

AtalantaDB

AtalantaDB Load Testing Report

Comparative Throughput and Latency: AtalantaDB, Aerospike, and Cassandra

Contents

1. Introduction and editions
 2. Test environment and methodology
 3. Single-node results: replication factor 1
 4. Three-node results: replication factor 2
 5. Latency
 6. Findings and supporting data
- Appendix A. External reference material

atalantadb.com

1. Introduction and editions

AtalantaDB is a distributed key-value database designed for high-throughput workloads and large datasets. This report presents 21 INSERT, UPDATE, and GET test cases comparing AtalantaDB with Aerospike Community Edition and Cassandra. The tests cover single-node and three-node deployments, single-record requests, and batches of 1,000 records.

AtalantaDB is available in two editions:

- Community Edition (CE): open-source software distributed under AGPLv3, with no limits on cluster size or data volume. CE is intended for trusted environments because access control is available only in EE. Source code: <https://github.com/atalantadb/server-community-edition>
- Enterprise Edition (EE): a proprietary edition with additional functionality, including access control. Features relevant to the results in this report include Read Cache and append update, which is enabled by default in EE. Product information: <https://atalantadb.com>

2. Test environment and methodology

2.1 Hardware

Database servers:

- CPU: 80 logical processors / hardware threads, 3.1 GHz.
- Storage: four 6 TB NVMe drives, using XFS.
- Memory: 750 GB.

Load generators:

- Number of hosts: 10.
- CPU per host: 80 logical processors / hardware threads, 2.3 GHz.
- Worker threads per load generator: 50.

The 50-thread setting was selected after higher thread counts caused Aerospike errors, including Device overload and Max error rate exceeded. The thread count was reduced until these errors no longer occurred.

2.2 Test tool, operations, and metrics

The tests used YCSB. The source code of the implementation used for these measurements is available at: <https://github.com/atalantadb/YCSB>

Three operation types were tested separately:

- INSERT: initial data loading.
- UPDATE: replacement of existing values with newly generated random values.
- GET: retrieval of previously loaded records.

Each column value was randomly sized between 50 and 150 bytes, with an average of approximately 100 bytes. The workloads used either one or ten columns per record. Single-record requests used a batch size of 1; batch workloads used 1,000 records per request.

Each load generator wrote to its own table. In AtalantaDB, each table had 100 regions. This partitioning is important for AtalantaDB because it enables efficient parallel processing of data across regions.

The throughput charts show operations per second per database node over time. Latency measurements are presented separately in Section 5. Chart labels are defined as follows:

- ADB_CE: AtalantaDB Community Edition.
- ADB_EE: AtalantaDB Enterprise Edition.
- AS_CE: Aerospike Community Edition.
- CS: Cassandra.
- AS_EE_AF: an illustrative All-Flash model, as defined in Section 2.5.

2.3 Deployment configurations and test matrix

Single-node tests used replication factor 1. This configuration provides a controlled baseline for comparing implementations without introducing the network effects associated with a multi-node deployment.

The cluster tests used three database instances with replication factor 2. They cover both Cassandra and Aerospike comparisons.

The 21 test cases are grouped as follows. Each group contains separate INSERT, UPDATE, and GET tests, in that order.

Test IDs	Deployment	RF	Columns	Batch size	Comparator
01-03	1 node	1	1	1	Aerospike CE
04-06	1 node	1	10	1	Aerospike CE
07-09	1 node	1	1	1,000	Aerospike CE
10-12	1 node	1	10	1,000	Aerospike CE
13-15	3 nodes	2	10	1	Cassandra
16-18	3 nodes	2	10	1	Aerospike CE
19-21	3 nodes	2	10	1,000	Aerospike CE

AtalantaDB CE and EE are compared in each group. The AS_EE_AF series, where shown, is not an additional measured test case.

2.4 Aerospike CE capacity and update procedure

The Aerospike Community version used in the tests was 8.2.0. Its reported capacity limit was two namespaces of 640 GB each, or 1,280 GB in total.

Aerospike appends updated data to free blocks rather than immediately overwriting existing records. The workload planning therefore allowed for the recommended 50% free disk space for updates. For the single-node test, this corresponds to a nominal data target of 640 GB out of the 1,280 GB limit.

The initial one-column workload inserted 500 million records from each of the 10 load generators, for a total of 5 billion records. With an average value size of 100 bytes, the nominal value payload was approximately 500 GB. The actual stored size was approximately 700 GB, close to the 640 GB planning target.

Aerospike defragmentation does not start immediately. Two update passes were performed, and the results from the second pass were used to include the effect of defragmentation in the comparison. Each pass replaced the values with new random values in the same 50-150 byte range.

2.5 Aerospike EE All-Flash: illustrative model

Aerospike EE was not benchmarked directly because of the benchmarking and competitive-analysis restrictions cited in Aerospike MSA Section 7.4(e).

The AS_EE_AF series is an illustrative model derived from the measured Aerospike CE series. The rationale for the selected coefficient is a comparison of Aerospike's own published reports using similar workloads on different hardware. In both referenced configurations, the replication factor was 2 and the record size was approximately 2 KB. The hardware differed.

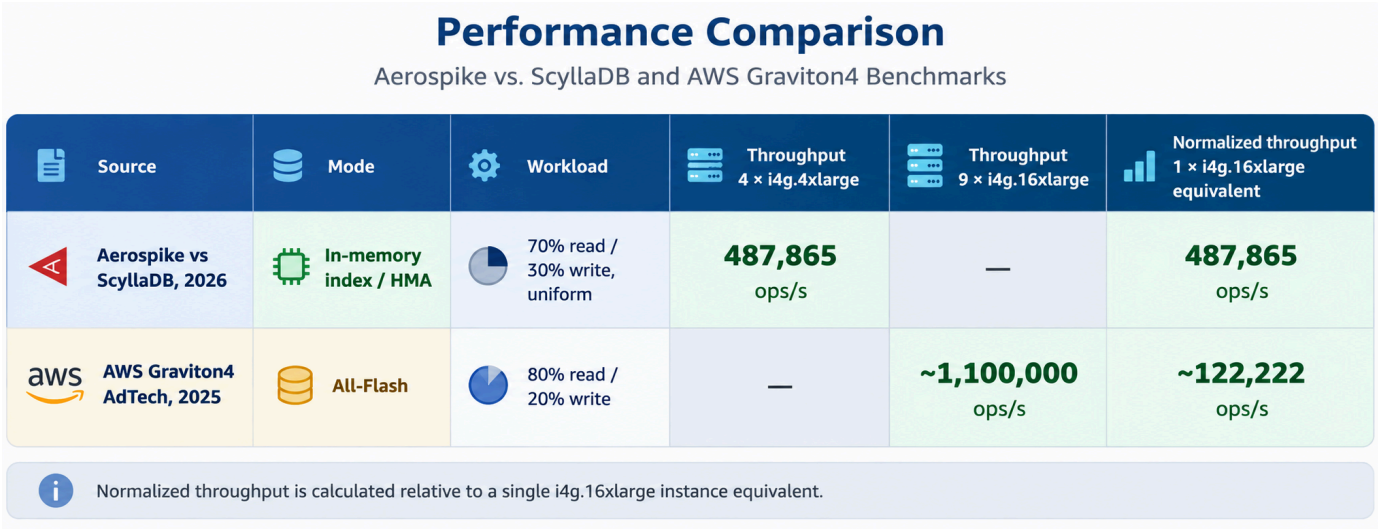


Figure 1

Assuming performance scales approximately with the hardware characteristics, that comparison suggests a slowdown of up to four times. A more conservative coefficient of two was selected for the illustration:

AS_EE_AF throughput = measured AS_CE throughput / 2

This series illustrates the selected coefficient. It is not a measurement or a validated estimate of Aerospike EE All-Flash performance. It must be read separately from the directly measured database results.

3. Single-node results: replication factor 1

3.1 Single-record requests, one column: tests 01-03

Each load generator inserted 500 million records with one column per record. UPDATE covered the same 5 billion records in total, with statistics taken from the second update pass. GET issued 500 million reads per load generator.

Test 01. INSERT

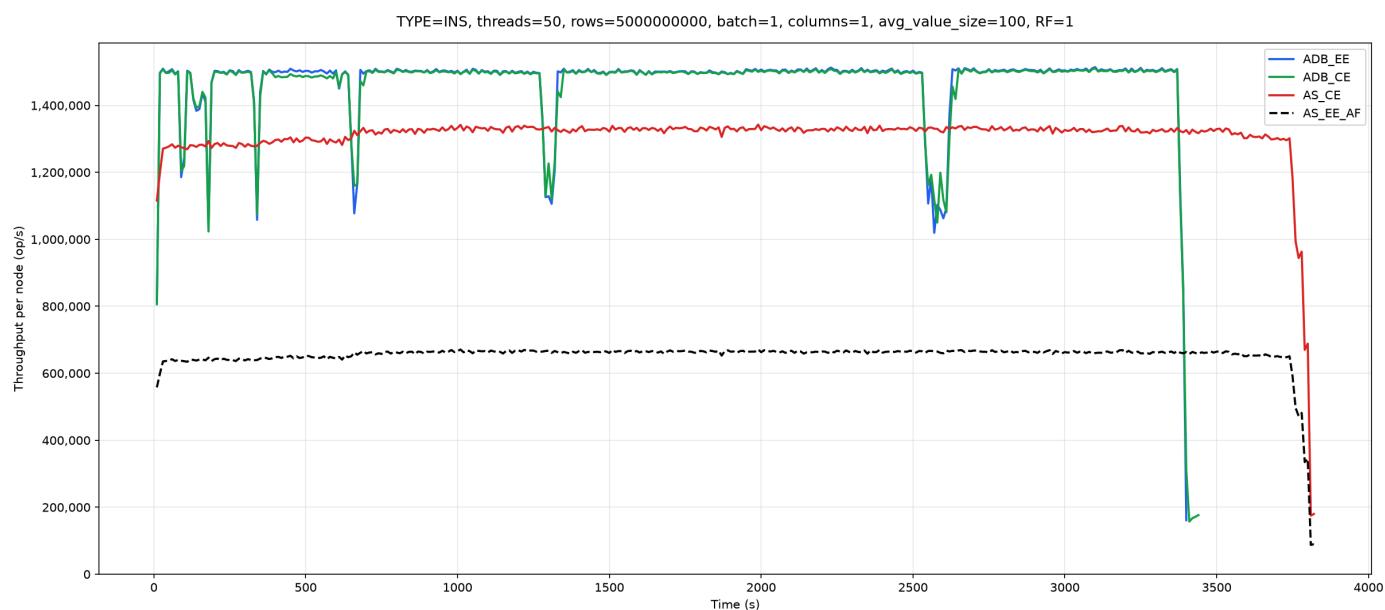


Figure 2

The temporary throughput drops in AtalantaDB correspond to index resizing. As the index grows, these events become less frequent. If the expected key count is known in advance, it can be specified when the table is created to avoid intermediate index resizes.

Test 02. UPDATE, second pass

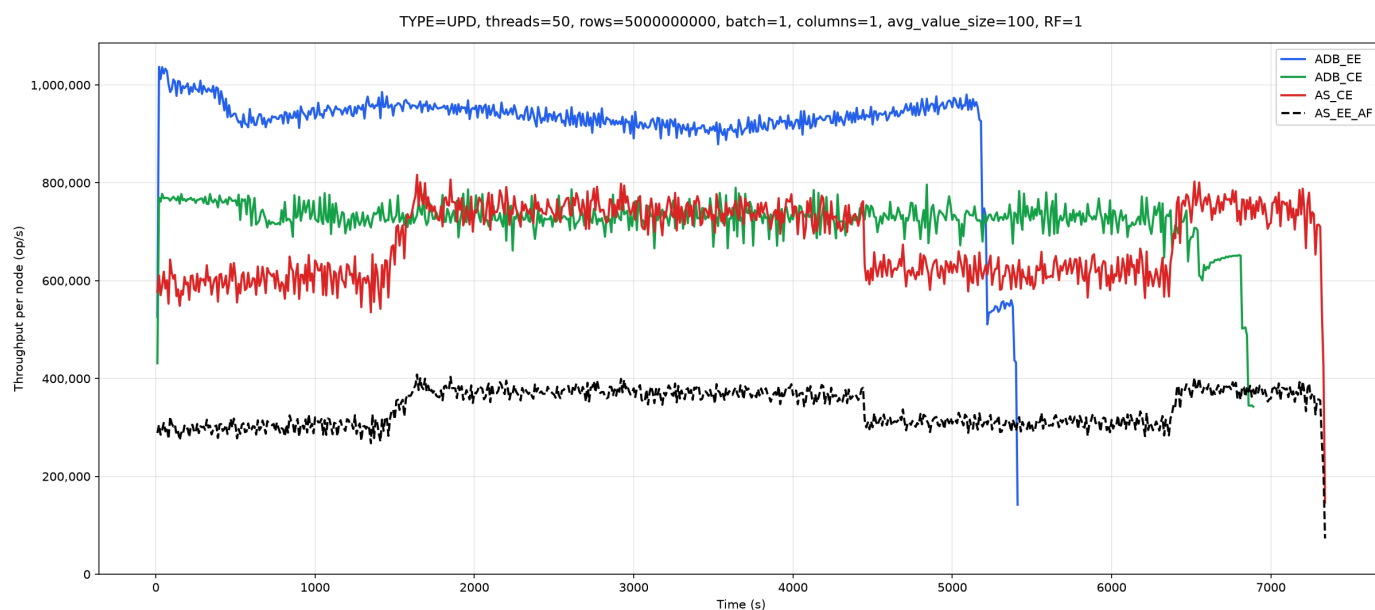


Figure 3

Test 03. GET

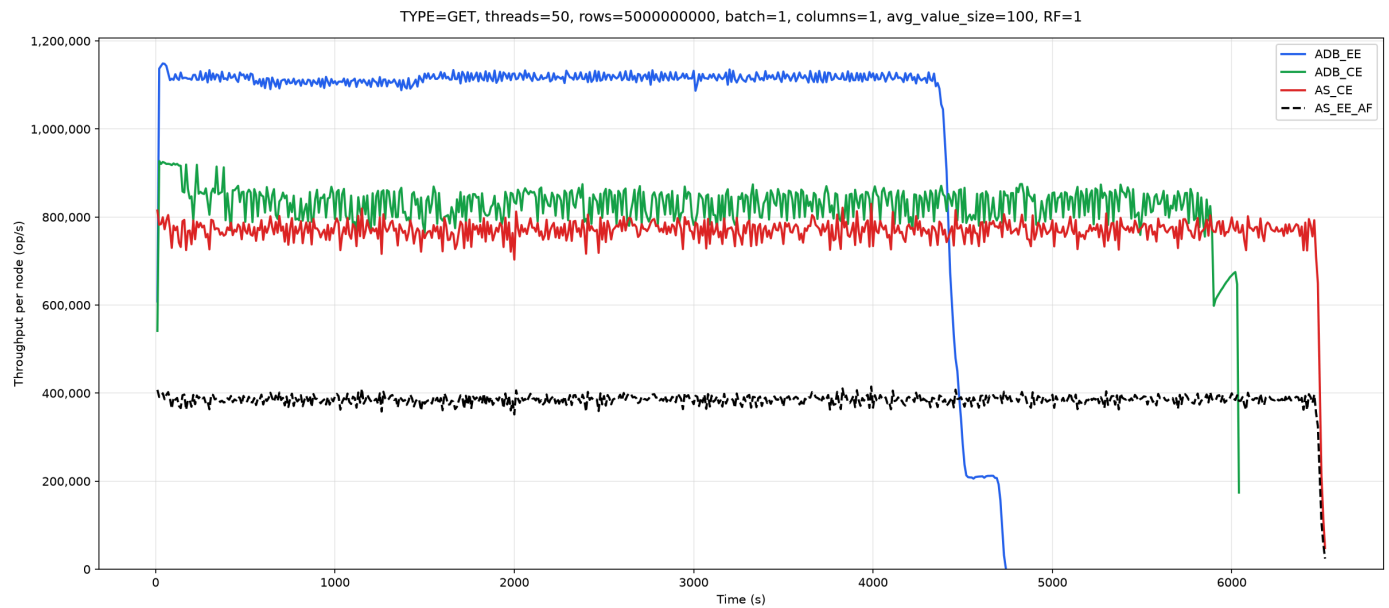


Figure 4

The higher read throughput of AtalantaDB EE in this test is attributed to Read Cache. This feature uses available memory to cache data. If memory is needed elsewhere, for example for growing indexes, the caches are cleared automatically so that the memory can be reused.

3.2 Single-record requests, ten columns: tests 04-06

The number of columns was increased from one to ten. Each column retained the same random value size of 50-150 bytes. The record count was reduced by a factor of ten to remain within the Aerospike CE capacity limit: 50 million records per load generator instead of 500 million.

Test 04. INSERT



Figure 5

Test 05. UPDATE, second pass

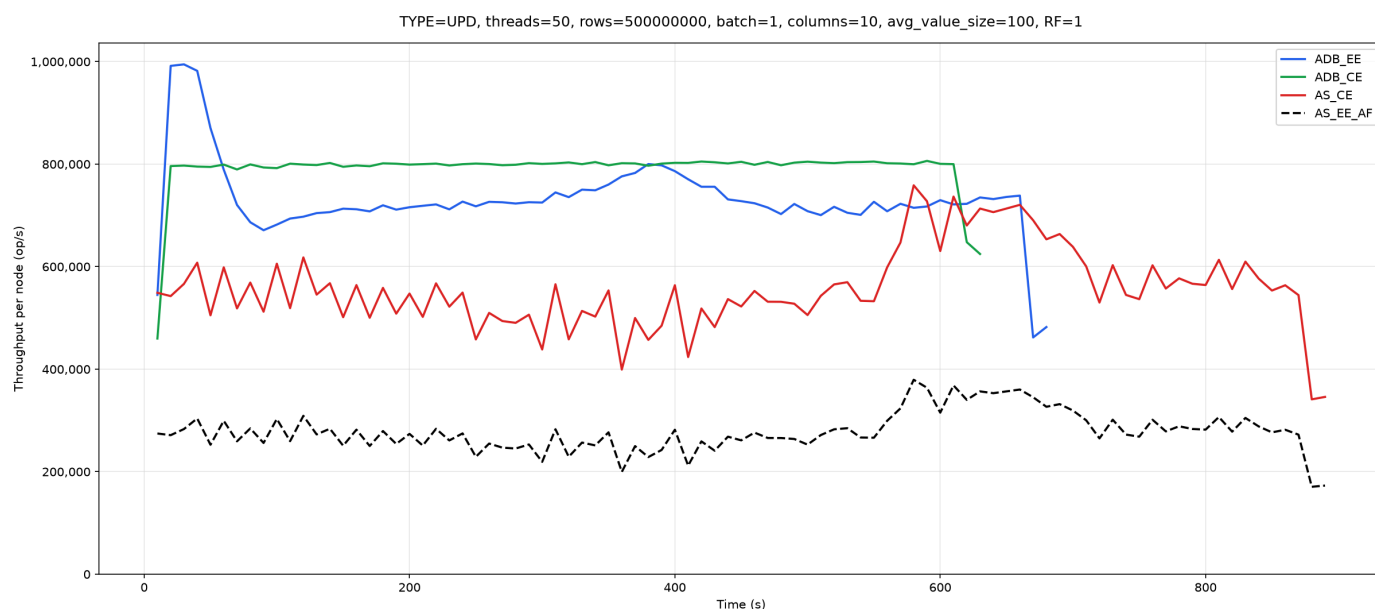


Figure 6

The difference between AtalantaDB CE and EE in the update test is explained by append update, which is enabled by default in EE. An update reads the existing record, merges it with the version supplied by the client, and appends the resulting record to the file. The previous version remains on disk as obsolete data. CE instead overwrites the previous version in place.

Append update improves throughput and reduces SSD wear compared with immediate in-place overwrites. Updated records are appended until the obsolete-data ratio reaches 50%, after which blocks are compacted. This allows multiple blocks with a high obsolete-data ratio to be rewritten in one pass, reducing the impact on SSD endurance over time.

The beginning of the chart shows updates while append space is still available and compaction is not required. Throughput decreases when compaction starts. A trade-off of append update is that larger data files take longer to load during startup index initialization.

Test 06. GET, after the update workload

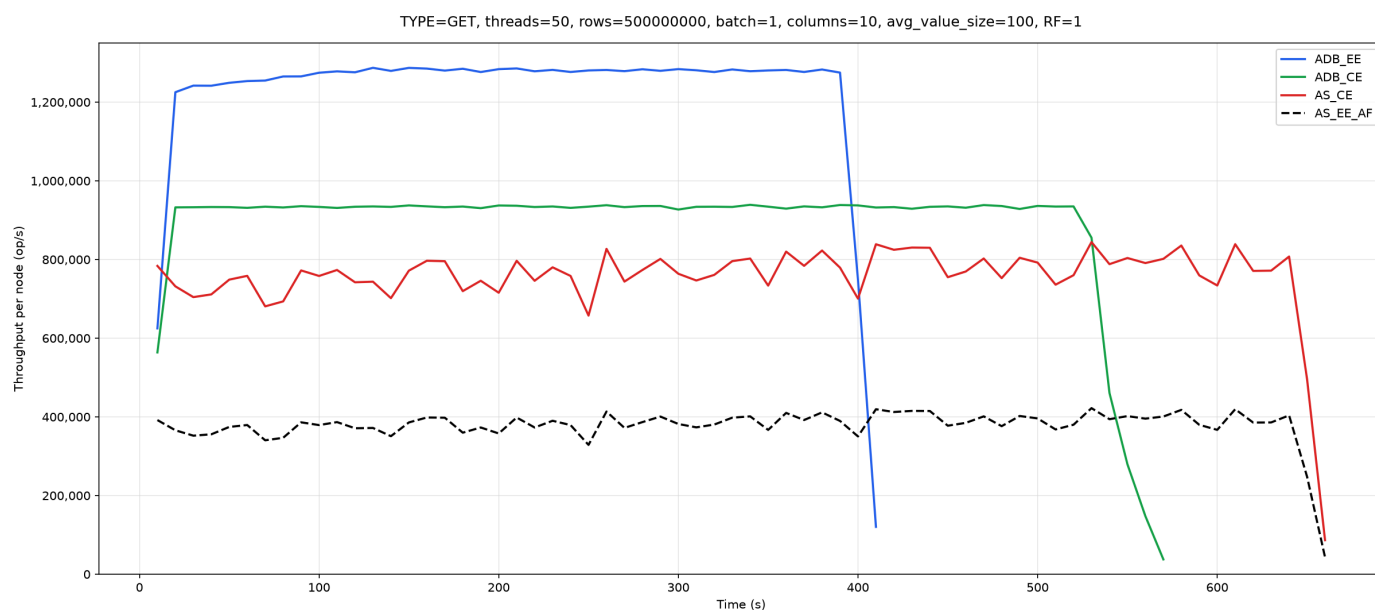


Figure 7

3.3 Batch requests, one column: tests 07-09

The first single-node workload was repeated in batch mode. Each load generator issued 500,000 batch operations of 1,000 records each, equivalent to 500 million records per generator. The total record count was unchanged.

Test 07. INSERT

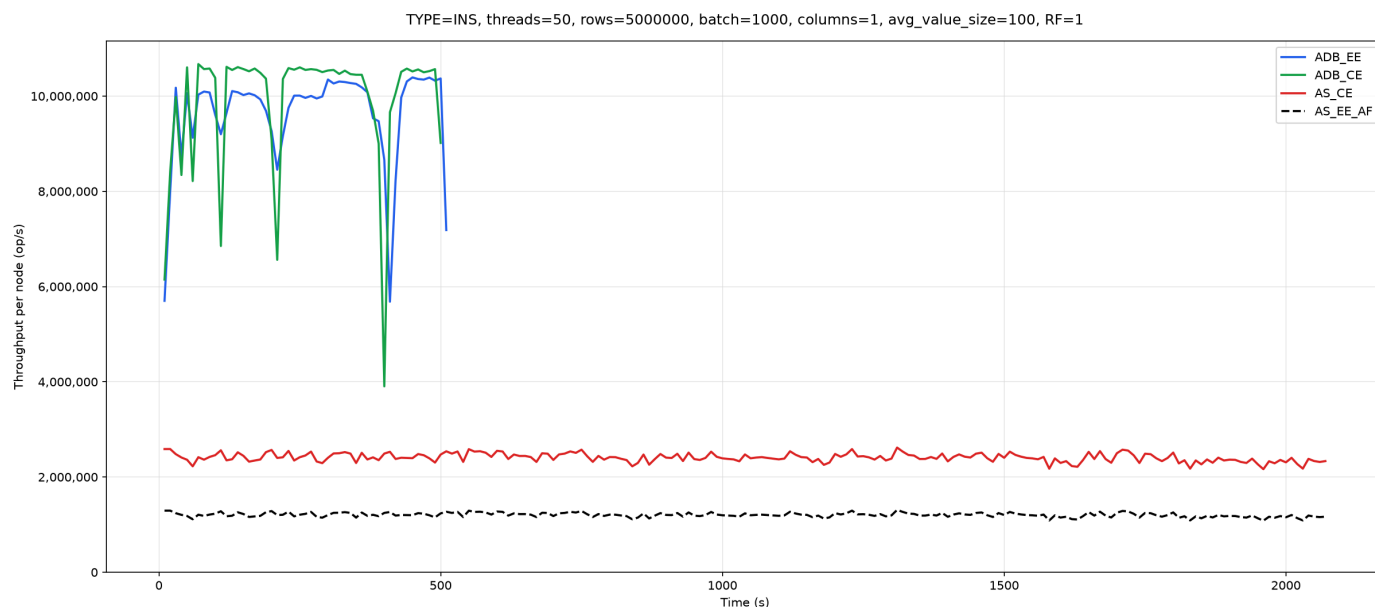


Figure 8

AtalantaDB benefits particularly from combining many individual record operations into one request.

Test 08. UPDATE, second pass

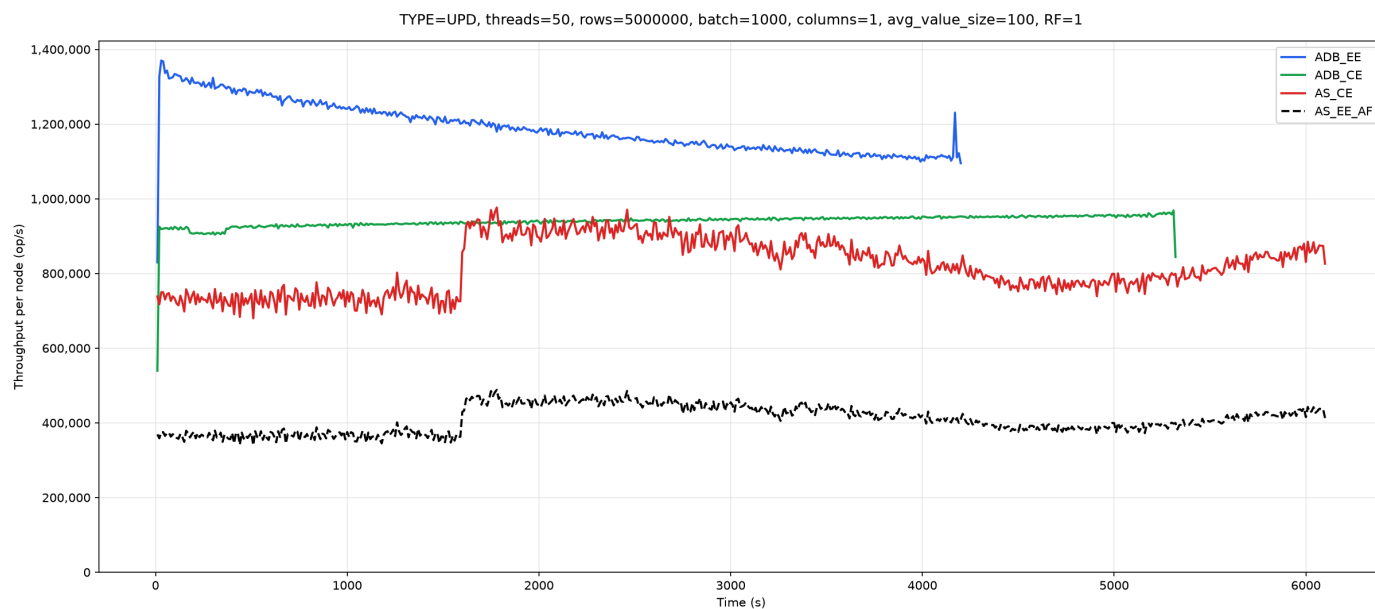


Figure 9

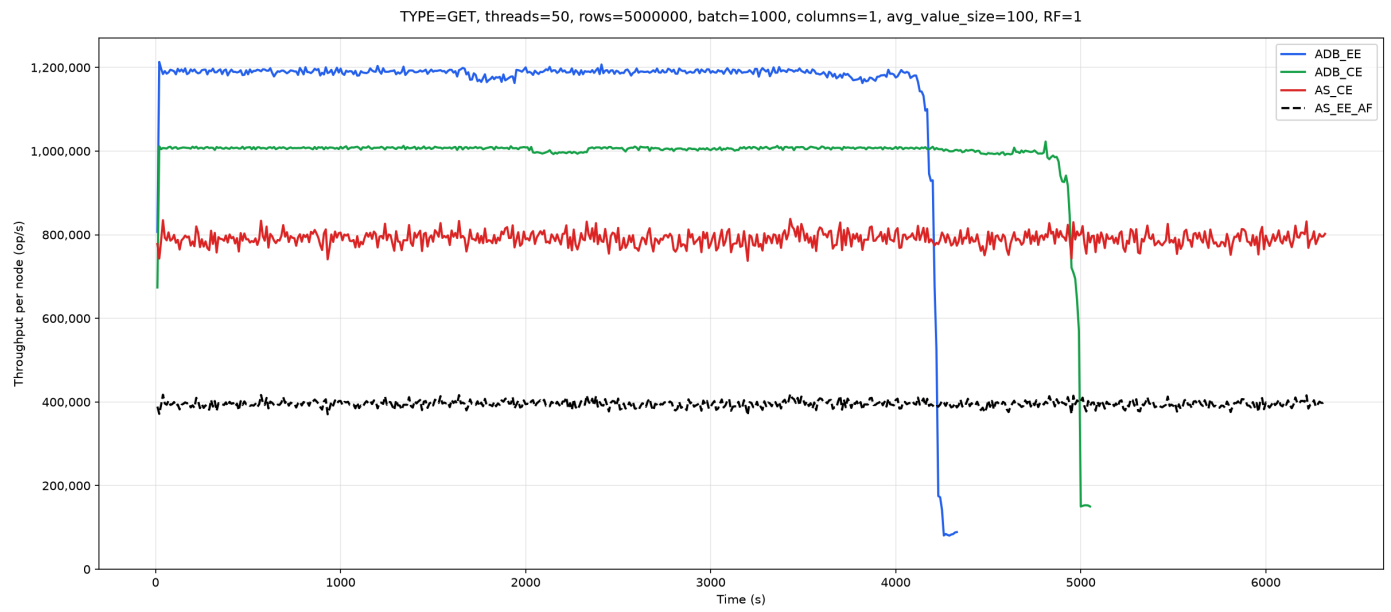
Test 09. GET

Figure 10

3.4 Batch requests, ten columns: tests 10-12

The number of columns was increased to ten, and the number of batch operations was reduced to 50,000 per load generator. At 1,000 records per batch, this corresponds to 50 million records per generator.

Test 10. INSERT

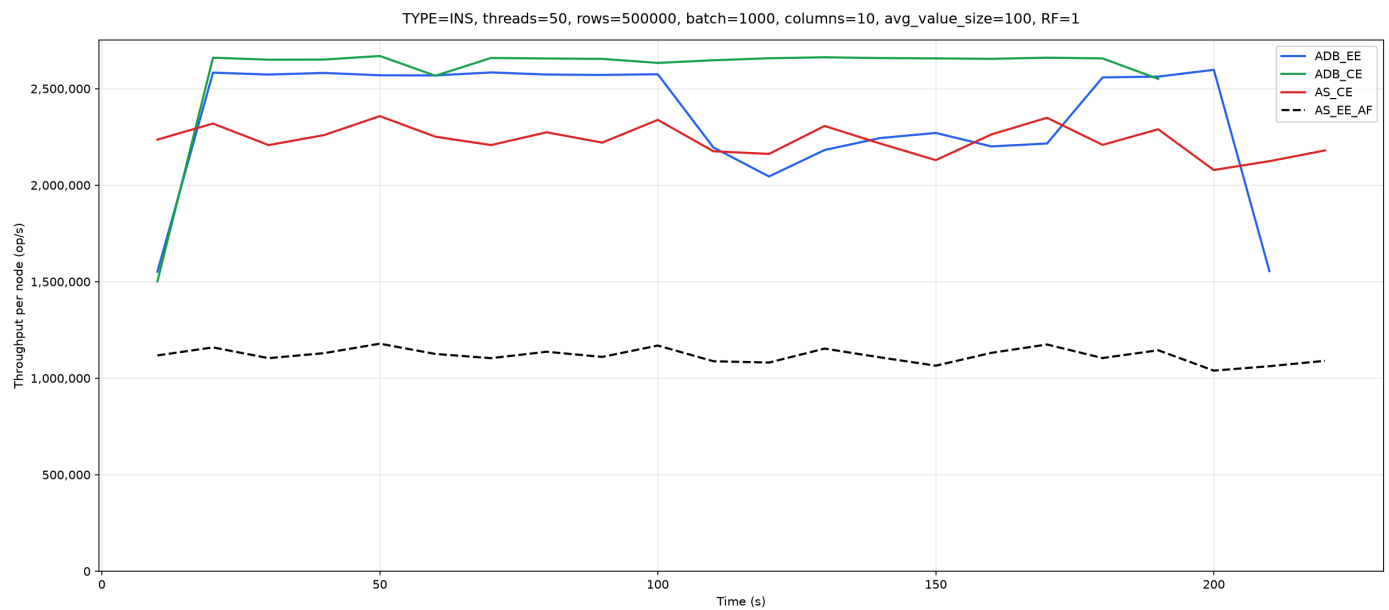


Figure 11

Test 11. UPDATE, second pass



Figure 12

Test 12. GET

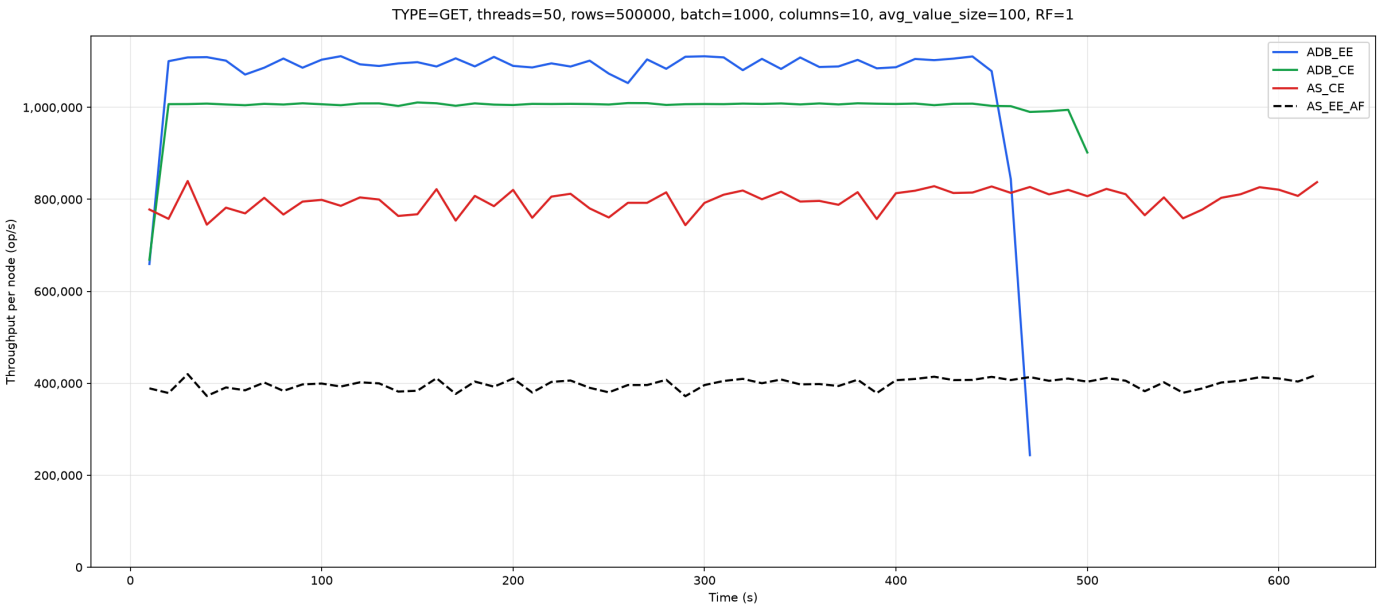


Figure 13

4. Three-node results: replication factor 2

4.1 Comparison with Cassandra: tests 13-15

These tests compare AtalantaDB CE and EE with Cassandra on three database instances with replication factor 2. The charts show single-record requests with ten columns per record.

Test 13. INSERT

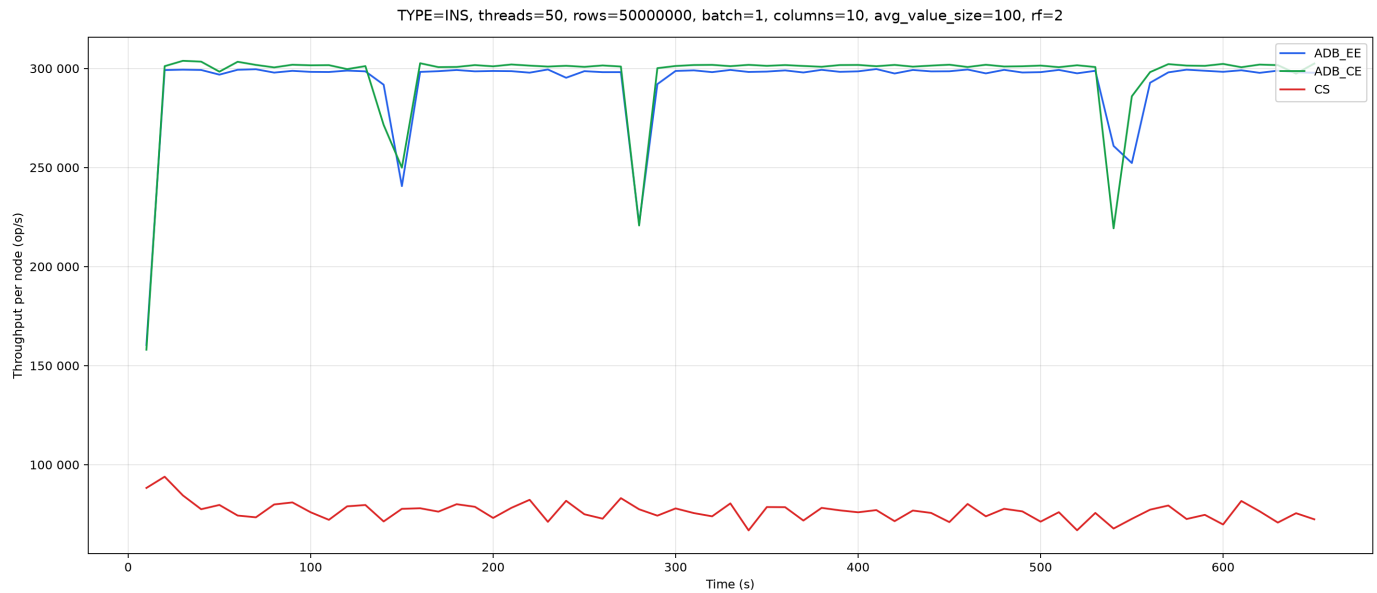


Figure 14

Test 14. UPDATE

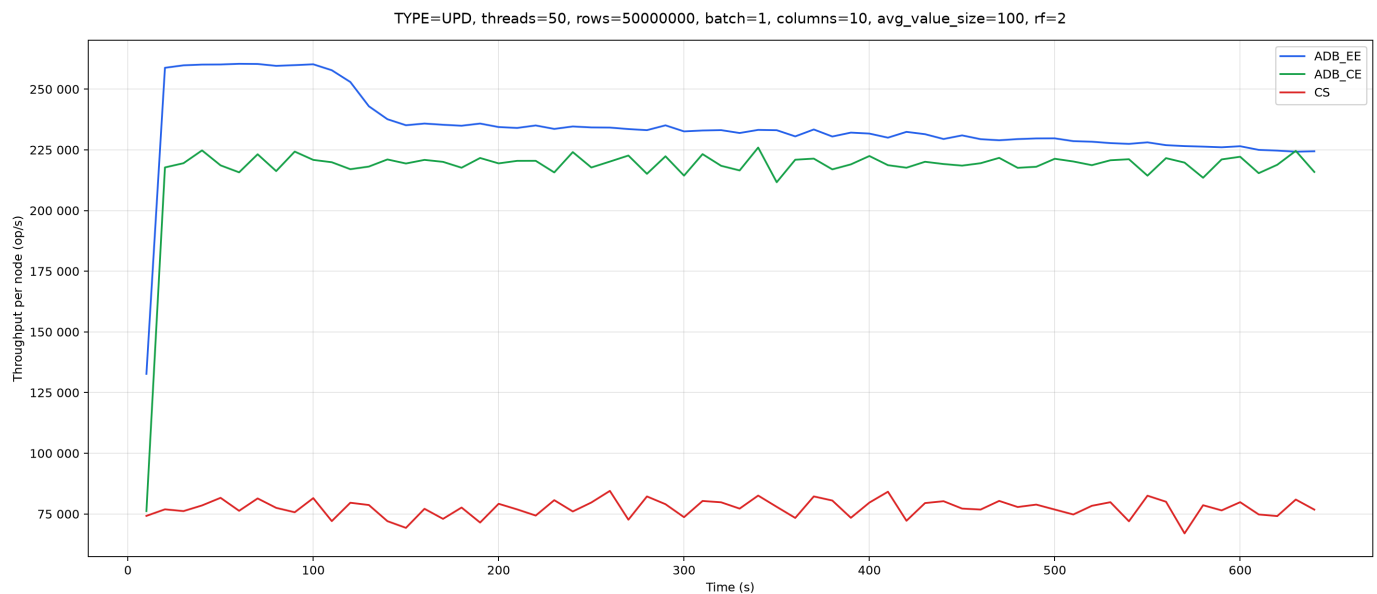


Figure 15

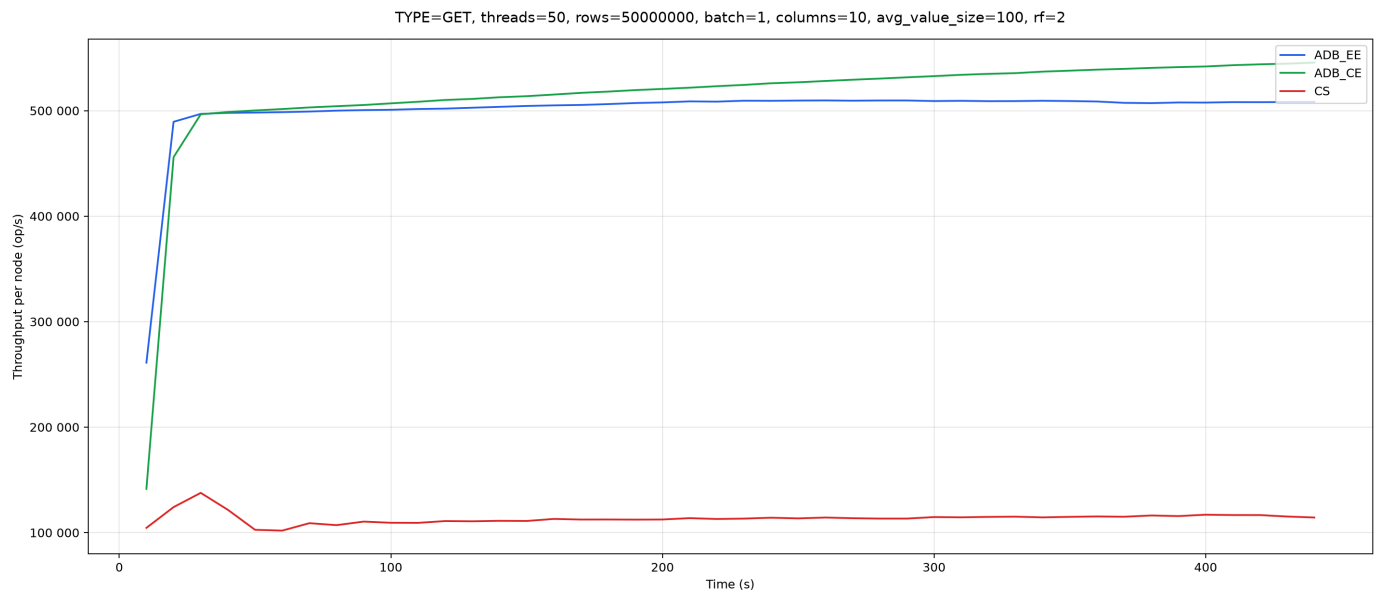
Test 15. GET

Figure 16

In the reported tests, AtalantaDB throughput was approximately three to five times that of Cassandra, with an approximate average advantage of four times. This comparison describes the workloads shown in these charts.

4.2 Comparison with Aerospike CE, single-record requests: tests 16-18

These tests used three database instances, replication factor 2, and ten columns per record. The workload count was 150 million single-record operations per load generator. Column values retained the 50-150 byte range.

Test 16. INSERT

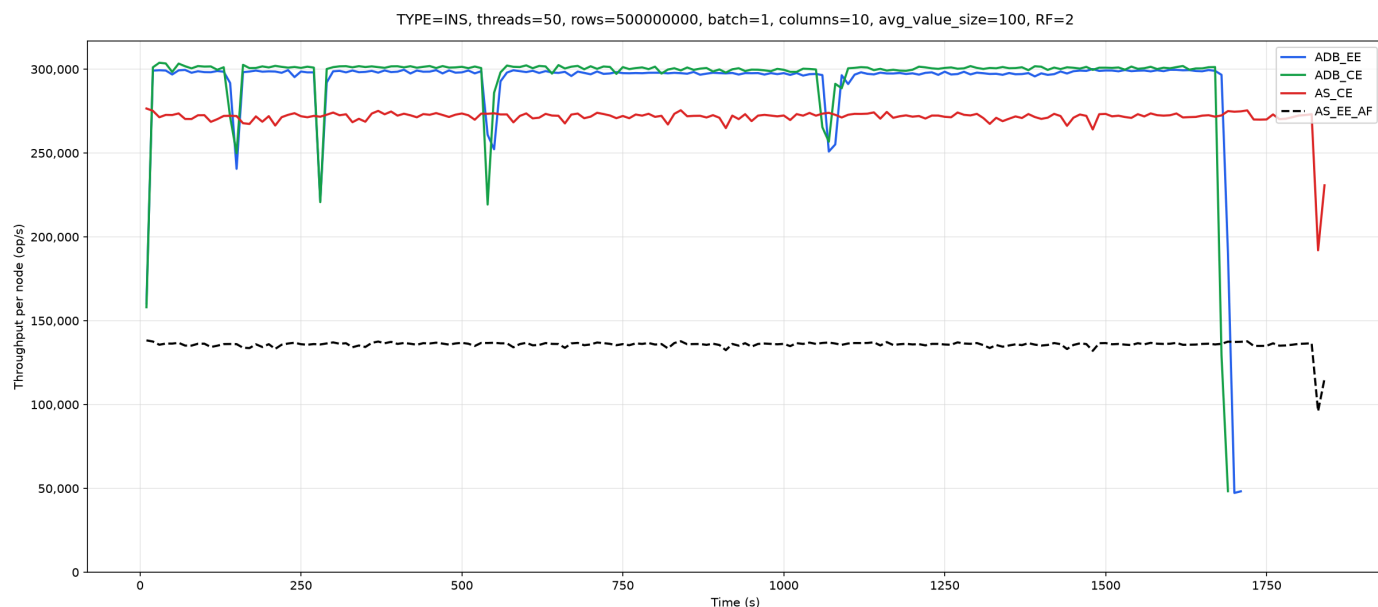


Figure 17

Test 17. UPDATE, second pass

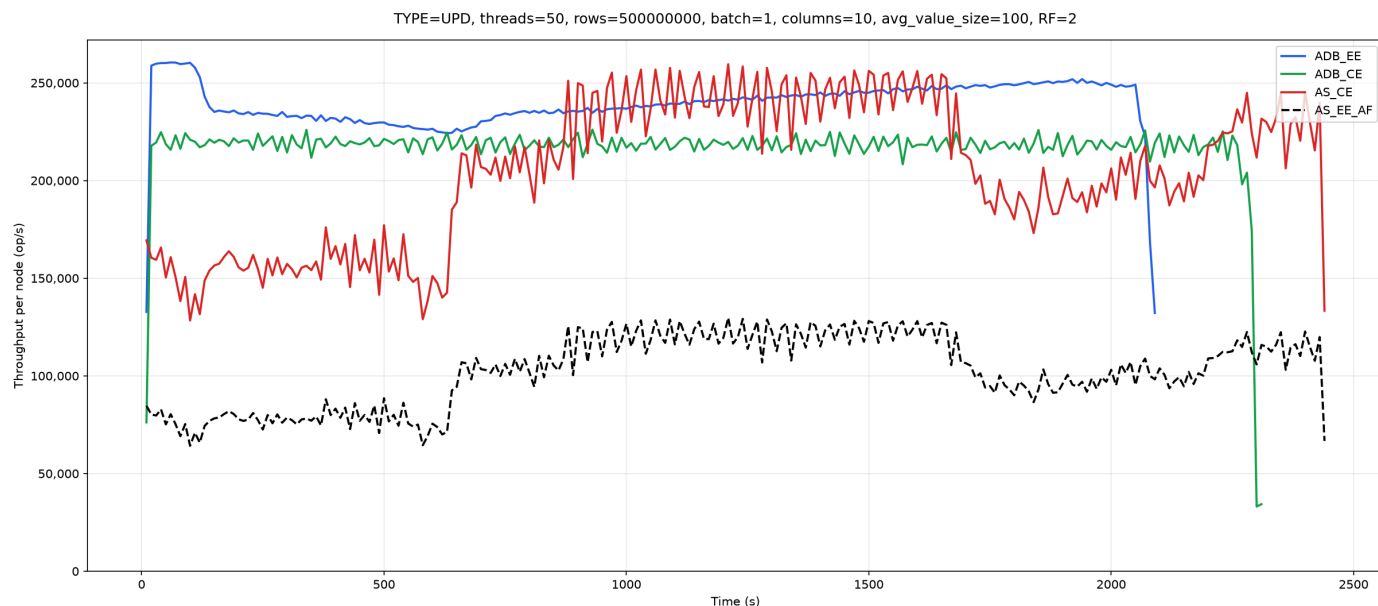


Figure 18

The decreases and subsequent increases in Aerospike throughput correspond to defragmentation starting and stopping.

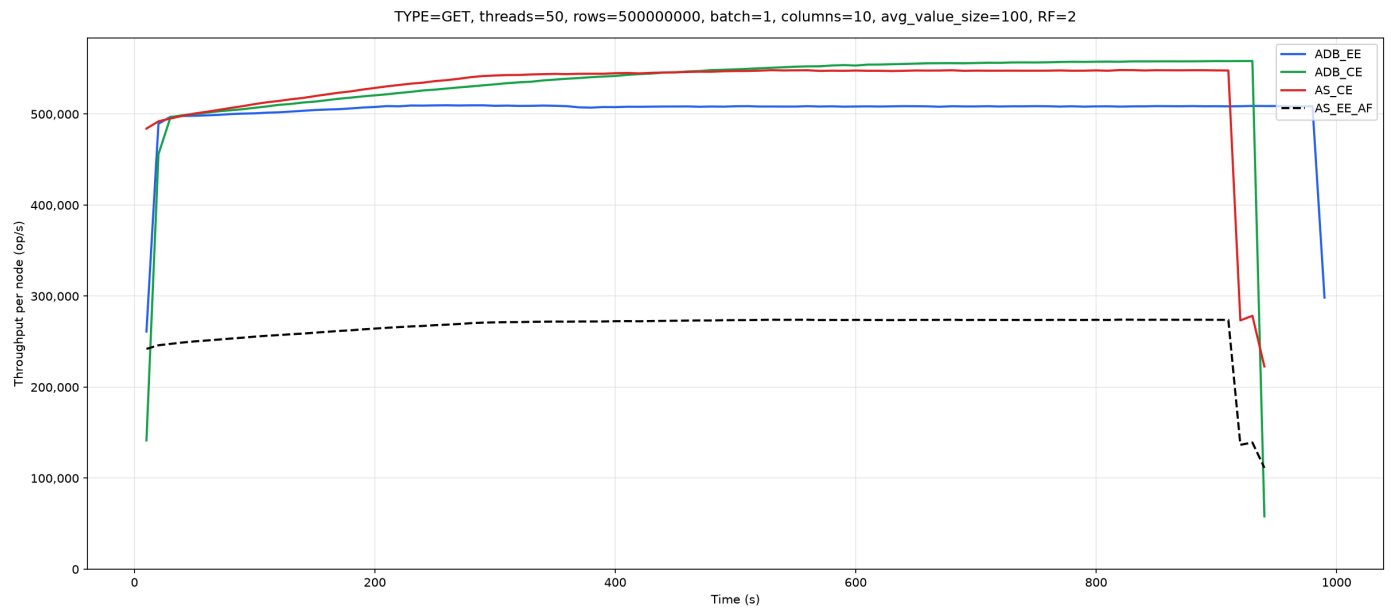
Test 18. GET

Figure 19

AtalantaDB CE achieved higher read throughput than EE in this test. The explanation is that append update produced larger files in EE, making caching less effective for this workload.

4.3 Comparison with Aerospike CE, batch requests: tests 19-21

The workload used 150,000 batch operations per load generator, with 1,000 records per batch and ten columns per record. This corresponds to 150 million records per generator. The deployment retained three database instances and replication factor 2.

Test 19. INSERT

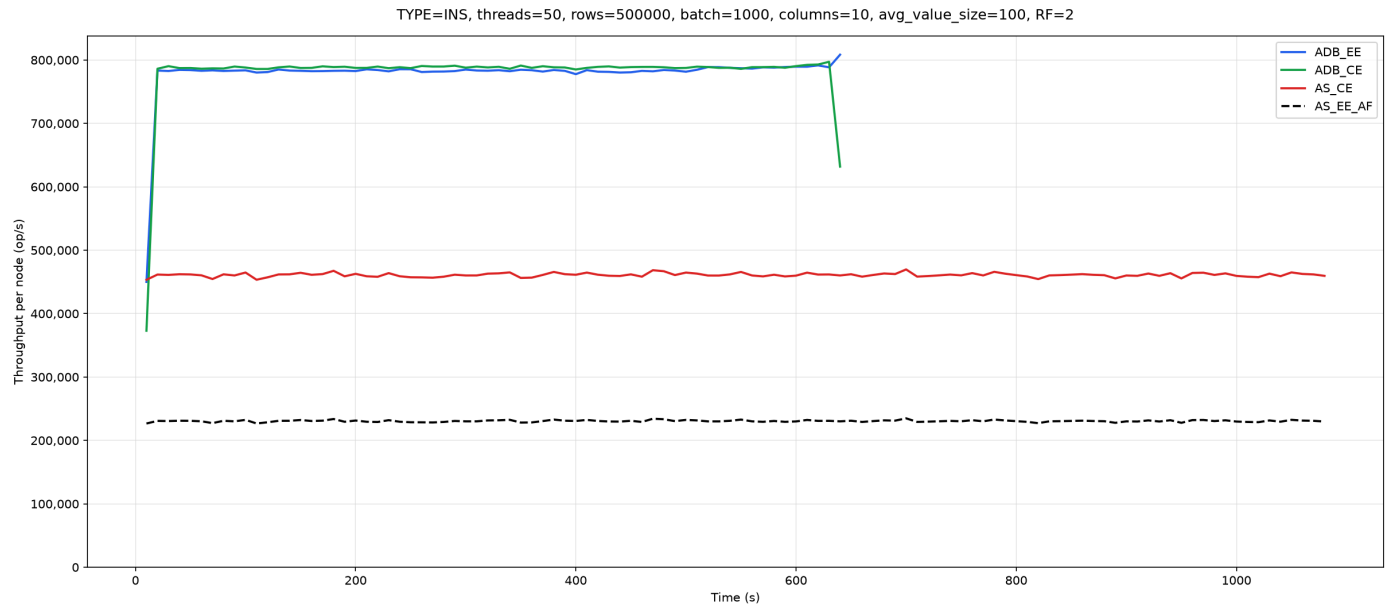


Figure 20

Test 20. UPDATE, second pass

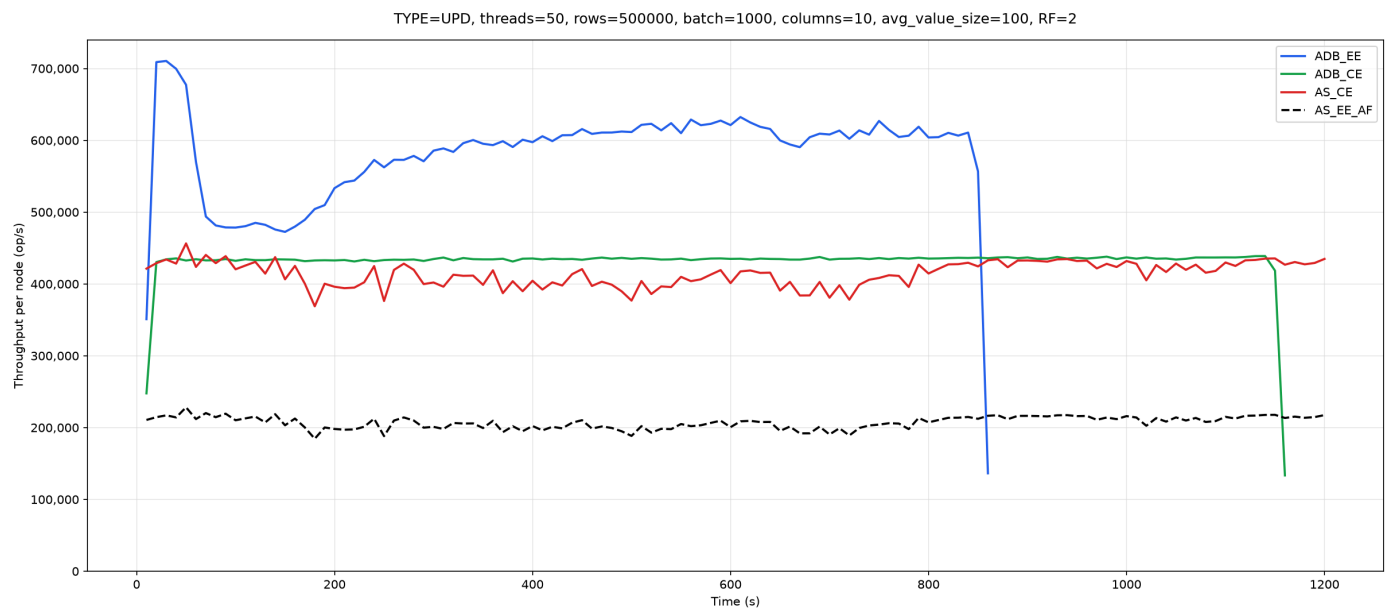


Figure 21

Test 21. GET

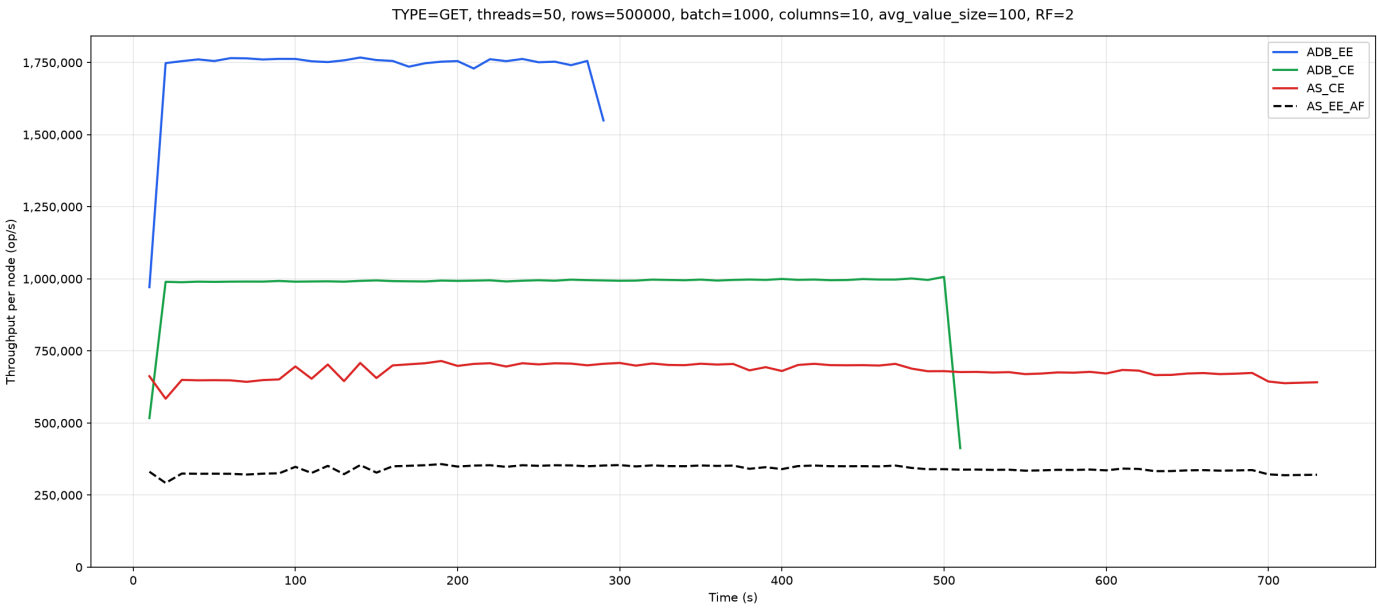


Figure 22

5. Latency

The following table presents the corresponding operation latency measurements for the compared configurations. The throughput and latency results should be considered together when assessing a workload.

Operation Latency Comparison						
Aerospike vs Atalanta (cnt=50M / bs=1 and cnt=50K / bs=1000)						
Operation	cnt = 50M / bs = 1			cnt = 50K / bs = 1000		
	Aerospike (cnt=50M / bs=1)	Atalanta (cnt=50M / bs=1)	Atalanta vs AS	Aerospike (cnt=50K / bs=1000)	Atalanta (cnt=50K / bs=1000)	Atalanta vs AS
 READ	626.50 µs	395.99 µs	-36.8%	624,362.66 µs	421,242.24 µs	-32.5%
 INSERT	412.96 µs	389.91 µs	-5.6%	218,361.03 µs	197,903.42 µs	-9.4%
 UPDATE	873.26 µs	666.65 µs	-23.7%	711,818.90 µs	461,034.60 µs	-35.2%
 Negative values in "Atalanta vs AS" indicate lower latency compared to Aerospike (i.e., better performance).						

Figure 23

6. Findings and supporting data

The results vary by workload and edition. In the six three-node comparisons with Aerospike CE, the largest throughput gains occur in batch workloads. For single-record GET, AtalantaDB CE is close to Aerospike CE, while AtalantaDB EE has slightly lower mean throughput. The relative performance of AtalantaDB CE and EE depends on the workload, available cache memory, and update mode.

The results apply to the hardware, datasets, concurrency, replication factors, and request patterns described in this report. They do not imply that AtalantaDB will outperform every other database in every workload. In particular, these tests used a sufficient number of regions to enable efficient parallel processing in AtalantaDB. The AS_EE_AF series remains a model and provides no directly measured evidence about Aerospike EE All-Flash performance.

Supporting data includes YCSB execution logs and operations-per-second measurements from one AtalantaDB instance. The archive is supplied as logs.zip alongside this report. The designated publication URL is: <https://atalantadb.com/benchmarks/nt1/logs.zip>

The throughput charts and latency table referenced above are supplied in the accompanying Pics directory. Their original filenames are retained to identify the corresponding workloads.

Appendix A. External reference material

Aerospike was selected as the primary comparator based on its published performance comparisons and the authors' own testing. The following table summarizes the external reports cited in the source article. These results are background material, not additional measurements from this test suite.

No	Report	Workload	Diff throughput
1	Aerospike vs Cassandra (2016)	YCSB 50/50, 400M × 1 KB, RF=2	≈14×
2	Aerospike vs ScyllaDB — Initial Comparison (2016)	YCSB 50/50, 400M × 1 KB	≈9×
3	Aerospike vs Cassandra Brief (2019)	YCSB balanced R/W, 2.65 TB unique	≈15.4×
4	Aerospike vs ScyllaDB Brief (2023)	YCSB 50/50, 4 TB	≈7.2×
5	Aerospike vs MongoDB Atlas (2024)	Player Protection, ~20/80 R/W, up to 10 TB	≈8.9×
6	Aerospike vs Redis (2025)	YCSB 70/30 and 100/0, 1/5/10 TB	+11%...+25%
7	Aerospike vs DynamoDB / DAX (2025)	70/30 and 100/0, 1/5/10 TB	≈26×...118×
8	Aerospike Enterprise vs Cassandra — benchANT (2026)	50R/40W/6U/4D, ~3.4 TB	≈3.49×
9	Aerospike vs ScyllaDB — multi-TB (2026)	70/30, 3 TB and 6 TB	≈2.42×...3.16×

Figure 24

[1] Aerospike vs Cassandra - Comparing NoSQL Databases

<https://aerospike.com/blog/comparing-nosql-databases-aerospike-vs-cassandra-benchmarking-for-real/>

[2] Aerospike vs ScyllaDB - An Initial Comparison

https://pages.aerospike.com/rs/229-XUE-318/images/Benchmark_Report__Aerospike-vs-ScyllaDB-Initial-Comparison.pdf

[3] Aerospike vs Cassandra - A Brief Comparison

https://pages.aerospike.com/rs/229-XUE-318/images/Benchmark_Report__Aerospike-vs-Cassandra-Brief-Comparison.pdf

[4] Aerospike vs ScyllaDB - A Brief Comparison

https://pages.aerospike.com/rs/229-XUE-318/images/Benchmark_Report__ScyllaDB.pdf

[5] A document use case comparison: Aerospike vs MongoDB Atlas

<https://aerospike.com/files/white-papers/aerospike-vs-mongodb-whitepaper.pdf>

[6] Database Performance Comparison: Aerospike and Redis

<https://aerospike.com/files/benchmarks/database-performance-comparison-aerospike-and-redis-benchmark.pdf>

[7] Database performance comparison: Aerospike and DynamoDB with DAX

<https://aerospike.com/files/benchmarks/aerospike-enterprise-vs-dynamodb-dax.pdf>

[8] Aerospike Enterprise vs Apache Cassandra Benchmark

<https://aerospike.com/files/benchmarks/aerospike-enterprise-vs-apache-cassandra-benchmark.pdf>

[9] Aerospike vs ScyllaDB: Performance and Cost at Multi-Terabyte Scale

<https://aerospike.com/files/benchmarks/aerospike-vs-scylladb-performance-and-cost-at-multi-terabyte-scale.pdf>