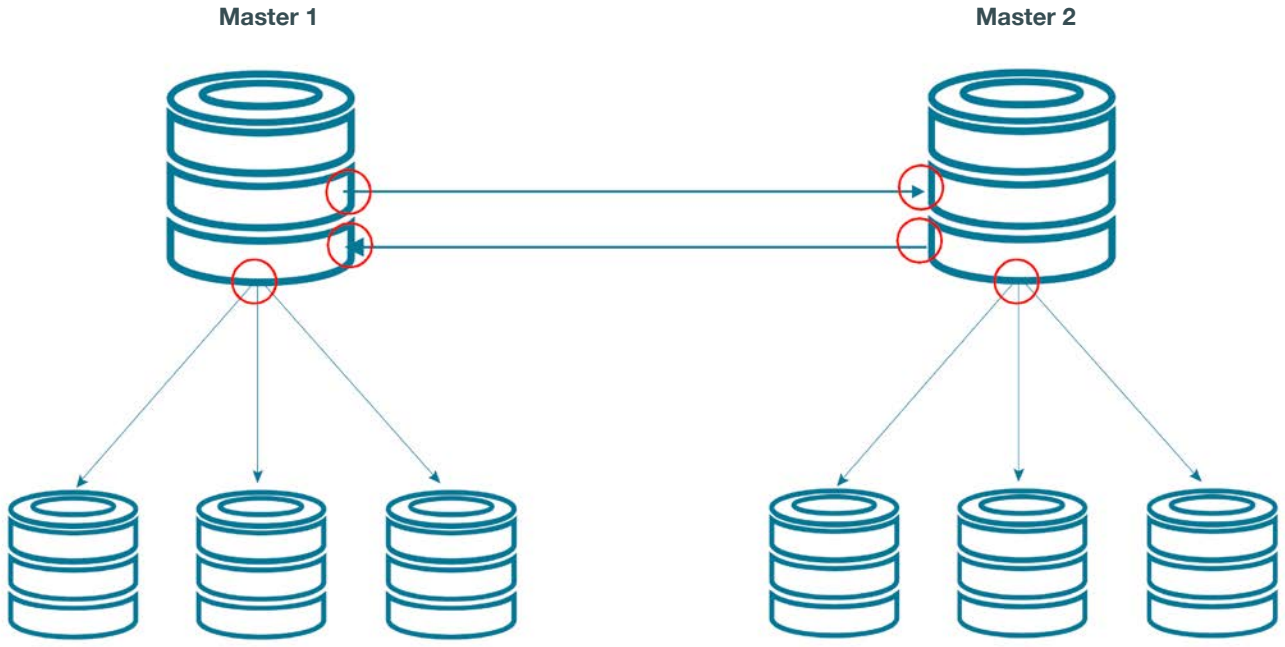
High Availability Databases

In the high-availability model, older (relational) database systems and many NoSQL database systems are built to run on a single server and rely on a master/slave architecture to provide availability. In many systems, all write traffic must be controlled by a single control point (the master), even as organizations add more machines to their database for scale. This model addresses the issue of read availability, with multiple slaves reading the data written by the single master. However, writes do eventually need to be propagated to slave nodes. As scale increases, this becomes a larger and larger issue—especially in more modern systems, which have increasingly higher amounts of write traffic. Anything involving sensors, IoT, video, and other high-volume data applications has to address the issue of write volume.

When data has to be distributed—such as in applications where users are located nationwide or globally—replication challenges can multiply. It also raises the question of which data is held locally, and where.

Another key challenge is avoiding downtime. In a master/slave architecture, when the master is out of service, the entire database is no longer fully functional. This means it will be unable to accept full read/write workloads until a new master is elected. Organizations may try to resolve this with a multi-master design strategy. However, this model struggles across broadly distributed networks because it introduces several individual points of failure. In the event of an outage, working to keep the entire database fully functional is a complicated endeavor that can negatively impact performance.

Issues compound in widely distributed applications that require more than two geographic locations for the database. When the above model is applied to a database spanning three, four, five, or N number of data center locations, complexity explodes and performance erodes. The multi-master model is not built for a world where data reads/writes *must* happen everywhere.



Always-On Databases

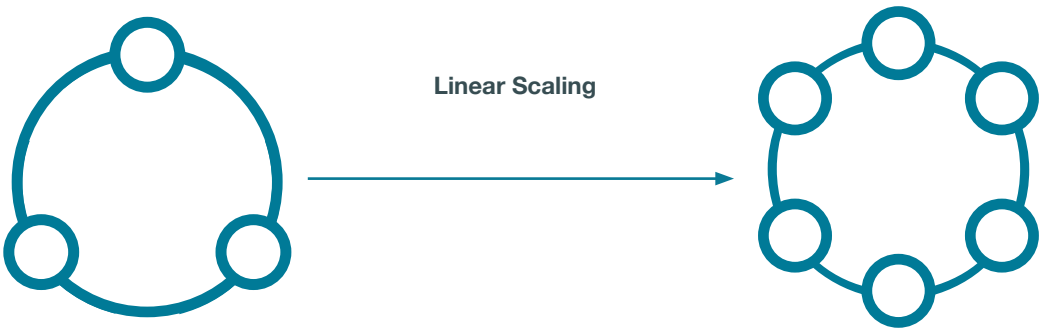
While the "high availability" model is designed to reduce downtime after a failure, an "always-on" NoSQL database model is intended to prevent downtime completely. It should be optimized to ensure that there is no interruption in service for the user when hardware or networks fail. Accordingly, this type of database needs to eliminate every possible single point of failure across its design, and be able to scale in vastly distributed environments without hurting performance.

It achieves this with a masterless architecture. Unlike the master/slave or multi-master design, every node in a masterless architecture plays the same role, and data is distributed evenly among them. Because there is no single master, there is no single point of failure. Any node can respond to a read or write request; if one node goes down, others automatically take over the request and operations continue without interruption. In this way, the system is truly "always on."

By design, this masterless model also supports high performance at scale. Because write traffic isn't controlled by only one master control point, it's easy to scale without creating bottlenecks and degrading performance. Since the workload is distributed proportionally across all nodes, scaling with additional nodes can actually *increase* performance.

DATASTAX ENTERPRISE, THE ACTIVE EVERYWHERE DATABASE

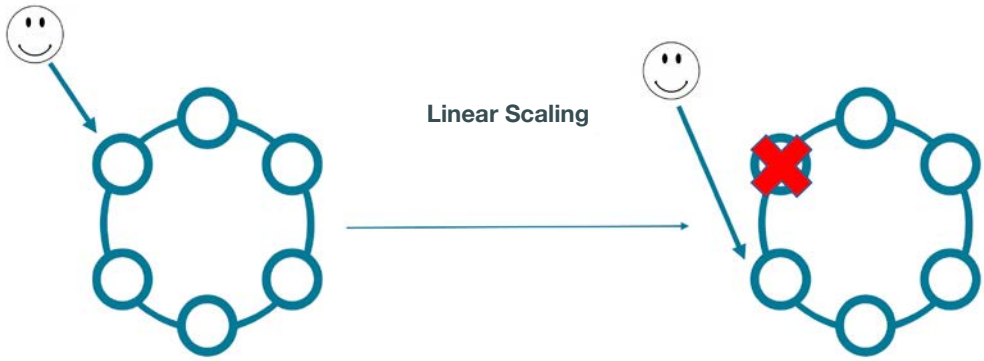
DataStax Enterprise (DSE) is the active everywhere database. With operational simplicity on a unified platform, this NoSQL database is purpose-built to address the challenge of high-velocity, high-volume, and high-variety data in distributed, hybrid cloud environments.



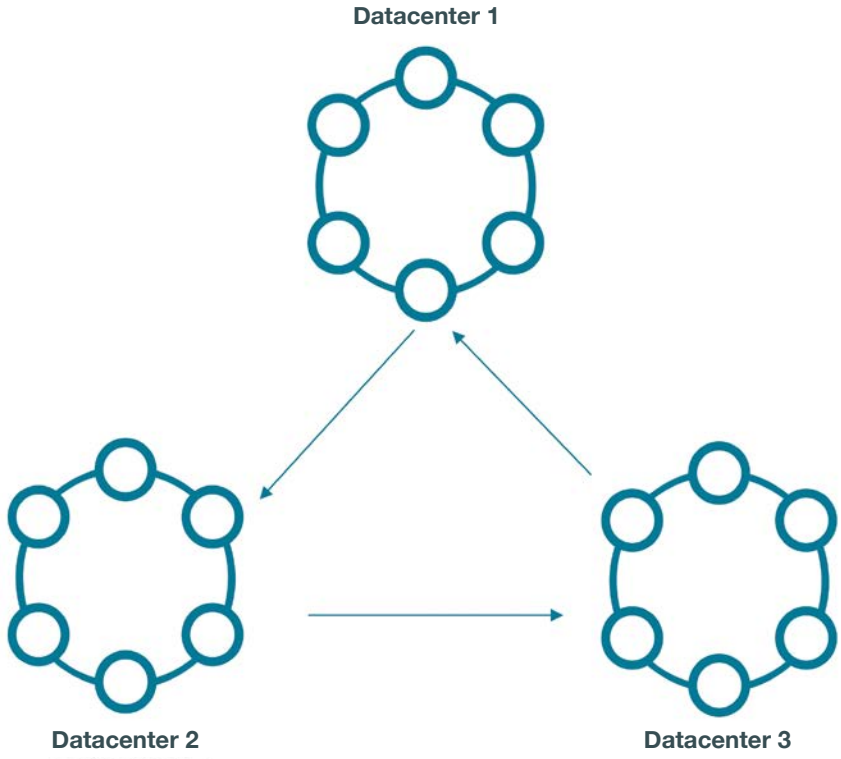
DSE is built on Apache Cassandra™'s masterless architecture, where multiple servers (or nodes) are grouped together in a cluster. With DSE, scaling is designed to be linear, so managing capacity is highly predictable. Doubling a cluster's performance is as simple as doubling the number of nodes in the cluster, which can be done infinitely. Moreover, as these systems scale, organizations can have their data closer to the application within any region in the world. This cuts data transfer costs and increases performance by reducing latency. In master/slave and multi-master models, this is very hard to achieve and comes with performance penalties. But with the masterless DSE, it is easily and automatically configured.

Continuous availability is built into DSE's core "always-on" architecture. Because it does not rely on a master, each node is identical, autonomous, and 100 percent active in function. This means that any node can serve an incoming request, and the data is then replicated across the cluster. The applications connecting to the database are automatically routed to all nodes that are up and functioning—without any special coding by developers and without any complicated failover procedures by administrators. If a node experiences an outage, other nodes will continue to service read

and write requests, providing system redundancy and minimizing the possibility of downtime. When the node comes back online, data is automatically synchronized with other nodes, and the node is fully operational to service read and write requests again.



Applying these same capabilities, spanning a DSE database across two or more data centers is very easy. Because there's no master, there are no concerns over which is primary, what needs to be replicated, and how failover happens. When a region fails or a data center goes down, DSE clients are smart enough to automatically reroute requests to the parts of the database that are still running in a different region. When the first region becomes available again, the clients will re-route the application back to the optimal performance connections. The database is automatically set for active-active configuration with all data being hot. And all of this happens *without any special coding* needed for database operation continuity.



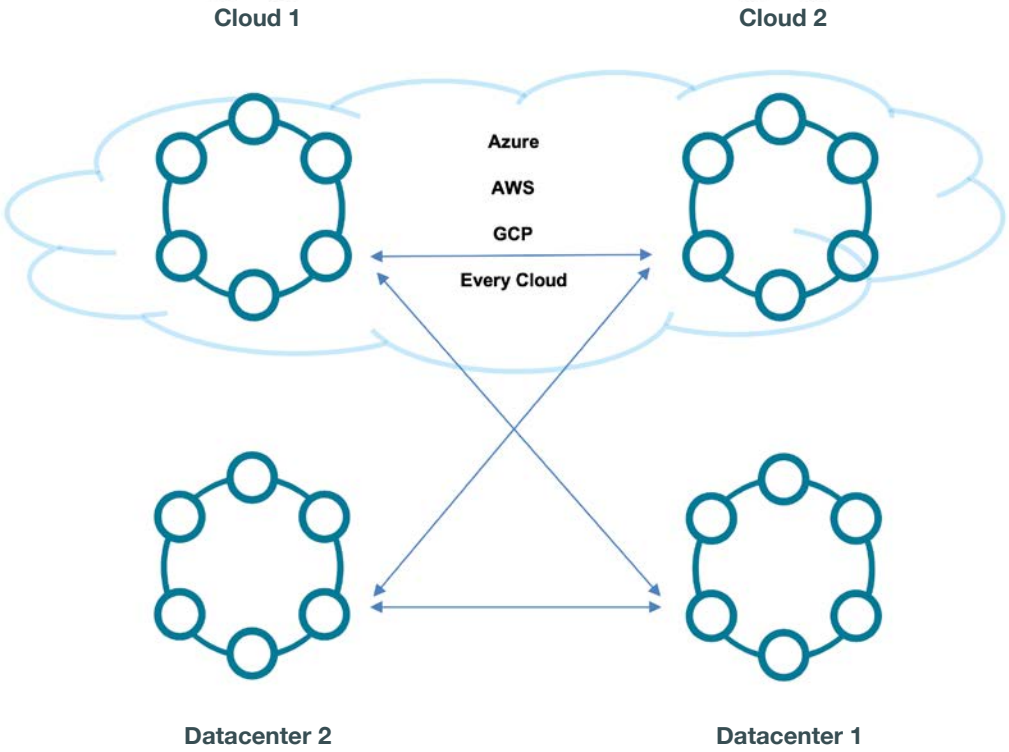
HOW DSE HELPS GOVERNMENT

DSE can help agencies modernize their systems and equip them to keep pace as technologies, policies, and mission demands evolve.

One use case is the adoption of public cloud infrastructure. While it offers many potential benefits to government agencies, it also presents operational challenges. For security policy reasons, most agencies have critical systems that will continue to reside on-premises in the near term. However, they must modernize their systems to be able to take advantage of cloud offerings in the future. So, how does an agency modernize a system with an on-premises architecture now, while enabling it to be "cloud native" when policy allows it in the future?

With DSE, expanding the same database into the cloud is just as easy as it is to move it between data centers. To move a part of the database into the cloud, it's as simple as standing up nodes with a cloud provider—*any* cloud provider—and having those nodes join the existing cluster. The data will rebalance to include the new nodes, with no further action required. To move the database entirely to the cloud, it's as simple as decommissioning the on-premises nodes and allowing the data to rebalance. Operations such as spanning multiple availability zones with one cloud provider, moving between cloud providers, spanning multiple providers simultaneously, or even moving the data back on-premises follow those exact steps: stand up new nodes, add to the cluster, decommission old nodes.

In multi-cloud deployments, the same database can be easily replicated into a different cloud. DSE creates a single, operationally simple database across multiple clouds that is shared by administrators, developers, and security teams. This avoids the data silos that typically come with multi-cloud deployments and cause inefficiencies in the system. Every node is fully active, ensuring uninterrupted uptime on premises, in a single cloud environment, or in multiple cloud environments.



With this operational simplicity on a unified platform, agencies can achieve faster time to value and reduce risk and cost in federal programs. DSE's "always-on" design ensures continuous availability of real-time data that's vital to those serving critical missions—ultimately saving lives and supporting mission success. By design, its masterless architecture offers an effective solution for managing diverse, high-volume, fast-moving data in distributed environments. With the ability to easily scale *while* improving performance, agencies can build and deploy their applications with a modern design that will support mission requirements, today and tomorrow.

ABOUT DATASTAX

DataStax delivers the always-on, active-everywhere distributed hybrid cloud database built on Apache Cassandra™. The foundation for personalized, real-time applications at scale, DataStax Enterprise makes it easy for enterprises to exploit hybrid and multi-cloud environments via a seamless data layer that eliminates the issues that typically come with deploying applications across multiple on-premises data centers and/or multiple public clouds.

Our product also gives businesses full data visibility, portability, and control, allowing them to retain strategic ownership of their most valuable asset in a hybrid/multi cloud world. We help many of the world's leading brands across industries transform their businesses through an enterprise data layer that eliminates data silos and cloud vendor lock-in while powering modern, mission-critical applications. For more information, visit www.DataStax.com and follow us on Twitter @DataStax.

Last Rev: MAR2019

Mission Modernization with Active Everywhere Architectures