MCKNIGHT CONSULTING GROUP

Agentic Data Management at Scale: Benchmarking Vector and Hybrid Search Across Postgres and Specialized Databases

Benchmarking the real-world speed, operational friction, and true financial cost of Multi-Model Operational Databases with Native Vector Search.

Comparison of Amazon Aurora PostgreSQL, Crunchy Bridge (Snowflake Postgres), Databricks, EDB Postgres AI, MongoDB and Open Source PostgreSQL

McKnight Consulting Group

William McKnight and Jake Dolezal

Sponsored by EnterpriseDB

August 2026

Table of Contents

© McKnight Consulting Group 2026

Executive Summary

Vector databases have officially graduated from the experimental sandbox. As enterprises scale agents into full production, the vector database layer has solidified into mission-critical infrastructure. At this stage of maturity, choosing the wrong architecture isn't just a minor technical hitch—it is a costly detour that competitive organizations simply cannot afford.

This report tests a practical proposition: enterprises may not need a separate vector database at all, because the Postgres they already run can serve production AI better than the specialized alternatives. McKnight Consulting Group benchmarked EDB Postgres AI (EDB PG AI) against Amazon Aurora PostgreSQL, Crunchy Bridge, open-source PostgreSQL, MongoDB Atlas, and Databricks on equivalent enterprise hardware and compute.

The evaluation covers the three workloads that define a production agent system, and each maps to a business outcome:

- **Indexing and core vector search:** How fast and how accurately the system answers as data grows, which sets the responsiveness and answer quality users experience.
- **Hybrid and filtered search:** How well the system combines semantic search with structured filters, which determines whether agents can query real business data (patient records, entitlements, transactions) alongside vectors.
- **End-to-end agentic retrieval:** How the system holds up under a live, concurrent agent loop, which drives both latency and the token cost of every LLM call.

To keep the tests grounded in a demanding real-world pattern, all three run against a fully synthetic clinical corpus, modeling a healthcare Retrieval Augmented Generation (RAG) scenario with strict accuracy and data-freshness requirements.

Two findings stand out for buyers. Postgres clearly outperformed the specialized vector and lakehouse platforms, and among the Postgres providers, EDB PG AI led the field.

Across all evaluated benchmarks, EDB PG AI consistently outperformed its enterprise competitors, establishing a distinct technical advantage in latency, recall, and token efficiency. In core vector indexing, EDB PG AI demonstrated exceptional scalability. At 10 million vectors, it led the field with a median p50 latency of 11ms, a p99 tail latency of 21ms, and a top-tier Recall@10 of 0.911. When scaled to 50 million vectors, EDB PG AI broke away from other Postgres variants, maintaining a highly responsive p50 latency of 50ms, a p99 latency of 119ms, and the highest overall accuracy with a Recall@10 of 0.884.

© McKnight Consulting Group 2026

This performance edge extended into hybrid retrieval and filtered search, where EDB PG AI secured the fastest query response times, including an impressive 12ms execution time for dense filtering at low selectivity. It also delivered the highest accuracy scores among Postgres engines, peaking at a recall of 0.907 for dense hybrid queries under high selectivity constraints, proving the effectiveness of its tuned hierarchical navigable small world (HNSW) post-filtering.

Finally, in end-to-end agentic RAG workflows, EDB PG AI delivered the strongest comprehensive profile. It achieved the fastest agent loop execution with a p50 latency of 27ms and a p99 latency of 40ms, while leading in context efficiency by minimizing semantic and hybrid wasted token rates to 8.0% and 17.0%, respectively. Under concurrent workloads, EDB PG AI sustained a high write throughput of 85.5 vectors/second, an industry-leading read throughput of 360.4 loops/second, and near-instant data visibility with a mean write-to-read freshness of just 12ms.

© McKnight Consulting Group 2026

Tests Conducted

These three testing areas were selected to comprehensively evaluate pgvector by isolating its raw scalability, its behavior under realistic database constraints, and its real-world application performance.

The Core Vector/Index Test measures pure algorithmic efficiency and memory scaling from 10M to 50M vectors, while the Hybrid Retrieval & Filtered Search forces the query planner to handle varying levels of metadata selectivity, replicating how enterprises query relational data alongside vectors. Finally, the Agent/RAG Workflow uses a clinical scenario to move beyond isolated database metrics, validating end-to-end latency, fixed embedding models, and chunking standards in a practical production environment.

- **Test 1 – Core Vector/Index Performance**
- **Test 2 – Hybrid Retrieval & Filtered Search**
- **Test 3 – Agent/RAG Workflow**

The following parameters are fixed for all three tests:

Embedding Model	OpenAI text-embedding-3-small (1536 dimensions)
Dataset Domain	Clinical/healthcare fully synthetic corpus
Chunk Size	512 tokens, 64-token overlap

Table 1: Test Parameters

© McKnight Consulting Group 2026

Platforms Tested

The platforms selected are a mix of open-source baselines, core cloud-managed PostgreSQL alternatives, and heavyweight enterprise data platforms. Utilizing standard PostgreSQL OSS as a control ensures a pure baseline for standard pgvector efficiency, while including major cloud players like Amazon Aurora and Crunchy Bridge accurately reflects the real-world choices architects face when deploying managed PostgreSQL in production.

Amazon RDS Aurora Postgres with pgvector (HNSW)
Crunchy Bridge with pgvector (HNSW)
Databricks – Delta Sync vector indexing and Mosaic AI vector search
EDB PG AI with pgvector (HNSW)
MongoDB Atlas – Vector Search on Atlas
Open-source PostgreSQL with pgvector (HNSW)

Table 2: Tested Platforms

Crucially, by extending the scope to include document-store and lakehouse giants like MongoDB Atlas and Databricks (utilizing Delta Sync and Mosaic AI), the benchmark moves beyond a narrow, single-engine ecosystem comparison to evaluate pgvector against completely different database paradigms. Normalizing these disparate architectures around a standardized 512 GB memory tier and 64 to 72 vCPU[1] compute profile provides an exceptionally level playing field, ensuring the final report delivers an apples-to-apples performance analysis based on equivalent, enterprise-grade resource investments.

The inclusion of (HNSW) for some explicitly indicates the exact indexing algorithm chosen within the pgvector ecosystem, distinguishing it from alternative index types like IVFFlat. For the PostgreSQL-based platforms—including Amazon Aurora, Crunchy Bridge, EDB PG AI, and Open-source PostgreSQL—specifying HNSW is necessary because pgvector requires the user to explicitly select and configure the underlying graph structure. Conversely, Databricks (utilizing Mosaic AI) relies on its own proprietary, native vector search engines rather than HNSW.

[1] MongoDB Atlas was allocated 72 vCPUs as opposed to 64 for the others because its managed cloud tiers bundle memory and compute together, meaning the 512 GB RAM target was only available on a 72-vCPU hardware configuration.

© McKnight Consulting Group 2026

Competitor	SKU	Hardware	Vector Index
Amazon Aurora	db.r8g.16xlarge I/O-Optimized	64 vCPU, 512 GB	HNSW
Crunchy Bridge	Memory-512 1TB Storage	64 vCPU, 512 GB	HNSW
Databricks	r8g.16xlarge Vector endpoint	64 vCPU, 512 GB	Vector Search
EDB PG AI	r8gd.16xlarge	64 vCPU, 512 GB	HNSW
MongoDB Atlas	M400_NVMe	72 vCPU, 512 GB	HNSW
PostgreSQL OSS	r8gd.16xlarge	64 vCPU, 512 GB	HNSW

Table 3: Infrastructure Specifications

© McKnight Consulting Group 2026

Test Results

Test 1 – Core Vector / Index Performance

Objective: Measure raw indexing velocity and query throughput as dataset size grows

Metrics:

- P50 / P95 / P99 query latency (ms). Percentiles indicate the percentage of data points that fall below a specific threshold, showing how a single performance metric compares to the rest of the dataset. P50 captures standard, steady-state workloads, making it the ideal metric for calculating baseline resource consumption and cost projections during capacity planning. P95 serves as the optimal baseline for standard Service Level Objectives (SLOs), ensuring an excellent user experience for the vast majority while filtering out rare edge cases. P99 targets core infrastructure and critical workflows, shielding high-volume systems and high-value users from impactful, tail-end latency spikes.
- Recall@10. This is the fraction of the true top-10 nearest neighbors (by exact cosine similarity) that an approximate search returns in its top-10 results.

Test Parameters:

Corpus size	10M records, 50M records
Concurrency	10 concurrent query threads
Top-k	10

Table 4: Test 1 Parameters

	Aurora	Crunchy	Databricks	EDB PG AI	MongoDB	OSS
Query p50 (ms)	12	13	1,893	**11**	149	13
Query p95 (ms)	19	19	2,417	**17**	646	20
Query p99 (ms)	27	26	2,894	**21**	2,278	25
Recall@10	0.899	0.908	0.779	**0.911**	0.722	0.903

Table 5: 10M Vectors Query Performance

	Aurora	Crunchy	Databricks	EDB PG AI	MongoDB	OSS
Query p50 (ms)	94.9	98	4,023	**50**	1,083	78
Query p95 (ms)	167.8	175	7,113	**94**	2,875	127
Query p99 (ms)	205.2	216	8,637	**119**	4,387	288
Recall@10	0.871	0.868	0.738	**0.884**	0.693	0.873

Table 6: 50M Vectors Query Performance

© McKnight Consulting Group 2026

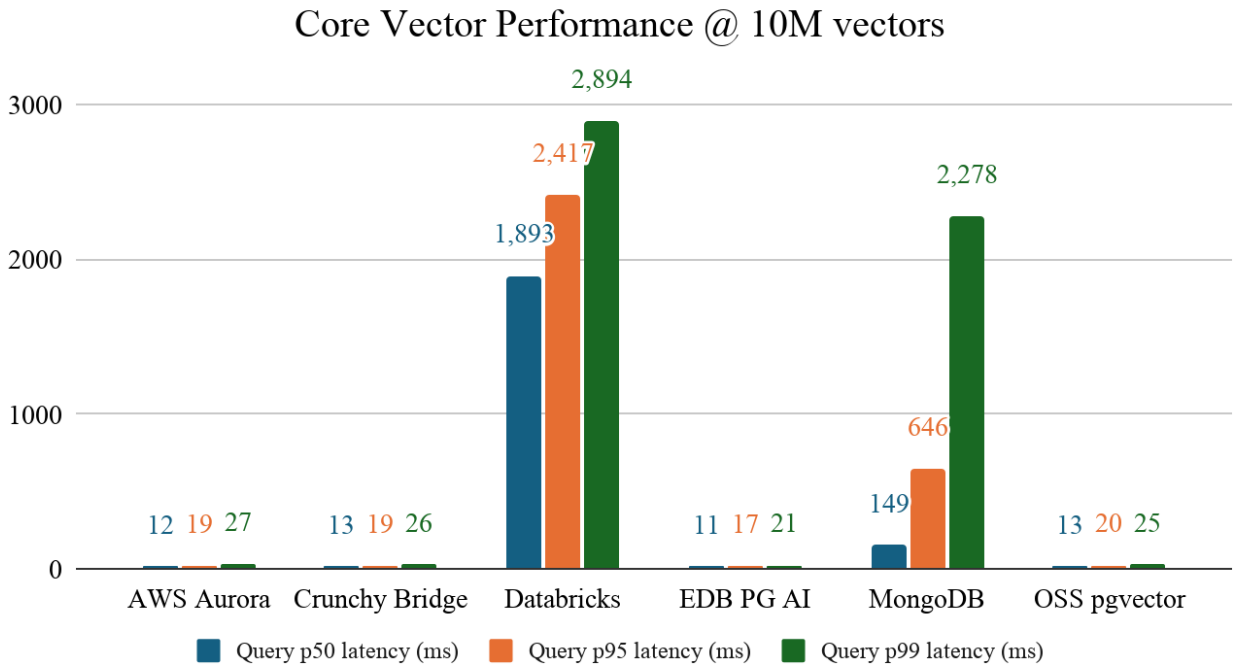

Figure 1: Core Vector Performance at 10M Vectors

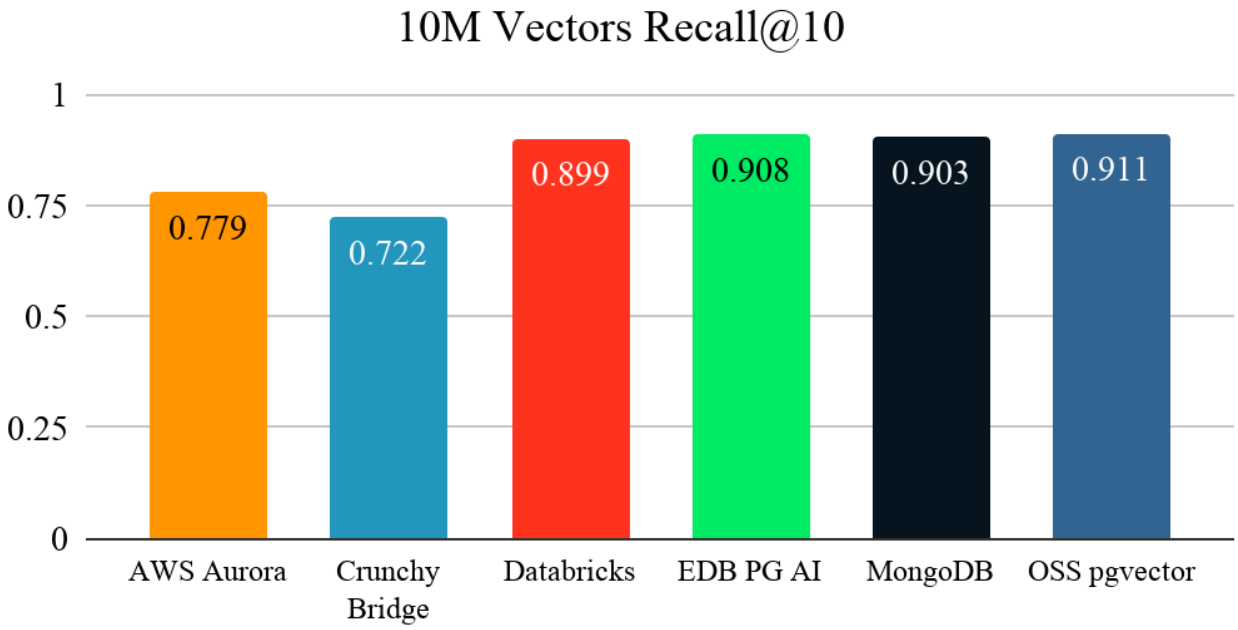

Figure 2: 10M Vectors Recall@10

© McKnight Consulting Group 2026

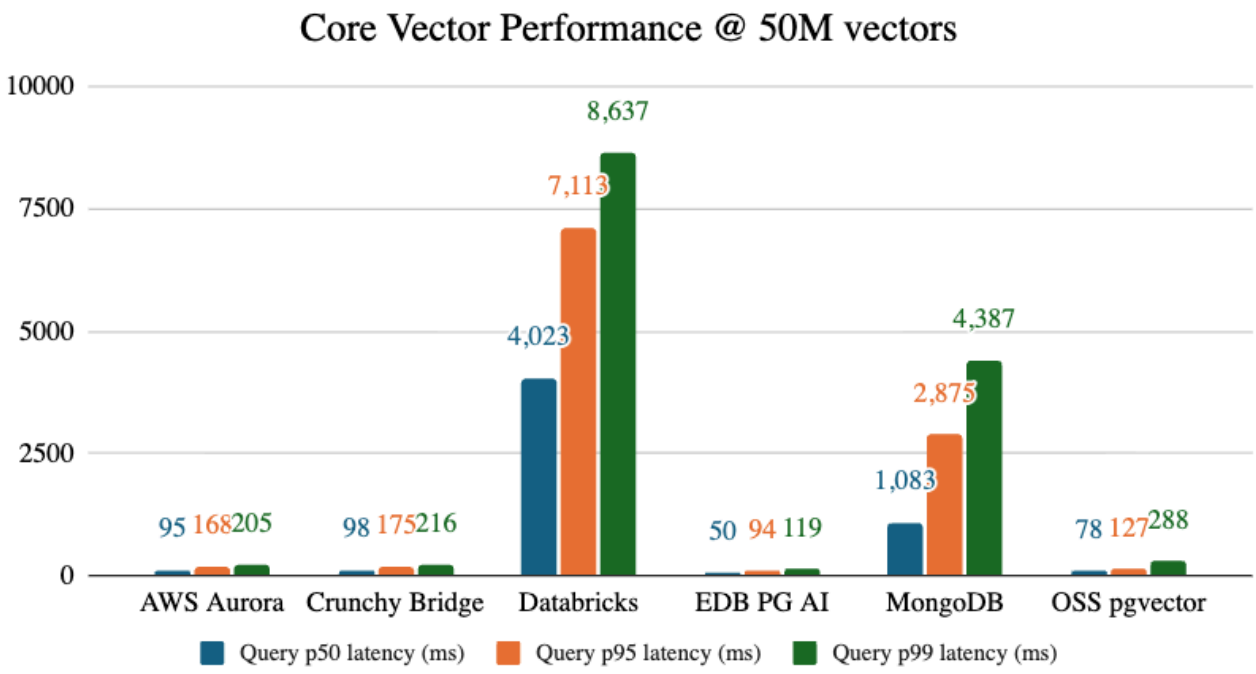

Figure 3: Core Vector Performance at 50M Vectors

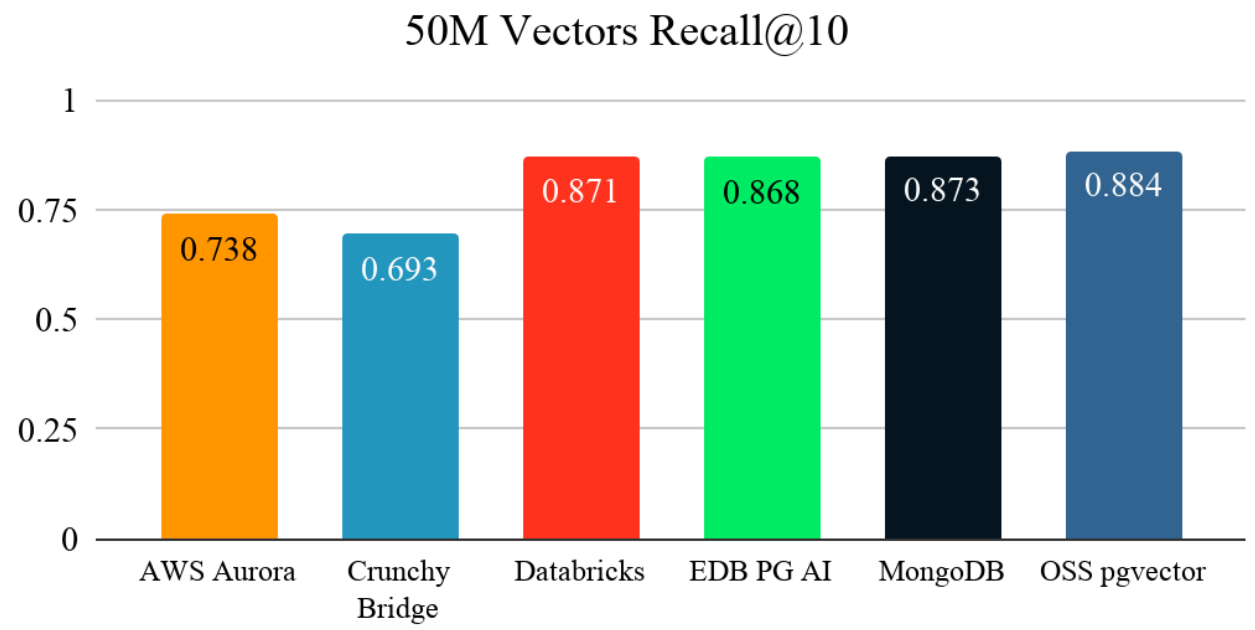

Figure 4: 50M Vectors Recall at 10

© McKnight Consulting Group 2026

EDB PG AI leads on ANN query latency at both scales and achieves the highest recall across all platforms at both 10M and 50M.

The performance benchmark data demonstrates a definitive latency and accuracy advantage for the PostgreSQL-based ecosystem when running HNSW vector searches under a 10-thread concurrency load. At the 10M vector tier, all four Postgres variants cluster tightly, with EDB PG AI leading the field at a typical p50 latency of 11ms and an exceptional p99 tail latency of 21ms while capturing the highest accuracy at 0.911 Recall at 10. In stark contrast, non-Postgres platforms exhibit a severe performance penalty. MongoDB Atlas's median response time is over 13 times slower than EDB PG AI's, and Databricks' Mosaic AI vector search proves highly sluggish with a p50 of 1,893ms, with both alternative platforms sacrificing recall accuracy below acceptable enterprise baselines.

When scaling the workload fivefold to 50M vectors, the architectural advantages of EDB PG AI become even more pronounced as it cleanly breaks away from the other Postgres competitors. While standard open-source PostgreSQL, Amazon Aurora, and Crunchy Bridge see their median latencies climb to between 78ms and 98ms, EDB PG AI maintains a highly responsive p50 of just 50ms and keeps its worst-case p99 tail latency to an impressive 119ms, beating open-source Postgres's p95 tier. Meanwhile, MongoDB Atlas and Databricks experience massive degradation under this scale, bottlenecking into multi-second latency profiles with p99 metrics of 4.4 and 8.6 seconds, respectively, that render them impractical for real-world, interactive AI applications.

The advantage is configuration, not hardware. EDB PG AI and open-source PostgreSQL ran on identical instances. EDB's enterprise defaults set shared_buffers and effective_cache_size far more aggressively than stock Postgres, so more of the HNSW graph stays resident in the buffer pool under concurrent load. That avoids the cold page reads that spike HNSW tail latency, which is why EDB PG AI holds a 119ms p99 where open-source Postgres climbs to 288ms.

Test 2 – Hybrid Retrieval & Filtered Search

Objective: Evaluate whether combining dense semantic search with structured filters produces measurably better retrieval accuracy than semantic-only methods

Metrics:

- P50 query latency per search mode and selectivity band
- Recall@10 – the fraction of the true top-10 nearest neighbors (by exact cosine similarity) that an approximate search returns in its top-10 results

© McKnight Consulting Group 2026

Test Parameters:

Corpus size	10M records (fixed)
Filter selectivity	High (~3.3% of corpus), Medium (~16.7%), Low (~55.6%)
Search mode	Structured-filter-only, Dense ANN, Hybrid (dense + sparse),

Table 7: Hybrid Retrieval & Filtered Search Test Parameters

Query Type	Selectivity	Aurora	Crunchy	Databricks	EDB PG AI	MongoDB	OSS
Structured filter only	High	356	555	7,378	157	**3**	378
	Med	372	382	4,319	329	**4**	331
	Low	453	322	4,222	299	**4**	304
Dense ANN + filter	High	18	19	567	**18**	905	21
	Med	17	15	617	**13**	560	17
	Low	14	14	610	**12**	154	16
Hybrid + filter	High	15	17	3,283	**15**	711	19
	Med	14	20	3,290	**15**	813	19
	Low	14	19	3,303	**14**	258	20

Table 8: P50 Query Latency

Query Type	Selectivity	Aurora	Crunchy	Databricks	EDB PG AI	MongoDB	OSS
Dense ANN + filter	High	0.786	0.788	0.353	**0.792**	0.302	0.783
	Med	0.891	0.890	0.420	0.883	**0.991**	0.880
	Low	0.711	0.721	0.350	**0.723**	0.211	0.702
Hybrid + filter	High	0.784	0.791	0.352	0.795	0.274	**0.803**
	Med	0.882	0.901	0.828	0.907	**0.993**	0.893
	Low	0.763	0.744	0.749	**0.778**	0.748	0.759

Table 9: Recall at 10

© McKnight Consulting Group 2026

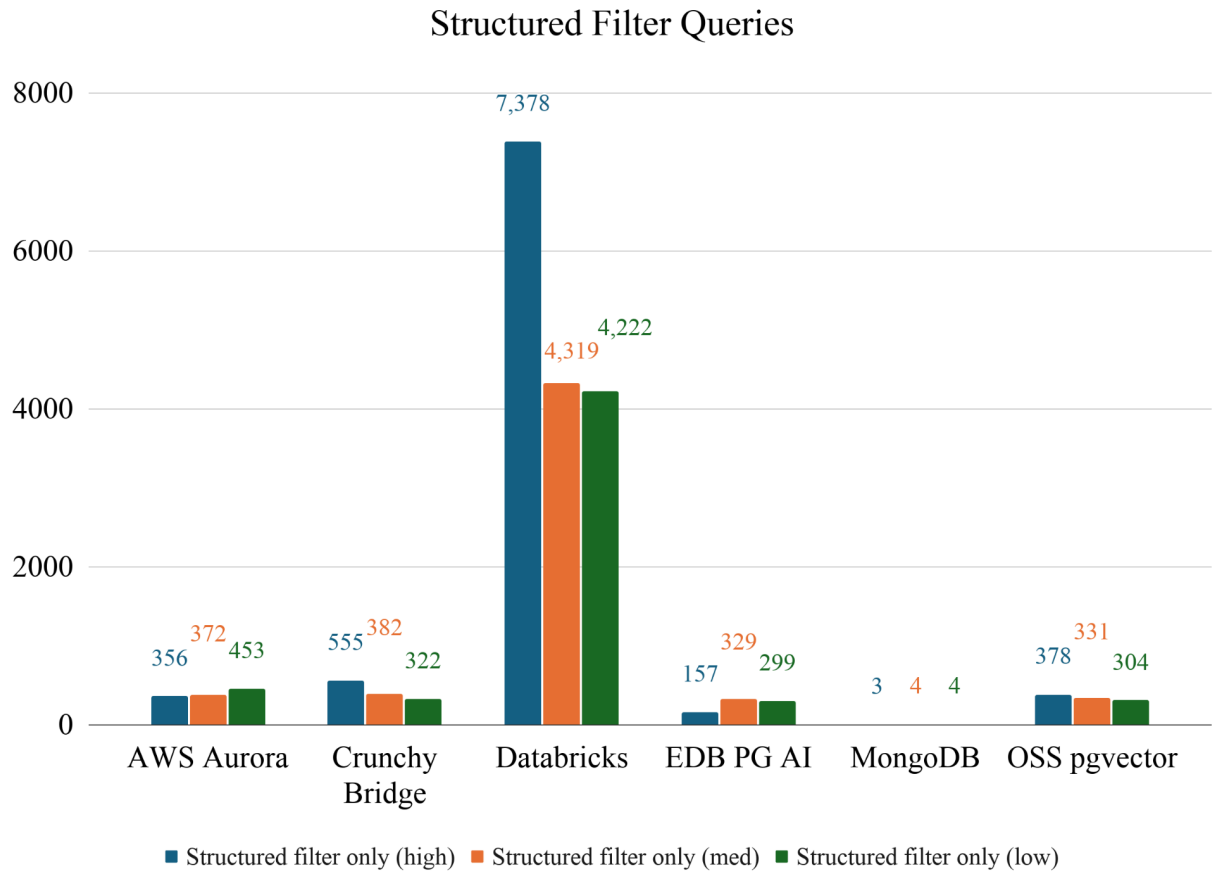

Figure 5: Structured Filter Queries

© McKnight Consulting Group 2026

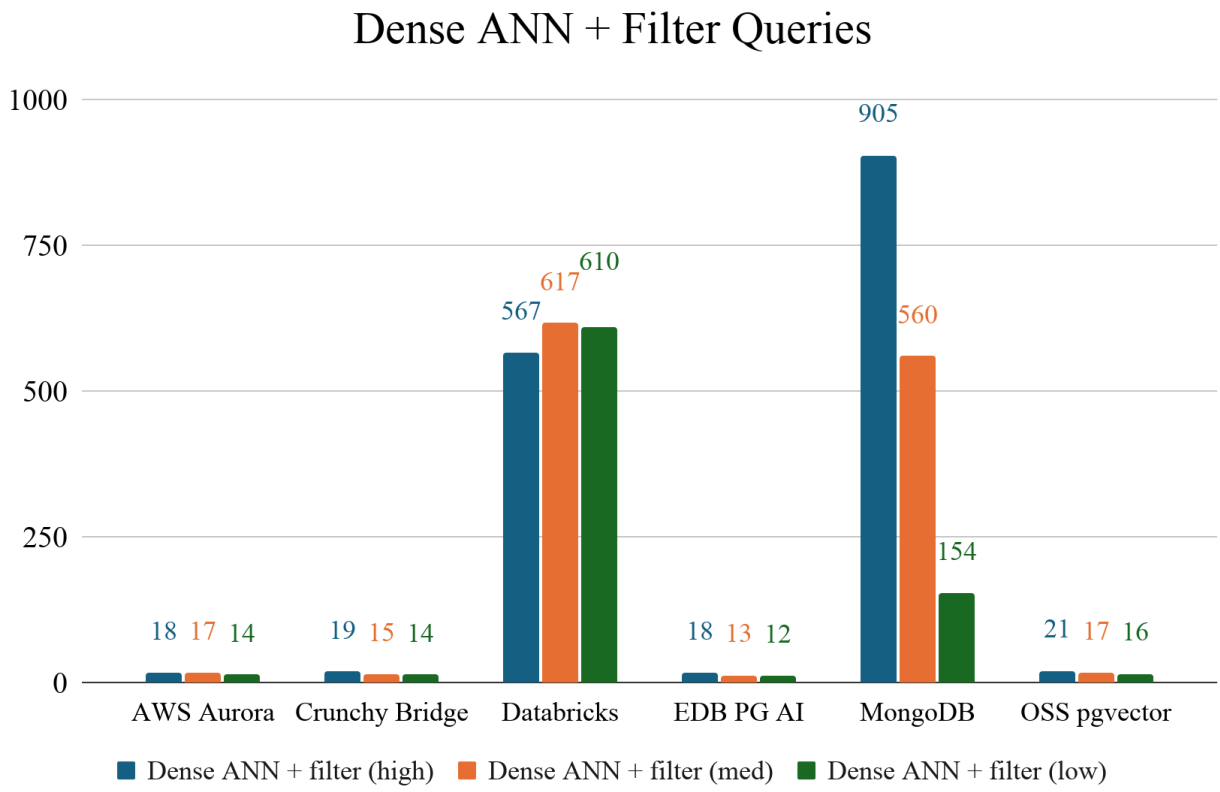

Figure 6: Dense ANN + Filter Queries

© McKnight Consulting Group 2026

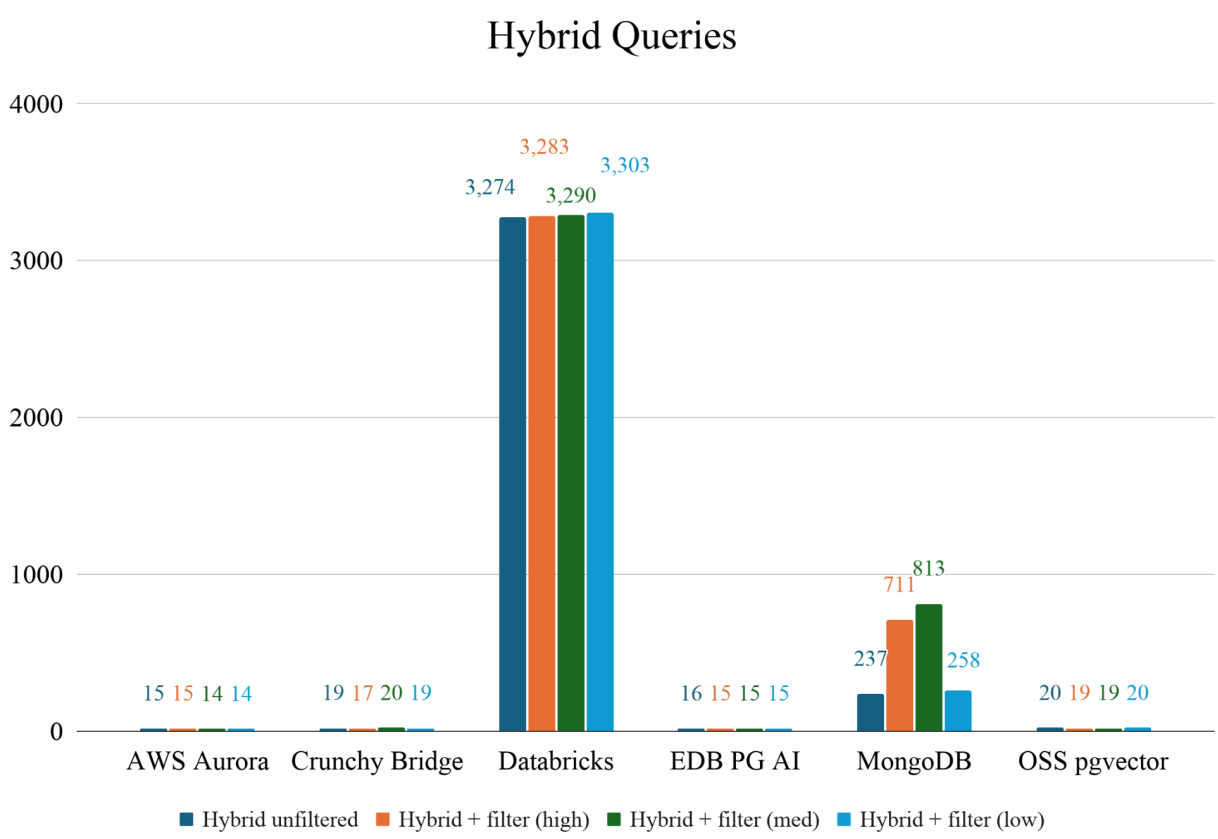

Figure 7: Hybrid Queries

© McKnight Consulting Group 2026

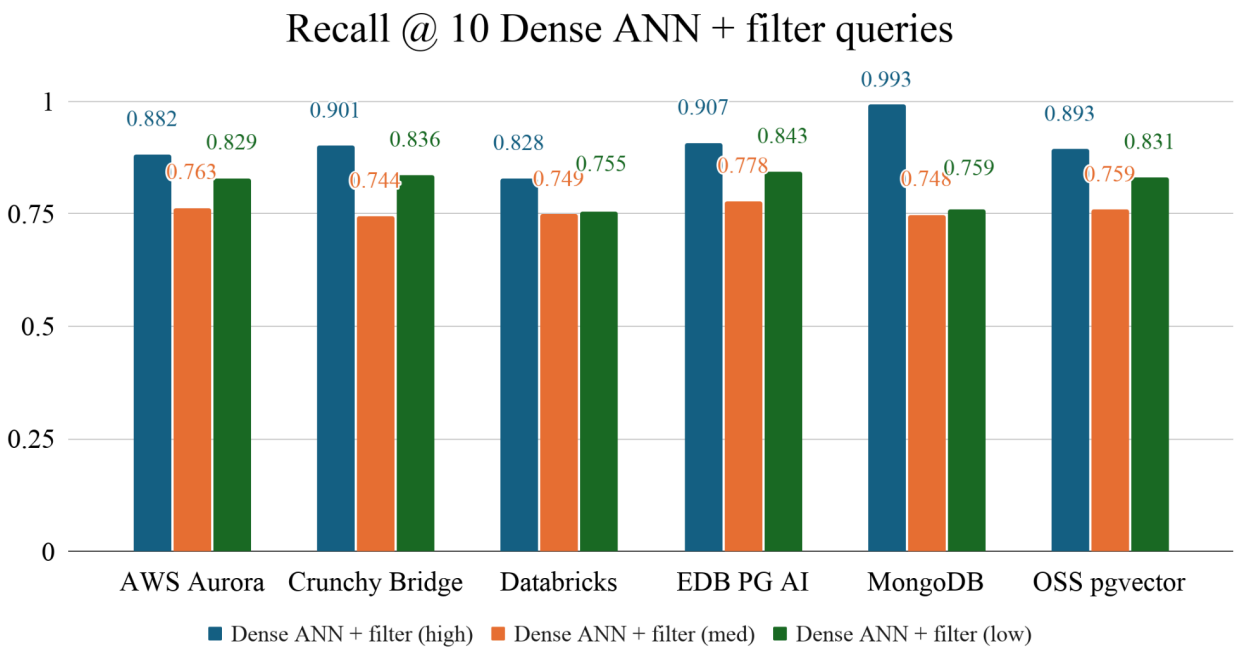

Figure 8: Recall at 10 Dense ANN + Filter Queries

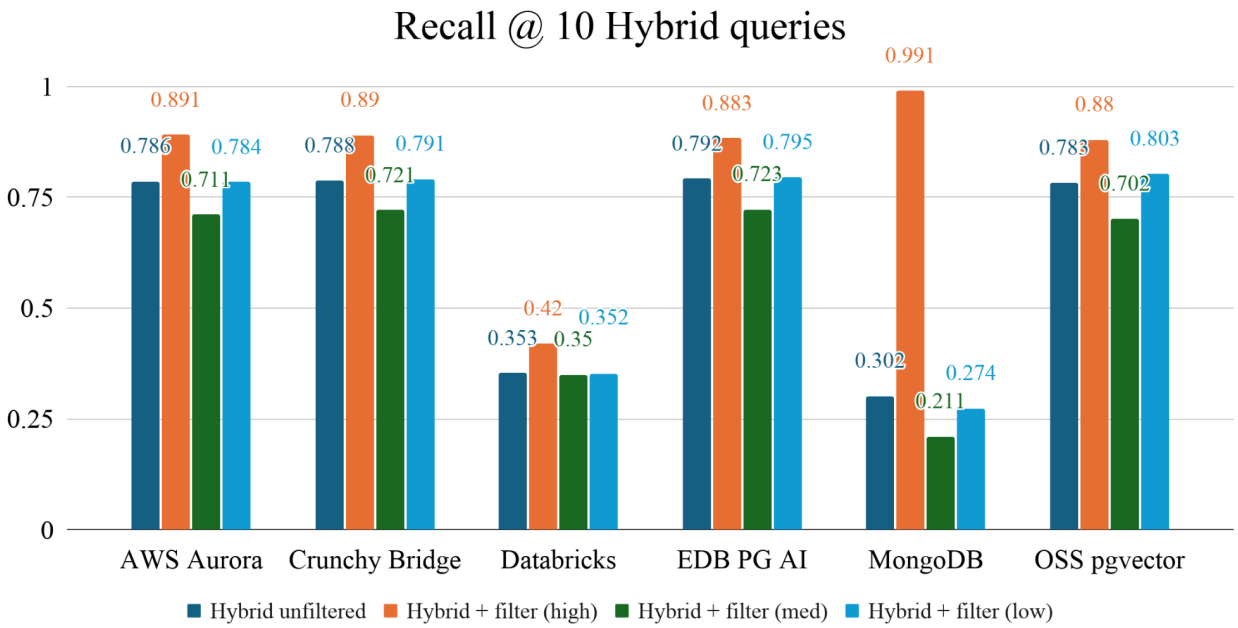

Figure 9: Recall at 10 Hybrid Queries

The Test 2 data reveals a striking performance divide between specialized relational architectures and general-purpose document or data lake platforms when executing

© McKnight Consulting Group 2026

combined semantic and structured metadata queries. MongoDB Atlas displays highly polarizing behavior, demonstrating dominant, single-digit latencies of 3 to 4 milliseconds and near-perfect recall on pure structured filtering, but severely degrading into high latencies (up to 905ms) and unacceptably low recall scores (as low as 0.211) the moment dense vector components are introduced. Conversely, the PostgreSQL ecosystem showcases exceptional query planner maturity; all four Postgres variants seamlessly execute the intersection of relational and vector logic, maintaining ultra-fast, highly stable latencies between 12ms and 21ms across all high, medium, and low selectivity bands.

Within the Postgres group, EDB PG AI consistently secures the performance edge, leading in latency across nearly every combined search configuration, including an impressive 12ms execution time for dense filtering at low selectivity. EDB PG AI also delivers the highest accuracy scores among the Postgres engines, peaking at a recall of 0.792 for dense hybrid queries under high selectivity constraints. Meanwhile, Databricks faces severe architectural bottlenecks across the board, struggling to bridge its storage and vector abstraction layers, which results in massive multi-second latency penalties exceeding 3.2 seconds for full hybrid queries alongside chronically depressed recall metrics.

EDB PG AI has the quickest query response time of dense ANN and hybrid query types. PostgreSQL-native platforms (EDB PG AI, Aurora, OSS, Crunchy Bridge) dominate hybrid and filtered ANN latency — all deliver sub-20ms across dense and hybrid modes. MongoDB's document-native filtering excels on pure structured queries (3-4ms) but degrades significantly when combining filters with ANN. EDB PG AI shows the most consistent recall across selectivity bands, with minimal precision loss as filters tighten—a signal that its HNSW post-filtering is well-tuned. Databricks is not competitive in any hybrid mode, reflecting its batch-oriented architecture.

Test 3 – Agent/RAG Workflow

Objectives:

- Demonstrate the practical architectural implications of each platform in a realistic agentic scenario
- Show what an agent loop looks like end-to-end on each platform, making the performance tradeoffs visible and tangible
- a three-arm clinical agent loop running concurrently (10 threads), plus a 30-second concurrent write+read window and a point-in-time write freshness check

Metrics:

- End-to-end query latency for the full agentic loop (milliseconds)
- Wasted-token reduction vs. semantic-only baseline (Token counts are estimated at 1 token ≈ 4 characters)

© McKnight Consulting Group 2026

Test Parameters:

Corpus size	10M records (fixed)
Scenario	Single end-to-end agentic loop
Execution	Each competitor runs the same logical loop Each agent loop fires three retrieval arms simultaneously: 1. **Vitals arm** — filtered ANN (record_type='vitals', top-k=5) 2. **EHR arm** — filtered ANN (record_type='ehr_note', top-k=5) 3. **Protocol arm** — FTS (record_type='trial_protocol', top-k=3)

Table 10: Test Parameters for Agent/RAG Workflow Test

Metric	Aurora	Crunchy	Databricks	EDB PG AI	MongoDB	OSS
p50 loop latency	33	29	5,108	**27**	699	36
p95 loop latency	40	40	6,547	**36**	800	49
p99 loop latency	48	49	6,680	**40**	853	55
Mean tokens/loop	**321.6**	329.2	434.8	330.9	435.0	330.8
Semantic wasted token rate	11.0%	12.0%	24.3%	**8.0%**	19.0%	9.0%
Hybrid wasted token rate	21.0%	18.0%	41.8%	**17.0%**	27.0%	18.0%

Table 11: Agent Loop Latency (ms):

Metric	Aurora	Crunchy	Databricks	EDB PG AI	MongoDB	OSS
Write throughput (vec/s)	40.1	74.7	0.2	85.5	**342.1**	38.3
Write p50 (ms)	25.5	12.6	3,000	11.5	**2.9**	25.3
Write p99 (ms)	35.3	31.7	16,180	15.6	**4.3**	37.9
Read throughput (loops/s)	309.6	312.0	1.5	**360.4**	14.0	274.6
Read p50 (ms)	32	29	6,486	**28**	717	37
Read p95 (ms)	41	43	10,544	**37**	888	50

Table 12: Concurrent Write + Read (30-second window)

© McKnight Consulting Group 2026

Metric	Aurora	Crunchy	Databricks	EDB PG AI	MongoDB	OSS
Immediately visible	Yes	Yes	Yes	Yes	Yes	Yes
Mean write-to-read (ms)	21	14	3,814	**12**	36	27
p99 write-to-read (ms)	36	23	4,620	**18**	857	39

Table 13: Write Freshness

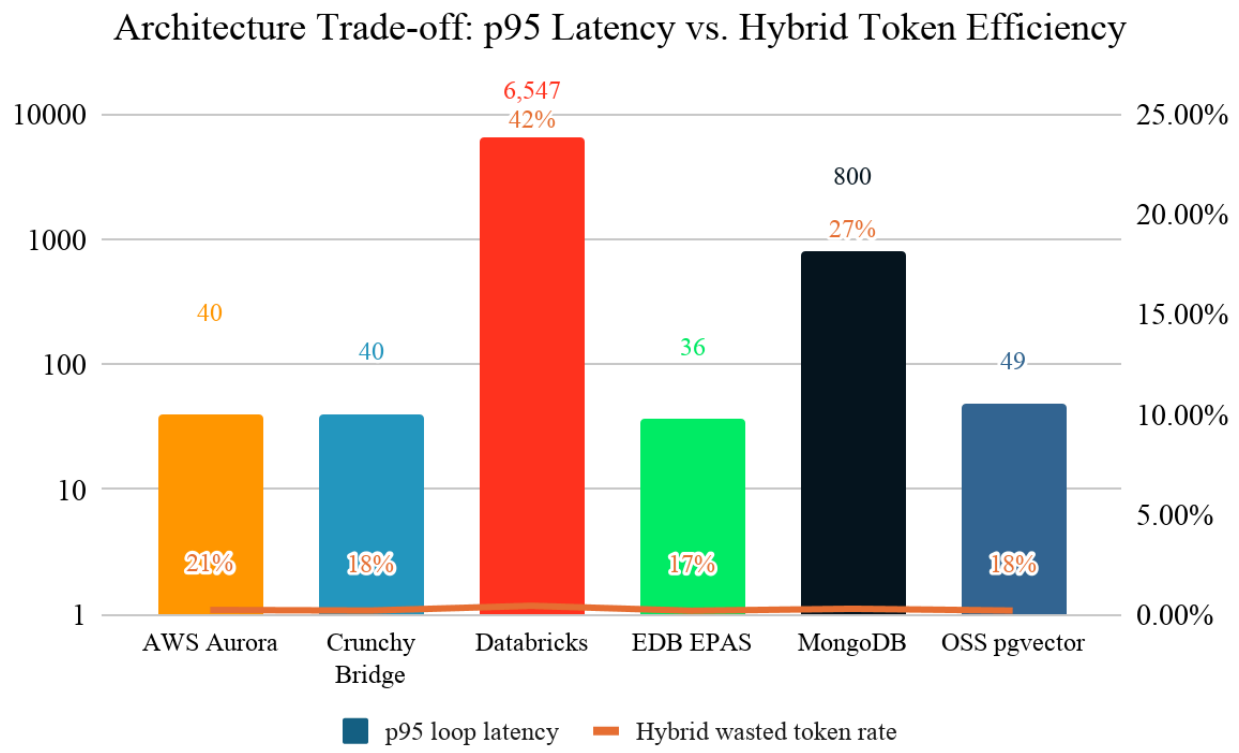

Figure 10: Architecture Trade-off: p95 Latency vs. Hybrid Token Efficiency

© McKnight Consulting Group 2026

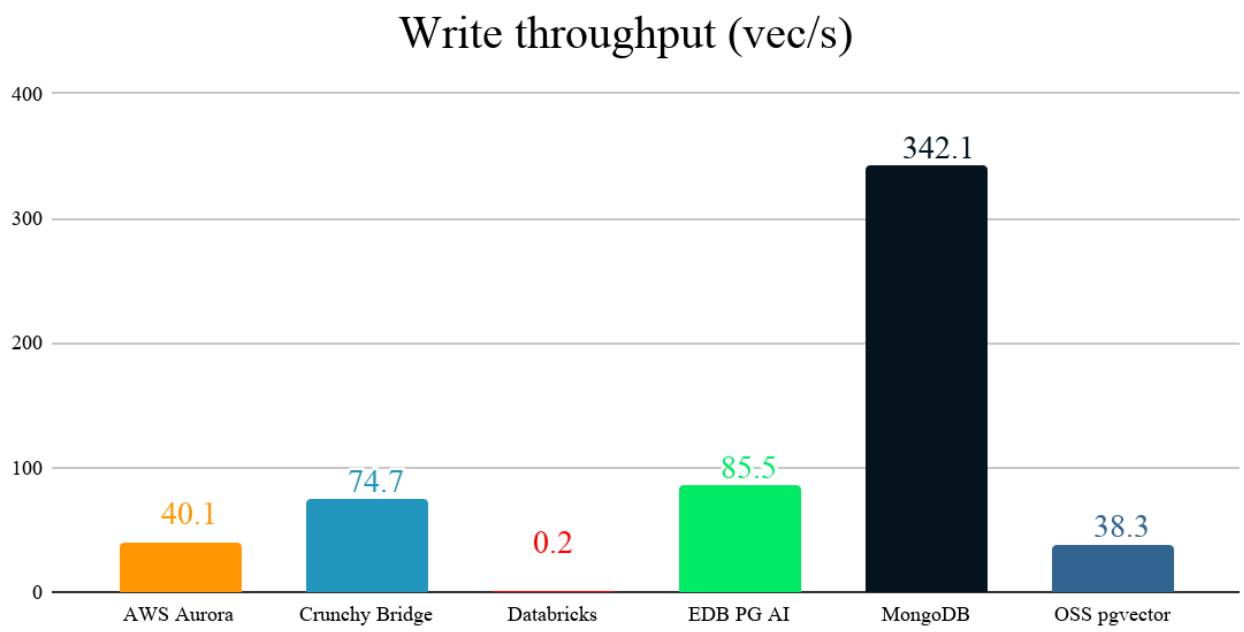

Figure 11: Write throughput (vec/s)

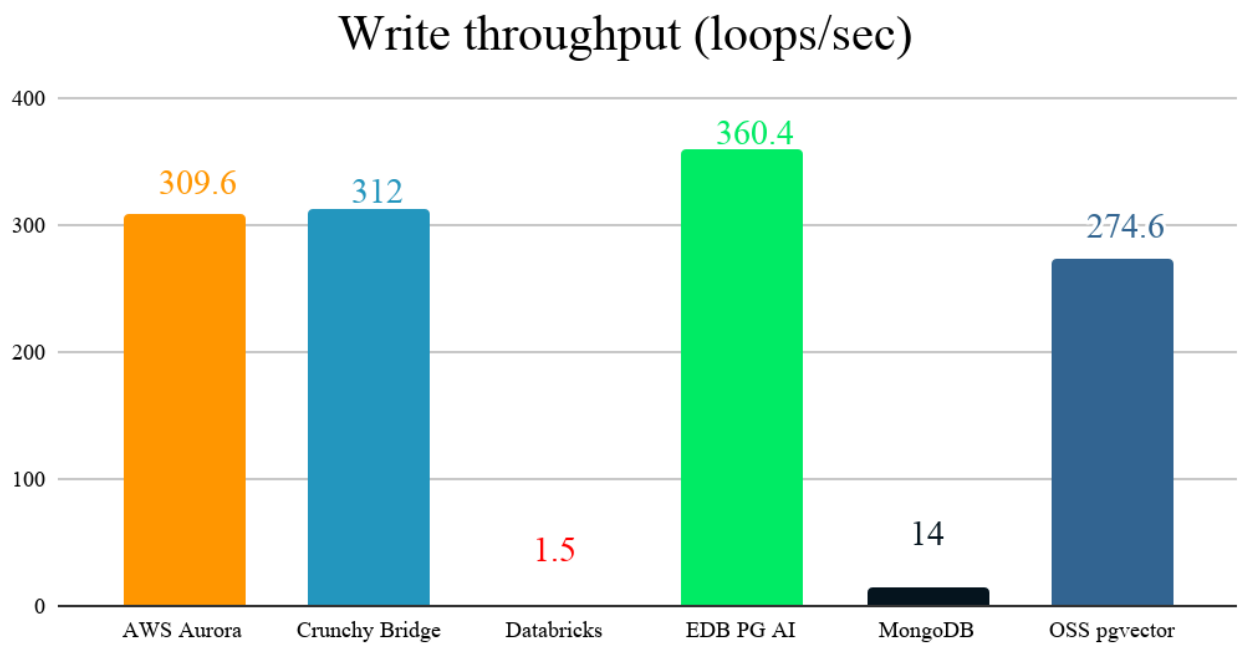

Figure 12: Write throughput (loops/sec)

© McKnight Consulting Group 2026

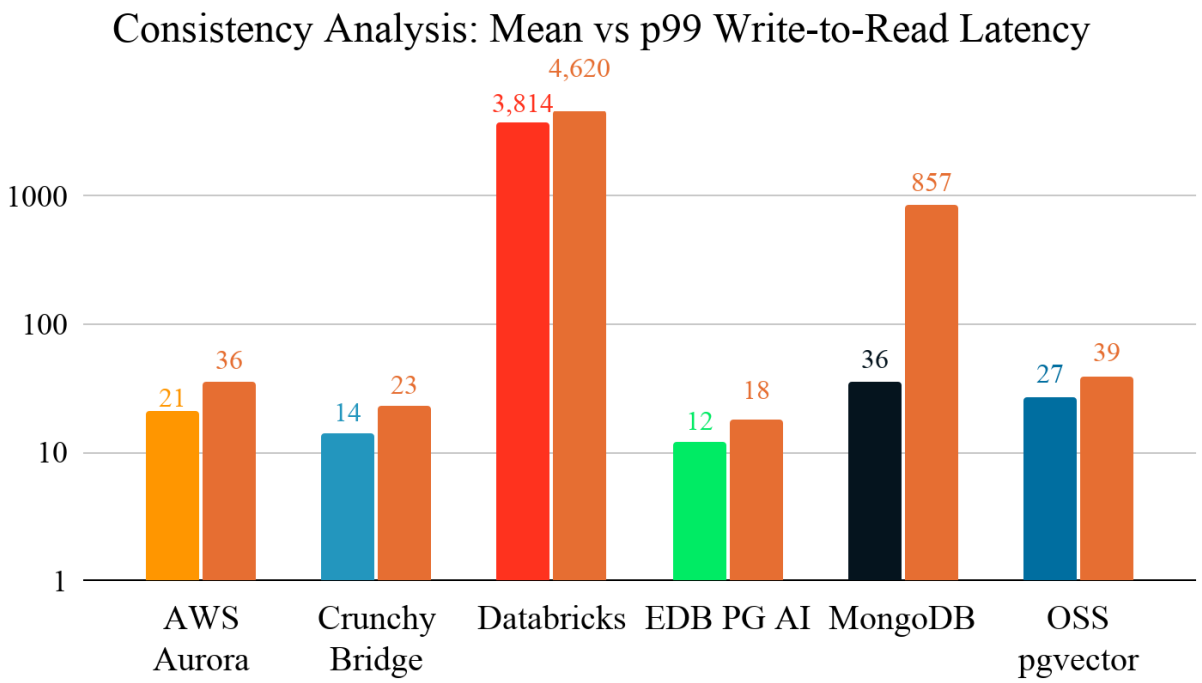

Figure 13: Consistency Analysis: Mean vs. p99 Write-to-Read Latency

The results reveal a massive latency advantage for the PostgreSQL ecosystem over non-relational alternatives. EDB PG AI achieves the top overall spot, registering a median p50 loop latency of just 27ms and maintaining excellent performance stability with a worst-case p99 tail latency of 40ms. The other Postgres engines—Amazon Aurora, Crunchy Bridge, and open-source Postgres—also display strong, tightly grouped results with p95 and p99 metrics remaining safely below 60ms. Conversely, MongoDB Atlas exhibits significant overhead with a p50 latency of 699ms, while Databricks experiences a bottleneck, taking over 5.1 seconds at p50 and jumping to 6,680ms at p99.

Beyond raw execution speed, the metrics expose a notable contrast in query optimization and token efficiency between the platforms. EDB PG AI leads the group in context efficiency, posting the lowest semantic wasted token rate at 8.0 percent and the lowest hybrid wasted token rate at 17.0 percent, closely supported by the other PostgreSQL platforms. This means Postgres query engines are highly precise at returning highly relevant chunks, minimizing context window bloat for downstream LLM generation. By comparison, non-Postgres platforms suffer from considerable inefficiency. Databricks exhibits a high semantic wasted token rate of 24.3 percent, and a hybrid wasted token rate of 41.8 percent, which forces the agent to process significantly more unnecessary text data per loop. Overall, the data shows that unified relational platforms like EDB PG AI execute concurrent, multi-vector agent loops faster while maintaining the tightest grip on context token efficiency.

© McKnight Consulting Group 2026

EDB PG AI delivers the strongest overall RAG profile: lowest agent loop latency, lowest wasted token rate (meaning the LLM receives the most relevant context per call), highest concurrent write throughput among all platforms, and fastest write-to-read visibility.

The same configuration edge explains the write results: EDB's autovacuum and WAL defaults absorb the high-frequency small inserts far more efficiently than stock Postgres, continuously indexing more efficiently, sustaining roughly twice the write throughput (85.5 vs. 38.3 vectors/second) and faster write-to-read visibility (12ms vs. 27ms)—the behavior that matters most for RAG systems that index new records continuously. Crunchy Bridge is the closest competitor on read latency and freshness but trails EDB PG AI by ~25% on write throughput. MongoDB's high write throughput is notable, but its agent loop latency (699ms p50) makes it impractical for interactive workloads. Databricks showed to be ineffective in a real-time RAG use case because of latency between its All-purpose Compute and Mosaic Search AI service.

Price-Performance

A price-performance evaluation was conducted using an effective hourly cost. EDB PG AI has a per-vCPU annual software license plus the EC2 infrastructure we used for the tests. All other platforms represent the price we paid—the on-demand “sticker price” published on each platform’s website. The table below reflects the all-in hourly rate for each platform[2]: cloud infrastructure plus managed-service fees or software licensing, on equivalent hardware in us-west-2.

Competitor	Class	SKU	Hardware	Price Per Hour
Amazon Aurora	I/O Optimized	db.r8g.16xlarge	64 vCPU, 512 GB	$11.49
Crunchy Bridge	Memory	Memory-512 1TB	64 vCPU, 512 GB	$10.66
Databricks	Mosaic AI	r8g.16xlarge	64 vCPU, 512 GB	$16.00
EDB PG AI	License + EC2	r8gd.16xlarge	64 vCPU, 512 GB	$16.97
MongoDB Atlas	Dedicated	M400_NVMe	72 vCPU, 512 GB	$26.66

Table 14: Price-per-hour as tested

A price-only comparison ignores the most important variable: how much performance do you get out of each dollar of spend? With higher price-per-performance at production AI scale, latency compounds into larger LLM API bills, longer agent loop cycles, and more compute headroom needed to hit a queries-per-second target. The table below normalizes each platform's hourly rate by how many times slower it runs than EDB PG AI at 50 million vectors, producing a speed-adjusted cost: the effective hourly price to achieve EDB PG AI-equivalent query performance.

[2] EDB's annual software license was broken down into an hourly rate and combined with its AWS EC2 infrastructure costs so its all-in expense could be compared directly against the hourly sticker prices of fully managed cloud services.

© McKnight Consulting Group 2026

Platform	Test 1 50M Vectors Query P50 (ms)	Times Slower than EDB PG AI	Effective $/hour	Price-Performance
Amazon Aurora	95	1.90	$11.49	$21.81
Crunchy Bridge	98	1.96	$10.66	$20.89
Databricks	4,023	80.46	$16.00	$1,287.36
EDB PG AI	50	-	$16.97	$16.97
MongoDB	1,083	21.66	$26.66	$577.46

Table 15: Price-per-performance

The speed-adjusted comparison shows EDB PG AI to have the lowest effective price-performance among all managed competitors once query performance is factored in.

Crunchy Bridge is 23% more expensive per unit of throughput. Amazon Aurora shows the same dynamic: a 28% premium over EDB PG AI. In both cases, the raw price advantage is completely absorbed by the query speed deficit at 50 million vectors.

The gap widens sharply for non-Postgres platforms. MongoDB Atlas combines a $26.66/hr rate with a 21.7x latency multiplier to produce a speed-adjusted cost of $577/hour, which is 34x higher than EDB PG AI. Databricks, where All-Purpose Compute and Mosaic AI Vector Search carry a high latency penalty, has an effective cost of $1,287/hour, 76x that of EDB PG AI.

For enterprise AI infrastructure planning, the conclusion is clear. EDB PG AI leads by price-performance. At 50 million vectors, no managed platform evaluated in this study delivers a lower cost per unit of query performance than EDB PG AI.

© McKnight Consulting Group 2026

Conclusion

Our testing demonstrated that Postgres outperformed the specialized platforms, and among Postgres providers, EDB PG AI led the field. It provides the single-query, omni-data access that AI agents require with ACID-guaranteed accuracy at millisecond speed — without the integration overhead that competing platforms impose.

For an enterprise buyer, these benchmarks prove that EDB PG AI eliminates the classic architectural tradeoff between speed, accuracy, and cost when deploying production-grade AI. By delivering ultra-low latencies (such as an 11ms p50 at 10M vectors) alongside the highest recall across all tested scales, EDB PG AI ensures that real-time applications remain highly accurate and responsive even as data volumes grow to 50 million vectors and beyond.

Furthermore, its industry-leading token efficiency (minimizing wasted semantic tokens to just 8.0%) directly translates to massive cost savings on LLM API bills, while its rapid 12ms write-to-read freshness ensures that RAG workflows always operate on live, up-to-the-minute data. Ultimately, this means an enterprise can confidently standardize on a single, highly scalable Postgres foundation to handle intensive AI workloads without the complexity, latency penalties, or overhead of maintaining separate, niche vector databases.

© McKnight Consulting Group 2026

About EDB

EDB Postgres® AI (EDB PG AI) is the sovereign data and AI platform for the agentic enterprise. It brings together a unified data layer, governance, sovereign control and orchestration, and an agent runtime environment, giving enterprises a trusted foundation for AI on infrastructure they own and control. The platform unifies transactional, analytical, and AI workloads in a single Postgres-based architecture—eliminating ETL, data movement, and operational fragmentation. And you choose where and how to deploy: on-premises, cloud, managed, or certified appliance. The outcome is production-ready sovereign AI in days or weeks, not months. To learn more, visit www.enterprisedb.com.

© McKnight Consulting Group 2026

About McKnight Consulting Group

Information Management is all about enabling an organization to have data in the best place to succeed to meet company goals. Mature data practices can integrate an entire organization across all core functions. Proper integration of that data facilitates the flow of information throughout the organization which allows for better decisions – made faster and with fewer errors. In short, well-done data can yield a better run company flush with real-time information... and with less costs.

However, before those benefits can be realized, a company must go through the business transformation of an implementation and systems integration. For many that have been involved in those types of projects in the past – data warehousing, master data, data lake, data integration, AI/ML, big data, analytics – the path toward a successful implementation and integration can seem never-ending at times and almost unachievable. Not so with McKnight Consulting Group (MCG) as your integration partner. MCG has successfully implemented data solutions for our clients for 15 years. We understand the critical importance of setting clear, realistic expectations up front and ensuring that time-to-value is achieved quickly.

MCG has helped hundreds of enterprises with analytics, big data, AI, master data management and “all data” strategies and implementations across a variety of industries and worldwide locations. MCG offers flexible implementation methodologies that will fit the deployment model of your choice. The best methodologies, the best talent in the industry and a leadership team committed to client success makes MCG the right choice to help lead your project.

MCG, led by industry leader William McKnight, has deep data experience in a variety of industries that will enable your business to incorporate best practices while implementing leading technology. See www.mcknightcg.com.

© McKnight Consulting Group 2026

Disclaimer

McKnight Consulting Group (MCG) runs all its tests to strict ethical standards. The results of the report are the objective and unbiased results of the application of tests to the simulations described in the report. The report clearly defines the selected criteria and process used to establish the field test. The report also clearly states the data set sizes, the platforms, the methods, etc. that were used. The reader is left to determine for themselves how to qualify the information for their individual needs. The report does not make any claim regarding third-party certification and presents the objective results received from the application of the process to the criteria as described in the report. The report strictly measures the metrics listed and does not purport to evaluate other factors that potential customers may find relevant when making a purchase decision. This is a sponsored report. The client chose its configurations, while MCG chose the tests, configured the databases and testing applications, and ran the tests. MCG also chose the most compatible configurations for the other tested platforms. Choosing compatible configurations is subject to judgment. The information necessary to replicate this test is included. Readers are encouraged to compile their own representative configuration and test for themselves.

© McKnight Consulting Group 2026