Why Aerospike?

The Database Evolution

Today’s Data and the Age of the Customer

Today’s enterprises collect and process more data than ever before. The explosive growth of the Internet and mobile devices has fueled an exponential increase in the amount and types of information being generated and stored: social media data, server logs, clickstream data, machine data (the Internet of Things), and geolocation data.

This data explosion has led to the “Age of the Customer”. Today’s consumers expect to be able to access information about products and services in real-time, and they demand high-quality interactions with consistently fast response times. They expect instant gratification, and they will go elsewhere if you can’t provide it.

In the Age of the Customer, enterprises must store and process data in real-time in order to attract, engage, and retain consumers across devices, media, and channels. Applications have emerged to address new use cases such as fraud, risk, personalization, authorization and settlement, and other real-time web and mobile use cases. Effectively processing massive amounts of data in real-time creates a competitive advantage that can have an enormous impact on business.

Scaling RDBMS for Today’s Data

Since the early 70’s, Relational Database Management Systems (RDBMS) have been a critical piece of database infrastructure. In a relational database, data is stored in tables, columns, and rows, and is controlled by a rigidly structured schema. The schema ensures data consistency, but limits flexibility. RDBMS tables contain indices that allow data to be retrieved and updated using SQL (Structured Query Language), which enables complex operations across multiple tables.

Performance issues arise when relational databases are used with today’s transactional workloads. Relational databases are simply not designed to handle the volume of data and the real-time demands associated with modern enterprise applications. Also, the advanced features of relational databases are not required by many of today’s web applications, which tend to store and process relatively simple data structures, and don’t require complex SQL operations.

As businesses scale, their database architecture must also be able to scale without affecting performance. The database architecture must continue to keep datasets in sync and enable micro-second application processing that does not disrupt the end-user experience.

There are two ways to scale a database: scale vertically (“scale up”), by adding CPU, storage, and memory to individual servers or scale horizontally (“scale out”) by adding more servers in parallel.

Relational databases were designed to run on a single server in order to maintain the integrity of the table mappings and avoid the problems of distributed computing. A RDBMS server can be scaled up by adding CPU, storage, and memory, but you eventually reach an upper limit that still cannot adequately address today’s performance requirements.

To address these issues, relational database vendors have created more complex “master-slave” architectures, in which the “slaves” are additional servers that can handle parallel processing and replicated data, or data that is “sharded” (divided and distributed among multiple servers) to ease the workload on the master server.

These new RDBMS architectures add complexity, cost, and some significant drawbacks. A distributed relational database typically is based on pre-defined queries, so flexibility is sacrificed for performance. Also, a distributed relational database is not designed to scale down. Once data has been distributed onto additional servers, it is very difficult to “undistribute” that data.

Modern Database Requirements

To deal effectively with the volume and variety of today’s data, and to meet the need for sub-millisecond response times, a modern database must include the following characteristics:

- Scalability – The database must be able to scale dynamically up to Terabytes or Petabytes of data, and up to thousands or millions of users.
- Flexibility – The database must be able to deal with unstructured and semi-structured (“schema-less”) data.
- Performance –The database must provide fast (~1 millisecond) transaction response times at scale.
- Agility – The database must provide developers with the ability to rapidly create and adapt applications.
- Persistence – Many enterprises must comply with regulations and requirements that dictate storing operational data persistently to disk.

NoSQL Databases

NoSQL technologies have been developed to address some of the issues associated with processing massive amounts of data. A NoSQL database can be used to solve new problems that require:

- Scalability -- A NoSQL database can scale horizontally to the massive scales required by today’s workloads. Applications can run in parallel on a Cloud-based cluster composed of dozens, hundreds, or even thousands of commodity servers. The NoSQL scale-out architecture enables web applications to scale dynamically up to thousands or millions of users. As the number of concurrent users grows, you can dynamically add more cluster nodes (commodity servers) to process the additional load.

- Flexibility – A schema-less NoSQL database can process and store unstructured and semi-structured data such as social media data, server logs, clickstream data, machine-to-machine data, and geolocation data. With RDBMS you cannot add data (columns) without updating the entire schema, which can take hours or even days, depending on institutional infrastructure. With NoSQL, data can be added and updated flexibly and efficiently.

- Speed – The parallel processing nature of a NoSQL database, along with caching and aggregation, can provide fast (sub-millisecond) transaction response times at scale. Developer productivity and agility – A NoSQL database enables developers to quickly create and adapt applications to take advantage of emerging business opportunities. With NoSQL, new applications can be rapidly prototyped, tested, and deployed to a Cloud-based cluster.

Adding a Caching Layer to an Operational Database

The need for sub-millisecond response times has led to the use of in-memory databases in a caching layer. Database caching is used with a wide variety of underlying datastore types including RDBMS, NoSQL, and legacy databases and transactional systems.

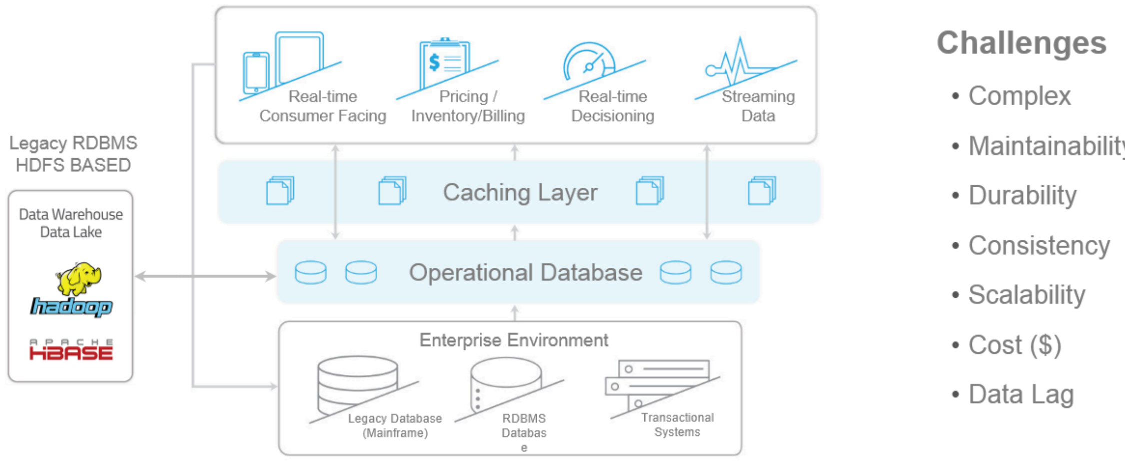
An operational database is used to manage dynamic data in real-time. For example, an operational database might be used to take and fulfill orders in an online store, and to keep track of payments, shipments, and inventory. As the database of record for transactional data, it contains the detailed data used to run the day-to-day operations of a business. The data constantly changes as updates are made, and reflect the current value of the most recent transaction. An operational database, as the name implies, is the database that is used to capture real-time data and supply data for real-time computations and workloads.

With a caching layer on top of an operational database, performance is improved by keeping frequently accessed data in system memory so that it can be retrieved quickly.

Many enterprises must comply with regulations and requirements that dictate storing operational data persistently for longer periods of time. These writes to disk consume system resources and slow down transactions (increase latency), and the workloads keep increasing with today's ever-expanding volumes of operational data. In general, as operational data volumes increase, operational database transaction speeds decrease.

To meet service-level agreements (SLA) as customer data expands exponentially, organizations of all sizes are turning to in-memory solutions to scale applications, offload over-burdened shared data services, and provide availability guarantees.

Increasingly, enterprises are turning to hybrid transactional-analytical processing database environments that can manage data in memory. Together these databases form an architecture stack that enable enterprise applications to provide personalized services — in which personal preferences, a customer's current location, external factors such as weather and time, and a business's inventory are accessible and analyzed in real-time in order to enable a purchase or provide a service.



Caching Layer Advantages

A caching layer with an operational database offers the following advantages:

- Speed – The primary advantage of adding a caching layer on top of an operational database is speed. With data immediately accessible in memory, much faster transactional response times can be achieved.
- Scalability – Distributed caching can help enable databases to scale horizontally.

Caching Layer Disadvantages

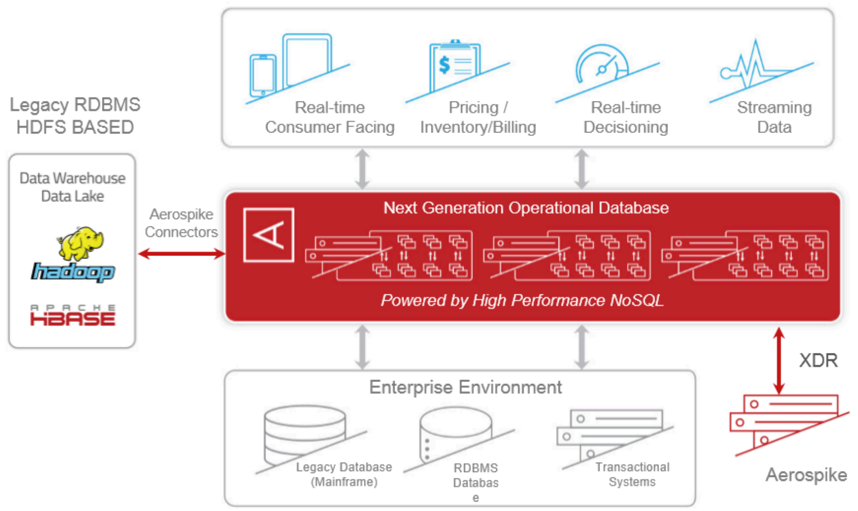
Unfortunately, a caching layer with an operational database includes some significant drawbacks:

- Caching strategies routinely fail for use cases with heavy write loads.
- Adding an additional caching layer increases complexity, which increases development and maintenance costs.
- Features such as cluster management, scaling, and data replication must be implemented manually.
- RAM is more expensive than disk drives or solid state drives, so in-memory datastores increase operating costs.
- In-memory datastores are extremely fast, but they do not work with data sets that exceed the available RAM.
- Persistence of data is an issue. In-memory data is not saved in the event of a server crash or power loss, so there is the potential for data loss between writes to persistent disk storage.
- Writes to persistent disk storage consume system resources and slow down transactions.
- Data consistency can be an issue. “Eventual consistency” is not acceptable for many use cases.

Aerospike: Speed at Scale without a Caching Layer

What if you could achieve in-memory speed without the added complexity, cost, and drawbacks of a caching layer? As the only flash-optimized NoSQL operational database, Aerospike makes this possible.

Aerospike is engineered to deliver in-memory performance while running exclusively on SSD (flash) drives. With Aerospike you get the sub-millisecond transaction speed of an in-memory database, the scalability and flexibility of a NoSQL database, and a host of other features such as data persistence, ACID compliance, 5-9s uptime, data replication, and dynamic scaling – all without a caching layer.



Benefits:

- Simplicity
- Maintainability
- Durability
- Consistency
- Scalability
- Cost ($)
- Data Lag Reduced

Aerospike is a next-generation open source NoSQL operational database engineered for high throughput with low latency, horizontal scalability, reliability, and operational efficiency. Aerospike delivers speed at scale, predictable performance, persistent data and high availability, all a fraction of the cost of other operational databases.

Aerospike Features and Benefits

Speed at Scale

Cache-based RDBMS and NoSQL solutions use RAM (memory) to achieve useful performance speeds. RAM provides less storage capacity than hard disk drives (HDD) or solid state drives (SSD, also referred to as flash), and is also more expensive.

Aerospike uses an optimized architecture with hybrid RAM/SSD storage, so you can achieve RAM-like performance without a caching layer.

Aerospike is the only NoSQL database specifically designed to optimize performance on SSD drives. Unlike other NoSQL solutions which were designed to write to rotating disks, Aerospike's unique storage layer incorporates both memory and flash storage, providing in-memory characteristics at the price of SSDs. This hybrid system is optimized for operational efficiency – indices are stored in a very tight format to reduce memory requirements.

Aerospike Hybrid Memory System Features

- Aerospike bypasses the Linux file system and writes data directly to raw disk in large blocks to minimize latency.
- Normal file systems write data on separate parts of a disk. Aerospike stores indices (via the primary Key) in DRAM for instant availability. To minimize DRAM usage, the size of an individual

index has been painstakingly minimized. At present, Aerospike can store indices for 100 million records in only 7 Gigabytes of DRAM.

- By storing data persistently in SSDs, record data is stored contiguously with automatic defragmentation and eviction, and is multi-threaded.
- With Aerospike, parallelism is powerfully utilized both within a node as well as across nodes, achieving the best performance by scaling up on one node and scaling out across nodes using DRAM and SSDs. Access is optimized for how flash works – with small block reads and large block writes – and parallelized across multiple SSDs for better throughput.

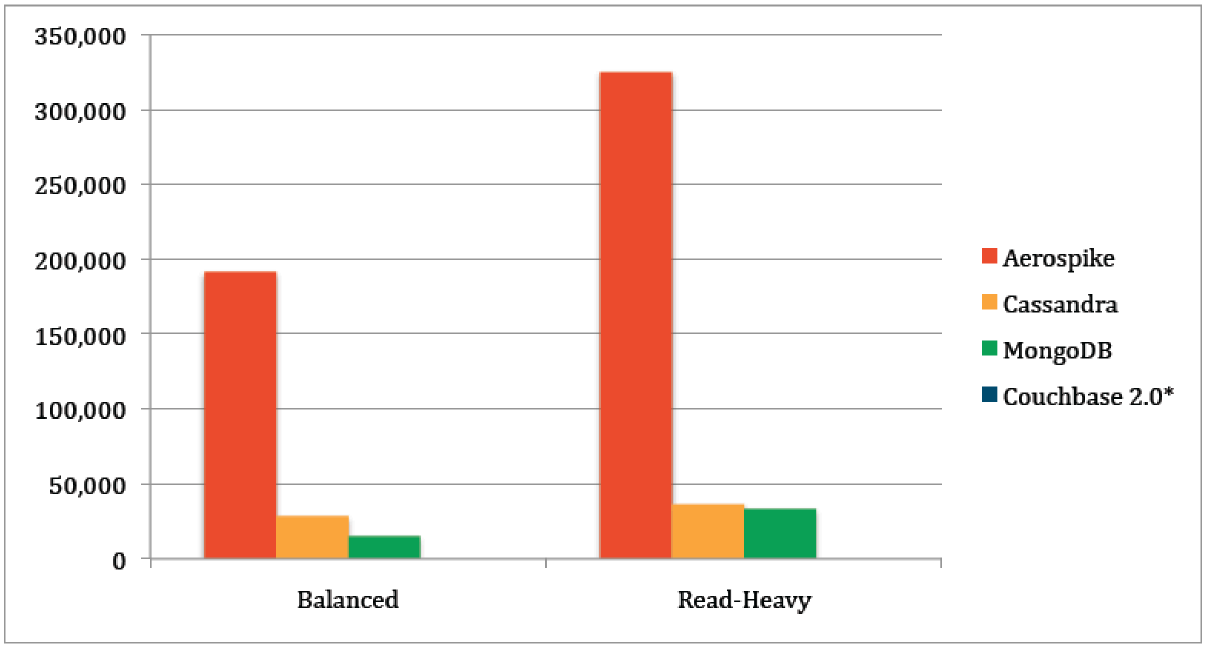
"We've known the innovators at Aerospike for a few years now, and today we are announcing more than 1 million transaction per second (TPS) on a single server with Aerospike's NoSQL database. That might not seem like such a big deal, until you realize we are not using DRAM for this…"

– Frank Ober, Data Center Solutions Architect, IT Peer Network, Intel

Predictable Performance

Today's mission-critical use cases require predictable sub-millisecond latency for reads in the presence of heavy write loads. Aerospike scales from thousands to millions of transactions per second (TPS), up to Terabytes of data, and delivers consistent sub-millisecond latency at all scales. And Aerospike scales dynamically, so you can easily add or remove nodes to adjust throughput or storage capacity, even while under production load.

In an independent benchmark test of NoSQL databases conducted by Thumbtack Technology, Aerospike out-performed the competition, delivering nearly 10 times the throughput. The Thumbtack benchmark used Yahoo! Cloud Serving Benchmark (YCSB) to simulate a high-throughput environment that is typical for many consumer-facing website applications.



As shown in the graph, Aerospike achieved nearly 200,000 transactions per second (TPS) in balanced read-write tests and over 300,000 TPS in read-heavy tests. That's the equivalent of less than 1 millisecond per transaction, on average. Cassandra and MongoDB delivered less than 30,000 TPS under the same loads, and Couchbase failed to complete the test. In early 2014, Aerospike performance doubled on the same YCSB tests.

According to the report:

"Aerospike was the dominant performer, showing durable, replicated behavior 5-10 times faster than what the others could achieve. Even when others were set to asynchronous replication, Aerospike retained this huge advantage."

After completing the benchmark, Thumbtack concluded:

"The most striking result was the raw throughput number Aerospike was able to achieve even while committing to disk across multiple nodes. We expected it to shine in this case, but maintaining a speed of 200 thousand operations per second with these strong guarantees put it far ahead of its nearest competitor and at speeds we would not normally associate with ACID semantics."

Aerospike is currently being used in many mission-critical production applications that leverage data sets larger than 100 Terabytes, and is the best choice for applications that require dynamic, cost-effective scalability with predictable performance.

"We keep adding servers, and the Aerospike database has scaled horizontally flawlessly. Scaling is hard, and having a good database is key. Aerospike is a success story and continues to play a central role in our ability to grow to meet global demand."

– David Pickles, The Trade Desk's co-founder and CTO

High Availability

Aerospike started being used in production in 2010, and since then has run non-stop with five-nine (99.999%) availability in some of the world's most demanding, large-scale deployments.

The unmatched level of uptime our customers have experienced can be attributed to Aerospike's automatic rebalancing, rolling upgrades, fault-tolerance, background backup and restore, and cross data center replication — ensuring no degradation to performance and availability even in the most demanding environments.

With Aerospike, high-availability is built in – there's nothing to configure. Aerospike's distributed "Shared-Nothing" architecture is designed to reliably store data with automatic failover, and to provide replication at the server level to gracefully handle failures. It's essentially self-managing.

"For us 100% uptime is the top metric of success, and that's what we've been able to achieve with the Aerospike real-time database."

– Michael Yudin, CTO, AdMarketplace

Lower TCO

What if you could out-perform your competitors by achieving superior speed at a fraction of the cost? As the first SSD-optimized NoSQL database, Aerospike makes this possible.

Aerospike can reliably manage millions of transactions per second and large data volumes with low latency. Because the Aerospike architecture optimizes hardware efficiency, customers can often reduce production hardware footprints by up to an order of magnitude without sacrificing performance.

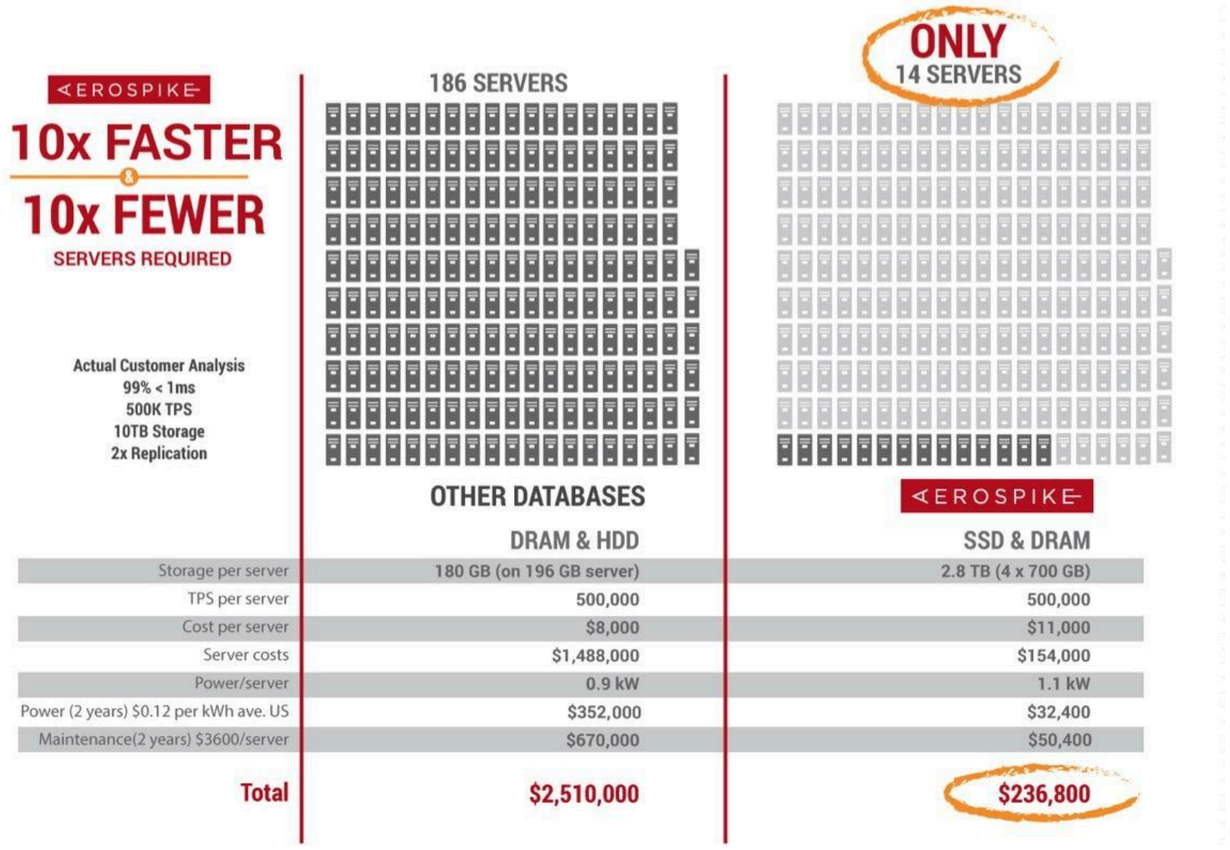
Cache-based in-memory solutions may work fine if the application that solves your business problem fits within a few hundred Gigabytes, but problems arise when you inevitably need to scale out. You start buying a lot of expensive RAM and paying for many more servers, and you find interesting applications that are not cost-effective at scale – where Terabytes of additional storage would create a much more compelling application.

In a typical use case, a modest social networking company using a first-generation in-memory NoSQL solution might be using several Terabytes of memcached servers for their existing applications – already a substantial investment in RAM and servers. When they try to broaden their reach and build applications that offer more features for each user request, they face two unpleasant choices:

- Push the limits of their existing cache and possibly thrash the caching layer and bring down their services.
- Beef up their cache tier by buying more RAM. Quite possibly a lot of RAM.

With Aerospike, you don't need to choose between limiting application development and spending a lot of money. Aerospike out-performs cache-based databases, and scales to Terabytes of data without the need to buy expensive additional RAM.

For a given load running on an operational database supplemented with a caching layer, that same load can typically run on an Aerospike cluster with one-tenth the number of servers, thereby reducing total cost of ownership (TCO) by a factor of ten.



	OTHER DATABASES DRAM & HDD	AEROSPIKE SSD & DRAM
Storage per server	180 GB (on 196 GB server)	2.8 TB (4 x 700 GB)
TPS per server	500,000	500,000
Cost per server	$8,000	$11,000
Server costs	$1,488,000	$154,000
Power/server	0.9 kW	1.1 kW
Power (2 years) $0.12 per kWh ave. US	$352,000	$32,400
Maintenance(2 years) $3600/server	$670,000	$50,400
Total	**$2,510,000**	**$236,800**

And the long-term outlook only gets better. SSD prices continue to fall even as storage capacity increases, and numerous industry sources agree that the cost and capacity of SSD drives will be equivalent to that of disk drives by the end of 2016.

Complement and Extend Your Legacy Systems

Aerospike is a next-generation open source NoSQL operational database engineered for high throughput with low latency, horizontal scalability, reliability, and operational efficiency. Aerospike delivers speed at scale, predictable performance, and high availability, all a fraction of the cost of other operational databases.

You should consider Aerospike if you need to store large amounts of data and your application requires response times in milliseconds or microseconds. Aerospike enables you to meet challenges and take advantages of business opportunities on a whole new scale – a scale that is simply not feasible with relational or NoSQL databases supplemented with a caching layer.

Aerospike is ideally suited for operational edge workloads that are directly responsible for revenue. A faster and more cost-efficient database allows you to create and deploy more applications with more features.

Next Steps

- See how much you can save with the Aerospike TCO calculator.
- The Aerospike Quick Start program is designed to help you fully leverage our fast, reliable, high-performance NoSQL key value database for real-time workloads, and helps you to better understand the business benefits we can offer.

  The Aerospike Quick Start Program includes:

  - (2) Admin and/or (2) Developer Online Training Classes
  - (8) hours of remote consulting
  - Cluster sizing and benchmark tool support for optimizing your implementation
  - Access to Enterprise Support during the 6 month contract term