EDB POSTGRES AI

Solving the Vector Database Dilemma

One platform, 4x performance, 68% faster AI deployment

www.enterprisedb.com

ACKNOWLEDGMENTS

Technical Writers and Editors

Purnima Phansalkar
Dave Stone
Dr. Sala Muthukrishnan

Reviewers and Contributors

Jack Christie
Dunith Danushka
Lucie Zeng
Pranish Kumar
Kamesh Vemu
Vibhor Kumar
Scott McLaughlin
Rick Hill
Sandeep Arora
Janus Hägele
Ava Chawla
Rob Gibbon
Tim Henrion

Creative Team

Storm Brewer
Maya Fisher
Mary VanClay

EXECUTIVE SUMMARY

EDB Postgres® AI (EDB PG AI), including its **AI Factory** and combined with native **vector solutions**, provides a **unified, enterprise-grade platform** for vector databases and AI workloads. Key differentiators include:

- **Unified data platform:** Combines transactional, analytical, and vector operations on a single, **ACID-compliant Postgres engine**, eliminating separate vector stores
- **Performance and efficiency:**
    - **4.22x faster query performance** via optimized indexing and intelligent query orchestration [1]
    - **18x greater storage efficiency** through advanced compression and optimization [2]
    - **67% reduction in development complexity** via automated AI pipelines and **low-code GenAI Builder** [3]
- **Accelerated AI adoption:** Compresses AI project timelines from months to weeks, enabling faster time to market
- **Enterprise security and sovereignty:** End-to-end protection with **encryption, role-based access control (RBAC), identity management, network isolation**, and full control over data and models
- **Seamless integration:** Enables hybrid AI and analytics workflows by unifying vector and relational workloads within one platform
- **Flexible deployment:** Supports **multi-cloud, hybrid, and on-premises environments** without sacrificing performance or compliance
- **Operational simplicity:** Through EDB PG AI Factory, automates embedding management, pipeline orchestration, and model workflows, reducing operational overhead while boosting agility
- **Proven benchmark advantage:** Superior **throughput, low-latency similarity search**, and efficient storage footprint to reduce costs compared to specialized vector databases

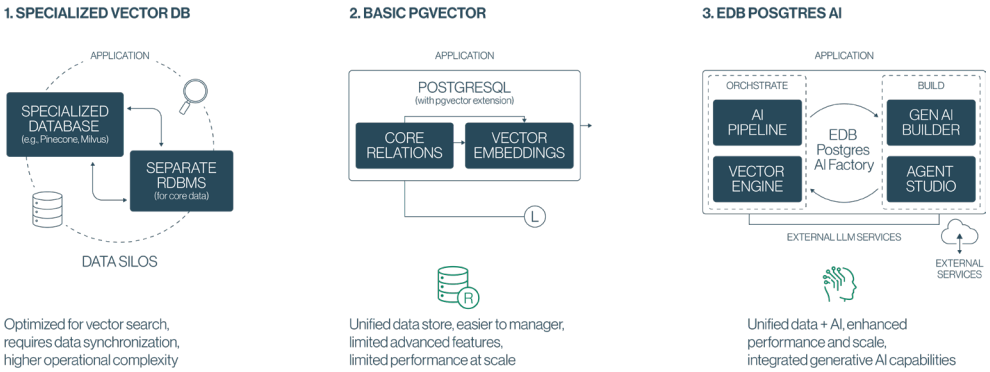
Impact: EDB PG AI empowers organizations to build **scalable, secure, and sovereign AI applications**, accelerating innovation and operational efficiency across enterprise AI initiatives.

TABLE OF CONTENTS

1. Introduction

The staggering growth of generative AI (GenAI)–driven applications, from semantic search and recommendations to intelligent assistants, has made vector databases a critical component of modern data infrastructure. While pgvector, the Postgres extension for vector similarity search, provides a strong open source foundation, enterprise-grade AI workloads demand far more: scalability, performance, reliability, and compliance.

EDB PG AI redefines what's possible by extending the power of pgvector into a software-optimized GenAI data platform—one that unifies transactional, analytical, and vector workloads under a single, sovereign Postgres engine. Through intelligent automation and the deep integration of AI Factory, EDB PG AI enables organizations to seamlessly operationalize retrieval-augmented generation (RAG), embeddings, and agentic AI solutions without abandoning trusted Postgres architecture.



At its core, **EDB PG AI** enhances pgvector with:

- **4.22x faster query performance** through optimized indexing and query orchestration
- **18x greater storage efficiency** achieved through **object storage integration, advanced compression, and optimized data handling**
- **67% reduction in development complexity** enabled by **automated AI pipelines** and **low-code GenAI Builder tools**, dramatically simplifying workflows compared to DIY implementations
- **Accelerated time to production—from 28 weeks to 9 weeks**, enabling rapid AI adoption

This white paper explores how **EDB PG AI** overcomes the architectural and operational limitations of vanilla pgvector, delivering an enterprise-ready vector database that adheres to database **ACID (atomicity, consistency, isolation, and durability) principles, data sovereignty, and open standards**. The sections ahead examine the evolution of vector-powered applications, dissect internal mechanics of pgvector, and present the EDB architectural enhancements that make **GenAI production scalable, secure, and efficient**—all within the familiar Postgres ecosystem.

With the growing complexity of AI-driven data ecosystems, understanding the foundation of vector intelligence is essential. The next section explores how vector embeddings have become the backbone of modern AI applications and why this shift demands a new class of data architecture.

2. Foundations of vector intelligence in modern data systems

Vector embeddings have rapidly emerged as the foundation for modern AI, powering applications such as semantic search, recommendation engines, and multimodal AI. While pgvector has made it easy for Postgres users to implement vector search, organizations quickly encounter limitations when scaling to production-grade GenAI solutions. Performance bottlenecks emerge from demanding vector indexing and retrieval operations. Storage costs escalate as high-dimensional vectors consume vast disk space. Developer complexity increases when we integrate multiple components such as retrieval systems, chunking pipelines, and embedding models while keeping pace with a rapidly evolving AI ecosystem. And time to production extends as piecemeal integrations introduce security considerations and operational overhead. To truly understand these bottlenecks and the need for next-generation data architecture, we must first look at the history and meteoric rise of vector data.

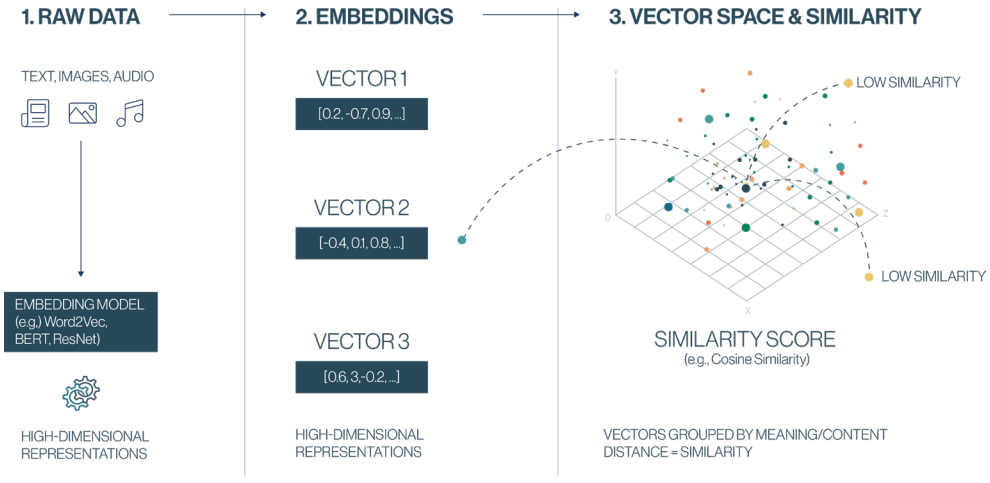
2.1. The rise of vector-powered applications

As per 2025 statistics, ChatGPT processed more than 2 billion queries daily,[4] many powering RAG applications that rely on vector similarity search to retrieve relevant context. This explosive growth in AI applications has fundamentally changed how enterprises think about data architecture.

Every day, 2.5 quintillion bytes of data are created worldwide. For enterprises, the challenge is acute: The vast majority of their data—documents, images, audio, and video—is unstructured. Traditional databases can store this data but cannot understand its meaning. **Vector embeddings** bridge this gap, enabling machines to comprehend the underlying meaning, context, and relationships among disparate data by representing them as numerical coordinates in vector space.

2.2. Understanding vector embeddings

Vector embeddings convert complex, unstructured data—text, images, audio, and more—into numerical arrays that machines can efficiently process. Each embedding is a dense vector ($x \in R^n$) representing latent features learned by models, typically through **neural networks**—computational systems designed to recognize patterns and relationships in data. These networks effectively compress the vast, sparse data space of the original data into a highly structured numerical representation. Geometric relationships in this vector space encode semantic or structural similarity: The closer two vectors are, the more related their underlying meanings. This closeness is measured using distance metrics such as **cosine distance**, which evaluates the angle between two vectors to assess directional similarity, or **Euclidean distance**, which measures the straight-line spatial separation between them.



For example, in a text-embedding space, *cat* and *kitten* lie closer together than *cat* and *airplane*. In essence, embeddings provide a continuous, algebraic representation of meaning, enabling similarity search, clustering, and reasoning across data types.

Different embedding models produce vectors of varying dimensionality (n), depending on their design and use case. These large vector sizes (often hundreds to thousands of dimensions) introduce the high-dimensional challenges that data architects must solve, as detailed later in this paper. The table below highlights popular models and explains how their embedding dimensions align with specific AI workloads, from text understanding and generation to multimodal and real-time applications.

Model	Dimensions	Use Case	Performance
BERT	768	Text understanding	Baseline
GPT-3	12,288	Text generation	High accuracy
CLIP	512	Image-text matching	Multimodal
Whisper	1,024	Audio transcription	Real time

By mapping diverse data into a shared vector space, machines can find semantically similar content regardless of exact wording. This capability powers a variety of use cases, including RAG for enhancing large language model (LLM) responses with contextual data, semantic search for meaning-based information retrieval, recommendation systems that infer user preferences from behavior and content similarity, and anomaly detection that identifies unusual patterns across complex datasets.

RAG Systems

Enhance LLM responses with contextual data from your knowledge base

Hybrid Semantic Search

Find information by meaning, not just keywords

Recommendations

Suggest content based on user behavior and preferences

Anomaly Detection

Identify unusual patterns in complex datasets

Vector embeddings power the shift from **literal keyword search** to **contextual semantic search**, enabling systems to interpret meaning rather than just text. This evolution allows applications to deliver more accurate, intent-aware results—a foundation for intelligent assistants, recommendation systems, and enterprise AI search.

For instance, here is a table comparing traditional keyword-based search vs. vector-based semantic search when a user searches for *cheap flights to Paris*:

Approach	Mechanism	Example Results
Traditional search (Keyword matching)	Matches only literal terms in text or metadata	✗ Misses related phrases such as: • *affordable airfare to France* • *budget travel France* • *low-cost carriers CDG*
Semantic search (Vector similarity)	Uses vector embeddings to capture meaning and context, not just words	✓ Recognizes related concepts and intent ✓ Finds semantically similar results ✓ Delivers more complete, relevant answers

This shift from literal to conceptual understanding forms the backbone of next-generation AI applications—enabling more accurate, intelligent, and human-like responses in search, recommendation, and conversational systems.

2.3. The database architecture dilemma

How can organizations integrate GenAI tooling and vector capabilities—such as similarity search and AI-driven recommendations—into their existing data infrastructure to enhance search, analytics, and insights, without disrupting existing transactional workflows, managing fragmented toolchains, or breaking systems that already work?

Implementing vector-powered applications requires a critical architectural decision, with fundamental trade-offs impacting cost, performance, and time to market:

- **Specialized vector databases** prioritize search speed but introduce high operational overhead and lack core enterprise guarantees.
- **Basic vector extensions** for traditional databases (such as standard pgvector) ensure data consistency but often compromise performance and lack the advanced tooling necessary for large-scale production use.

2.3.1. The dual challenge: Data synchronization and tooling fragmentation

The architectural dilemma extends beyond basic data storage to encompass the entire GenAI application development lifecycle, introducing two major operational roadblocks:

- **Data synchronization burden:** Specialized vector databases operate separately from transactional data stores, forcing teams to build and maintain complex **extract, transform, and load (ETL) pipelines**. Every data change—a product update, customer record modification, or content revision—requires extraction from the source database, transformation into vector embeddings (often calling external APIs), and loading into the vector store. This creates synchronization lag, multiplies failure points, and demands dedicated data engineering resources just to keep systems aligned. As data volumes grow and update frequencies increase, these pipelines become increasingly difficult to maintain and costly to operate.
- **Tooling fragmentation problem:** Building production GenAI applications requires orchestrating an entire ecosystem of tools: embedding models for vectorization, chunking strategies for document processing, retrieval frameworks for context assembly, prompt templates, and LLM API integrations. With specialized vector databases, each component exists as a separate service requiring its own configuration, monitoring, and maintenance. Development teams spend disproportionate time on integration work by connecting APIs, handling failures, managing versions, and debugging across system boundaries rather than focusing on application logic and user experience. This tooling sprawl multiplies operational complexity and extends time to market.

2.3.2. Key architectural trade-offs

The burdens of synchronization and fragmentation are part of a larger set of trade-offs impacting the long-term viability of an architecture. The following table details some of the key trade-offs spanning consistency, operational load, and GenAI integration for both approaches:

Feature	Specialized Vector Databases	Basic DBMS Vector Extensions
Data consistency	**Eventual/poor:** High risk of data drift; lacks native ACID guarantees	**Excellent/native:** Inherits strong ACID transactional guarantees from the host DBMS
Operational load	**High:** Requires separate infrastructure (provisioning, security, monitoring, failover) and custom ETL/sync pipelines	**Moderate:** Runs within the existing operational stack, but requires extensive custom code for advanced scaling and GenAI workflow
Indexing/ performance	**Optimized:** High performance for pure vector search, often using proprietary, specialized indexes (e.g., highly tuned HNSW)	**Compromised:** Performance can suffer at high scale due to reliance on general-purpose DBMS architecture and lack of dedicated vector workload isolation
GenAI integration	**Fragmented:** Requires building and maintaining external tooling for chunking, embedding generation, and retrieval orchestration	**Manual:** No integrated tooling; developers must build custom solutions for ingestion, embedding, and lifecycle management

The reality is that neither approach meets enterprise needs. What's needed is a unified platform that combines native vector capabilities with enterprise database features and integrated GenAI tooling—eliminating synchronization complexity, reducing toolchain sprawl, and delivering production-grade performance and reliability.

2.4. Why ACID compliance matters for GenAI

The loss of native database guarantees is the single greatest risk introduced by fragmented vector architectures. As AI workloads increasingly rely on vector embeddings for tasks such as RAG, ensuring **transactional integrity**, **data consistency**, and robust **security** is paramount.

While stand-alone vector databases often compromise reliability for performance, EDB PG AI with pgvector effectively threads the needle, delivering a unified platform that provides the high-performance vector search capabilities of specialized systems while retaining the rock-solid, enterprise-grade guarantees of Postgres—eliminating the need for compromise.

The table below details why each component of the ACID standard is nonnegotiable for GenAI applications:

ACID Property	Relevance to GenAI/Vector Applications	Implication of Failure (Fragmented DBs)
Atomicity	Ensures vector and source data updates succeed or fail as a single, indivisible transaction	If text fails to vectorize or index, the source data might still be committed, leading to an unsearchable data state
Consistency	Guarantees that vectors always accurately reflect the current state of the source data in real time	Reliance on asynchronous ETL leads to data drift, causing RAG applications to provide stale or incorrect context to the LLM
Isolation	Prevents concurrent indexing, search queries, and transactional writes from interfering with one another	High-throughput indexing operations (ANN building) can severely degrade the latency and stability of user-facing transactional applications
Durability	Guarantees that vector data and metadata survive system failures and are fully recoverable via standard backup/restore	Recovery relies on external, non-transactional backups, risking data loss or requiring a lengthy, complex, and manual reindex of the entire corpus

By unifying vector storage with the transactional guarantees of Postgres, EDB PG AI provides the required foundation. ACID compliance is not the only factor, but it is the essential bedrock that supports the performance, scalability, integration, and operational governance required by the modern enterprise.

2.5. Industry standard capabilities for vector database

Beyond ACID compliance, production-grade vector databases must deliver a comprehensive set of capabilities that support GenAI workloads at scale. The following is a framework to evaluate whether a vector offering truly meets enterprise requirements, such as ensuring that vector embeddings can be stored, queried, and managed efficiently while maintaining security, observability, and integration with AI pipelines.

Category	Industry-Standard RFI Requirements	What It Means
Core vector functions	• Native support for vector data types • Vector similarity search (cosine, inner product, Euclidean) • Indexing support (HNSW, IVFFlat, etc.) • Upsert, delete, and update of vector	Foundational vector operations needed for semantic search, recommendations, and RAG
Scalability and performance	• Horizontal scaling for storage and compute • Parallel query execution • Support for high-dimensional embeddings (>1,000 dimensions) • Real-time or low-latency retrieval	Ensures the database can handle millions to billions of embeddings efficiently

Table continued on next page.

Table continued from previous page.

Category	Industry-Standard RFI Requirements	What It Means
Advanced performance	• GPU acceleration for vector operations • Support for specialized hardware and customer accelerators	For ultra-low-latency requirements or massive-scale real-time inference, GPU acceleration can optimize embedding generation and similarity search
Integration and extensibility	• Standard APIs (SQL, REST, gRPC) • Integration with AI/ML frameworks (LangChain, LlamaIndex, Hugging Face, OpenAI, etc.) • Plugin or extension support	Enables easy embedding generation and consumption by AI pipelines and enterprise apps
Hybrid data support	• Ability to store relational and vector data together • Join vector and scalar queries natively	Essential for unified analytics and hybrid transactional-analytical processing (HTAP)
Security and compliance	• Encryption at rest and in transit • RBAC • Audit logging and data lineage • Compliance with GDPR/PDPA/ industry standards	Protects data integrity and supports auditability under regulatory frameworks
Deployment flexibility	• Cloud, on-premises, and hybrid deployment options • Kubernetes or container orchestration compatibility • Backup and replication support	Allows sovereignty and high availability across environments
Observability and management	• Monitoring dashboards • Query performance metrics • Backup/restore automation • Logging and alerts	Ensures operational reliability and proactive management
Enterprise readiness	• High availability and clustering • Multi-tenant isolation • Enterprise support SLAs • Integration with LDAP/OAuth	Defines production-grade reliability and enterprise-level governance

Having established why vectors are central to AI and the challenges of scaling traditional architectures, we now turn to pgvector—the foundational Postgres extension that enables native vector storage and search. This section examines its architecture, indexing strategies, and inherent production limitations.

3. pgvector foundation: Capabilities and limitations

Now that we have analyzed the critical challenges in data fragmentation and established a comprehensive framework of enterprise requirements (Sections 2.3–2.5), we can turn our attention to the **core technology** of EDB PG AI's unified approach: **pgvector**. This open source Postgres extension introduces native vector data types and similarity search capabilities directly within the database, allowing teams to store and query high-dimensional embeddings (up to 16,000 dimensions) alongside operational data using familiar SQL.

While pgvector eliminates the need for separate vector databases with your existing Postgres environment, there are production limitations at scale, including performance degradation beyond 10 million vectors, manual embedding management, lack of vector-specific monitoring, and operational complexity around ingestion and recovery. In this section, we look at architecture and indexing strategies, then explore these production challenges to establish the context for how EDB PG AI extends pgvector with enterprise-grade optimization, automation, and unified GenAI orchestration.

3.1. Introduction to pgvector

pgvector is an open source Postgres extension that introduces native vector data types and similarity search capabilities directly within the database. It provides the essential, high-quality primitives needed to begin building vector-powered applications within a familiar relational environment.

3.2. Distance metrics

The effectiveness of vector search relies on accurately calculating the similarity between a query vector and the stored embeddings. pgvector provides native support for three distance metrics, allowing developers to choose the scoring function most appropriate for their specific AI workload:

Metric/Algorithm	Description	Primary Use Case
Cosine distance	Measures the angle between two vectors, capturing directional (semantic) similarity	Best for text similarity and conceptual matching
Inner product	Measures the dot product, capturing both direction and magnitude	Often used in recommendation systems (e.g., user-item pairing)
Euclidean distance	Measures the straight-line spatial separation (L2 norm) between two vectors	General geometric distance, effective for visual or spatial data

3.3. Indexing strategies and performance characteristics

Vector search is not based on exact match but on approximate nearest neighbor (ANN) search, which requires specialized indexing to deliver fast, scalable results. Efficient vector search in pgvector depends heavily on the choice of indexing strategy, as each algorithm offers distinct trade-offs in build time, recall accuracy, and resource consumption.

pgvector supports two primary indexing methods—**IVFFlat** (inverted file flat) and **HNSW** (hierarchical navigable small world)—each optimized for specific workload patterns and data characteristics.

IVFFlat indexing employs k-means clustering to partition vector space into inverted lists, enabling faster index builds and lower memory overhead. It is best suited for static or batch-oriented datasets where data updates are infrequent.

HNSW indexing constructs a multilayer proximity graph that allows efficient ANN search with high recall and low latency, making it ideal for dynamic, high-dimensional, or real-time workloads.

The essential characteristics and trade-offs between IVFFlat and HNSW indexing are summarized below:

Characteristic	IVFFlat	HNSW
Algorithm	Inverted file with flat quantization (K-means clustering → inverted lists)	Hierarchical navigable small world (multi-layer graph)
Build time	Fast — O(n); quick index build; bounded insertions	Slower — O(n log n); supports dynamic insert/update
Query speed	Moderate	Fast
Memory usage	Lower	Higher (1.5–2x)
Accuracy/recall	95%–98% recall	98%–99.5% recall
Data requirements	Requires pre-populated data for optimal performance	Handles frequently updated datasets efficiently
Performance Sensitivity	Depends on number of inverted lists configured	Consistent performance across high-dimensional vectors
Ideal Use Case	Batch processing, static datasets, cost-sensitive scenarios	Real-time queries, dynamic and high-dimensional datasets

To further mitigate the significant memory usage and storage challenges associated with high-dimensional vectors, recent pgvector versions introduce **quantization techniques** such as scalar quantization (halfvec) and binary quantization to reduce vector size and improve query performance.

3.4. Limitations and production challenges

While pgvector provides a powerful foundation for vector storage and similarity search inside Postgres, moving from proof of concept to enterprise-grade deployment reveals significant gaps. Production environments demand automation, scalability, resilience, and performance consistency. Without these capabilities, organizations encounter operational friction, performance bottlenecks, and high maintenance overhead as workloads scale.

The core challenges fall into four categories. First, performance degrades exponentially as vector datasets grow beyond 10 million records, with query latency increasing from milliseconds to seconds, directly impacting user experience. Second, pgvector lacks automated embedding management, requiring teams to build custom pipelines for generating, updating, and synchronizing embeddings with source data. Third, without vector-specific monitoring and observability, operations teams have no visibility into recall rates, index health, or query performance trends, making it difficult to diagnose issues before they impact production. Finally, operational complexity increases dramatically as organizations must manually configure ingestion workflows, implement backup and recovery procedures, and integrate multiple disconnected tools to build complete RAG systems.

The following table outlines these key limitations and their business impact:

Challenge	Problem	Example	Impact	Solution
Performance degradation at scale	Query latency increases exponentially beyond 10M vectors	100ms @ 1M vectors → 2.5s @ 10M vectors	User experience degradation, timeout errors	Optimized indexing and query engine ensures low query latency even at large scale
Manual embedding management	No built-in embedding generation	50+ lines of Python code required per embedding pipeline	Adds 3–4 weeks of development time	Automated embedding pipelines automate embedding generation and ingestion
Lack of monitoring	No vector-specific metrics or alerting	Unable to track recall rates or index efficiency	Blind to performance degradation and index health	Built-in monitoring and alerting provide vector-level observability and proactive alerts
Operational and integration complexity	Manual setup for ingestion, backups, and disaster recovery	Multiple disconnected tools for RAG systems	Increased operational risk and maintenance effort	Unified AI Factory platform integrates monitoring, automation, and orchestration

Let's explore how EDB PG AI addresses these challenges, offering a unified, production-ready platform that combines transactional, analytical, and vector workloads, orchestrates GenAI workflows, supports RAG pipelines, and provides enterprise-grade data management and governance.

4. EDB PG AI Platform:
Orchestrate, build, and deploy

As we have established, production challenges emerge when scaling pgvector to enterprise workloads, such as performance degradation beyond 10 million vectors, manual embedding management, lack of vector-specific monitoring, or operational complexity around ingestion and recovery. EDB PG AI addresses these limitations by extending pgvector with enterprise-grade capabilities while eliminating the tooling fragmentation that plagues specialized vector databases.

4.1 From vector store to unified AI platform

EDB PG AI delivers core components working together as a unified AI Factory for GenAI application and agent development across three phases: **Orchestrate** (data preparation and embedding generation), **Build** (creating AI agents and applications), and **Deploy** (production rollout with full governance).

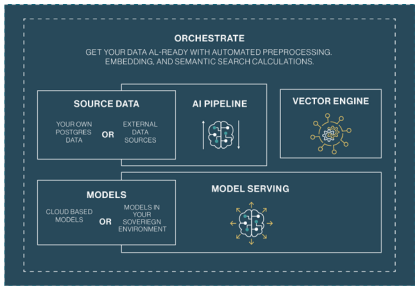
Together, these components transform Postgres from a database with vector support into a complete AI platform. The following sections explore how each component works and the enterprise value it delivers.

Orchestrate: Data preparation and vector embeddings

Purpose: Automates the ingestion, transformation, and embedding of data from diverse sources while maintaining full data sovereignty

Key capabilities:

- **Data integration:** Connects natively to Postgres tables, object stores, or external sources. Data remains within enterprise control.
- **AI Pipeline:** Handles continuous syncing, data transformation, and intelligent embedding with minimal code
- **Vector Engine:** Leverages pgvector for Postgres-native vector storage and rapid semantic search in your trusted environment, eliminating the ETL and vulnerabilities that come with maintaining a separate vector database.
- **Model Serving:** Supports commercial AI models or fully open source models deployed within the organization's infrastructure.



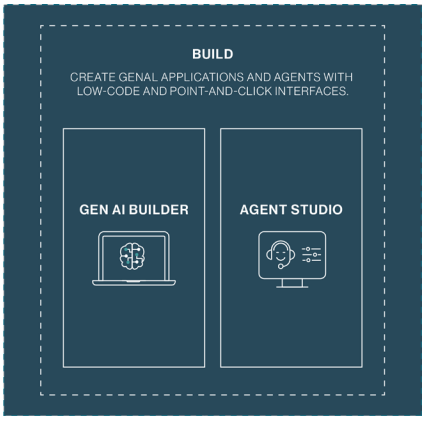
Impact: Enterprises can unify their knowledge base and business data in a single, secure location, enabling semantic similarity search and RAG workflows while preserving regulatory compliance and governance.

Build: Creating AI agents and applications

Purpose: Python SDK and point-and-click tools enable both technical and business teams to rapidly develop GenAI applications and agents

Key capabilities:

- **GenAI Builder:** Create sophisticated GenAI applications, including RAG-based workflows, in hours rather than months.
- **Agent Studio:** Build AI agents that autonomously query data, automate workflows, and generate insights—without requiring coding expertise.
- **Security and observability:** Leverage enterprise-trusted Postgres, robust security features, AI guardrails, and centralized monitoring to ensure compliance and responsible AI use.



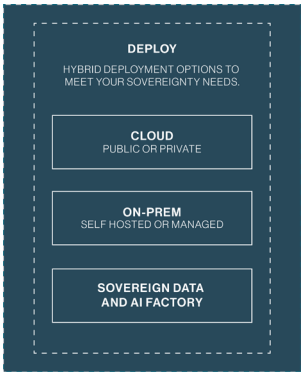
Impact: All teams—from marketing and analytics to operations—can build custom AI solutions that integrate relational and vector data, driving real business outcomes on a secure Postgres foundation.

Deploy: Production-ready AI applications

Purpose: Provides flexible deployment options for AI solutions, ensuring performance, security, and sovereignty

Key capabilities:

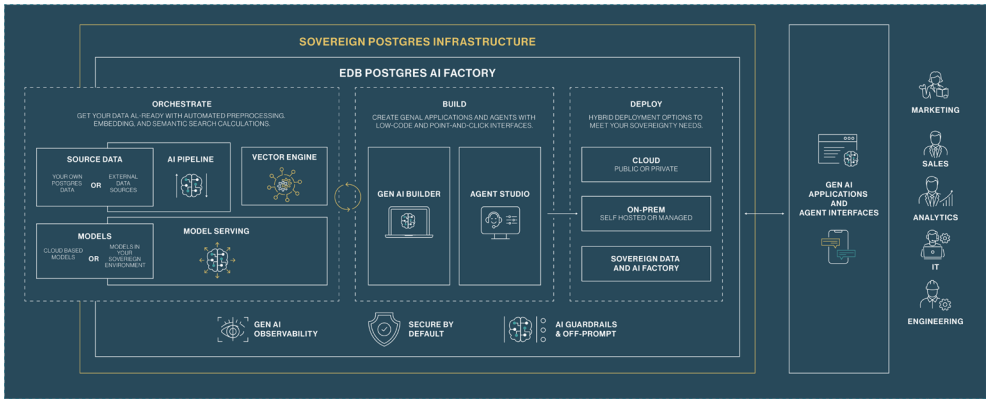
- **Deployment flexibility:** Supports public cloud, private VPCs, on-premises environments, or fully sovereign AI factories with no external infrastructure
- **End-to-end control:** Helps organizations maintain complete control over data, models, and infrastructure
- **Rapid production rollout:** Enables moving from proof-of-concept to production AI in weeks instead of months



Impact: Enterprises can **accelerate time to value**, maintain full governance, and achieve a secure, unified AI infrastructure without compromising compliance or operational control.

4.2. To summarize

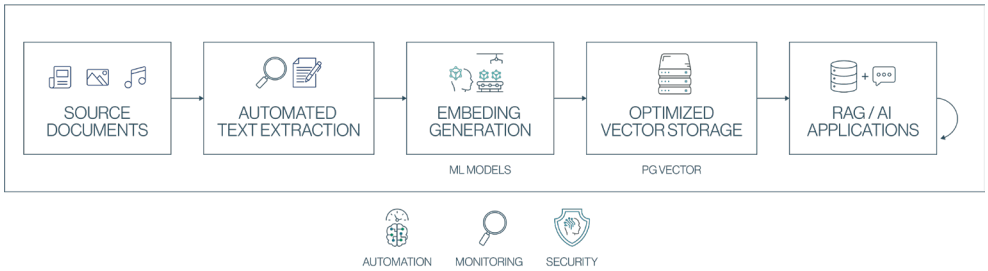
The Orchestrate → Build → Deploy workflow in AI Factory provides a **complete, vector-enabled AI lifecycle** on a single Postgres-native platform. By combining **pgvector embeddings**, AI orchestration, low-code/point-and-click development, and flexible deployment, AI Factory delivers **sovereign, secure, and enterprise-ready AI solutions** that unify business and AI data, accelerate AI adoption, and enable rapid, compliant production of intelligent applications.



4.3. AI Pipelines and RAG pipeline orchestration

AI Factory leverages AI Pipelines to provide automated, production-ready vector workflows. This unified approach directly addresses data synchronization burden and tooling fragmentation problems by minimizing manual development and ensuring enterprise-grade performance and governance. AI Pipelines form the backbone of vector AI solutions, orchestrating every step from document ingestion to intelligent retrieval, embedding, and optimization.

4.3.1. Automated vector workflows



- **Document processing automation:** Supports multiple file formats (PDF, Word, etc.), automatically extracting text, applying intelligent chunking strategies, and preserving metadata to ensure high-quality input for embeddings.
- **Embedding generation and storage:** Seamlessly converts text into vector embeddings using integrated models and populates Postgres vector columns via optimized insertion strategies, maintaining high throughput and data integrity.
- **Vector optimization and indexing:** Applies continuous performance tuning, indexing strategies, and monitoring to ensure low-latency retrieval even as datasets scale, while automatically adapting as new data is added.

Off-prompt context management: Maintains a dedicated knowledge store in TaskMemory, a specialized vector store, sending only small references to the LLM instead of full content, optimizing performance and security. This approach delivers three critical advantages:

- **Cost savings:** Processes content once and stores it in TaskMemory, eliminating redundant token usage for repeated queries. Reduces GPU infrastructure costs by minimizing embedding regeneration and avoids substantial storage expansion typical of separate vector databases by leveraging integrated Postgres vector capabilities.

- **Consistency:** Handles large documents without context window failures, provides granular control over which tools send data directly to LLMs versus storing off-prompt, and enables seamless workflow orchestration across complex multistep processes.
- **Control:** Keeps confidential information, customer records, and intellectual property within customer environments—critical for regulated industries. Offers tool-by-tool privacy controls and simplified compliance auditing.

The table below contrasts the high operational friction of a manual, fragmented approach with the streamlined efficiency provided by AI Pipelines.

Aspect	Without AI Pipelines	With AI Pipelines
Ingestion and sync	Manual ETL/custom scripts required to extract, chunk, vectorize, and load; high risk of data drift	Automated pipeline: Document processing automation ensures continuous, real-time sync with source data via optimized extraction and loading
Development complexity	Requires orchestrating multiple external tools (LangChain, OpenAI API, vector store connector) via custom glue code	Unified workflow: Seamlessly integrates embedding models and indexes into the Postgres engine; no external tooling or custom API plumbing required
Optimization and tuning	Manual index tuning (e.g., HNSW parameters) requires deep knowledge of vector algorithms; no continuous monitoring	Intelligent optimization: Automatically handles vector optimization, indexing strategies, and performance monitoring to ensure low-latency retrieval at scale
Time to production	Extended; time is spent on integration plumbing, security, and debugging across systems	Accelerated: Focus is shifted entirely to application logic and user experience due to automated infrastructure

4.3.2. RAG pipeline orchestration

AI Pipelines also **automates the complexity of RAG workflows**, enabling enterprises to build intelligent applications on top of unified vector and relational data.

- **Retrieval optimization:** Configures similarity search parameters and indexing to maximize semantic search accuracy and performance
- **Context management:** Integrates retrieved documents with language model prompts, ensuring context-aware responses for GenAI applications
- **Knowledge base management:** Maintains dynamic updates, version control, and content freshness while providing comprehensive audit trails for compliance and traceability
- **Standardized APIs:** Offers simplified connectivity to LLMs, automating prompt construction, context injection, and response formatting for end-to-end traceability and consistent AI outputs

4.3.3 Impact

By combining AI Pipelines with RAG orchestration, AI Factory enables organizations to:

- Accelerate AI deployment with **pre-built, automated workflows**.
- Maintain **high performance and scalability** for vector workloads.
- Ensure **security, auditability, and compliance** across AI processes.
- Reduce operational complexity while **integrating seamlessly with enterprise data**.

4.4. GenAI Builder and Agent Studio:
Low- and no-code development platform

Recognizing the hurdles enterprises face—from workflow complexity and fragmented tools to data sovereignty requirements and slow time to market—the EDB PG AI team developed the GenAI Builder and Agent Studio. These low- and no-code tools are designed to accelerate AI application and agent development while addressing the enterprise needs for control, security, and scalability.

4.4.1. How GenAI Builder and Agent Studio address enterprise needs

- **No-code workflow design:** A **visual workflow designer** and **pre-built templates** allow developers, data scientists, and business users to rapidly prototype AI workflows without deep coding expertise. Teams can iterate on concepts such as document summarization, knowledge retrieval, or compliance Q&A, reducing AI project time to market from **28 weeks to 9 weeks**.
- **Multi-model orchestration**: GenAI Builder and Agent Studio abstract the complexity of managing multiple AI models—whether cloud-hosted, on-premises, or within AI Factory.

  Features include:

  - **Dynamic model routing:** Switch models per task or workflow stage. For example, a request categorized as a "simple summary" can be routed to a small, fast, and cost-effective model, while a "complex reasoning" or legal compliance request is automatically routed to a larger, more accurate and expensive model. This optimizes for both performance (speed) and cost per transaction.
  - **Prompt and context management:** Consistent integration of relevant data retrieved from pgvector. This feature ensures that the LLM receives not just raw text but structured, contextual information retrieved by the RAG pipeline. This consistency eliminates manual prompt engineering, drastically reducing the risk of generating inaccurate or hallucinated outputs.
  - **Evaluation hooks:** Capture latency, accuracy, and confidence metrics for governance. These hooks are crucial for validating the reliability of GenAI output. They allow teams to automatically run RAG responses against preset ground truth data or quality checks, ensuring the model is producing accurate, nontoxic, and contextually relevant answers before delivery.
  - **Secure credential handling:** Enforce enterprise security policies for model access. This is vital for compliance and security. Instead of hard-coding API keys in application logic, Builder manages and rotates credentials centrally. This ensures that sensitive keys for external LLMs are securely handled and prevents unauthorized or noncompliant access to AI services.
  - **Model Context Protocol (MCP) integration:** AI Factory acts as an MCP host, providing instant access to hundreds of pre-built integrations across the enterprise ecosystem, including GitHub, Slack, Google Drive, and enterprise databases. This open standard enables agents to connect to external tools and data sources through automated connection management, eliminating the need for custom connector development.

  This orchestration supports EDB's vision of sovereign AI, allowing organizations to control which models process their data and under what conditions.

- **Native pgvector integration:** GenAI Builder integrates directly with pgvector, enabling:
  - Automatic vector type conversions and optimized distance functions
  - Seamless storage and retrieval of embeddings within Postgres schemas

- Real-time monitoring for query latency, accuracy, and explainable retrieval

By keeping vector data within the enterprise's data perimeter, organizations reduce operational complexity and eliminate the need for external vector stores.

- **Multisource data connectivity:** EDB PG AI's data abstraction layer allows AI workflows to access and unify structured, semi-structured, and unstructured data across the enterprise data landscape. This includes connectivity to existing relational databases, data warehouses, object storage, and streaming platforms. This flexibility enables enterprises to create comprehensive data-to-insight pipelines, essential for production-grade AI applications.
- **Rapid prototyping and production deployment:** GenAI Builder accelerates the journey from idea to production:
  - **Prototype fast:** Low-code design and templates allow rapid iteration.
  - **Deploy anywhere:** Workflows can be deployed on-premises, in hybrid environments, or in the cloud.
  - **Lifecycle automation:** Once validated, workflows scale automatically, with monitoring, version control, and production-grade governance.

4.4.2. Business impact

GenAI Builder and Agent Studio empower organizations to build and deploy GenAI and agentic solutions efficiently and securely. The duo enables faster, compliant AI adoption across business, analytics, and operations teams, while keeping all vector data, embeddings, and AI models within a controlled and auditable environment.

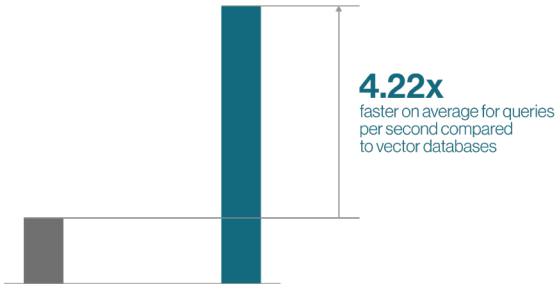
With the architecture and key components of EDB PG AI established, we now analyze how these innovations translate into measurable performance gains. The following section presents benchmarking results demonstrating the platform's superiority in speed, scalability, and efficiency.

5. Performance analysis and benchmarking

EDB PG AI extends the native pgvector capabilities with enterprise-grade optimizations that deliver measurable improvements in performance, scalability, and storage efficiency. Through a combination of advanced indexing algorithms, intelligent caching, and automated lifecycle management, the platform ensures consistent performance even under demanding AI and analytics workloads.

5.1. Query performance characteristics

EDB PG AI delivers substantial performance improvements over basic pgvector implementations through optimized query execution, intelligent indexing, and automated performance tuning. Benchmark testing demonstrates EDB PG AI's performance advantages:



- **Single query performance** shows **4.22x improvement** in average query response time compared to basic Postgres with pgvector, maintaining sub-second response times for millions of vectors.
- **Concurrent query throughput** maintains linear scalability for thousands of simultaneous searches.
- **Complex query operations**, including filtered vector searches and hybrid queries, demonstrate consistent performance.

5.2. Storage efficiency and cost analysis

EDB PG AI platform achieves up to **18x cost efficiency over specialized vector databases** by leveraging intelligent compression, optimized data layouts, and object storage integrations:

- **Intelligent compression:** Optimized compression techniques typically reduce raw vector storage by **60%–80%** without sacrificing query accuracy.
- **Index size reduction:** The platform minimizes index storage through quantization. HNSW indexes using scalar quantization (halfvec) reduce size by approximately 50%, while binary quantization can shrink index size by up to 90% for compatible workloads—all while maintaining or improving query performance.
- **Cost-effective object storage integration:** The platform intelligently separates the compressed **vector index** (the "hot" data used for fast search) onto high-speed block storage, while decoupling the raw vector data (the "cold" data) onto highly available, low-cost **object storage** (e.g., S3, Azure Blob, MinIO, local file system).
- **Reduced operational expenditure:** This decoupling eliminates the industry requirement of mandating expensive solid-state drive (SSD) storage for all vector data, drastically reducing operational expenditure and maximizing cost efficiency for large-scale deployments.
- **Predictable scalability:** By integrating with elastic object storage, the platform ensures that storage costs grow linearly with data volume, providing predictable, high-volume scalability unmatched by single-node or monolithic vector databases.

5.3. Scalability testing results

Scalability testing confirms the platform's capacity to support enterprise deployments with datasets of up to ***50 million high-dimensional vectors***.

- **Horizontal scaling** across multiple Postgres instances sustains near-linear performance growth as data volume increases, while automated load balancing optimizes resource distribution across the cluster.
- **Index build performance** benefits from parallel processing, demonstrating efficient scale-up with increases in data volume and cluster size. The platform **dynamically selects optimal parallelization strategies** based on available system resources and data characteristics.
- **Operational scaling** includes fully **automated backup and recovery procedures** that adapt to growing data volumes, monitoring systems providing consistent visibility across distributed deployments, and maintenance operations designed to minimize production impact.

5.4. Reliability and availability metrics

Enterprise-grade deployments require **99.999% availability**, supported by automated failover and robust disaster recovery capabilities.

- **High-availability (HA) architecture** is designed for zero data loss, leveraging synchronous replication between primary and standby instances. **Failover operations typically complete in under 30 seconds**, ensuring seamless continuity with minimal service interruption.
- **Disaster recovery workflows** enable **point-in-time (PIT) restoration** of both vector data and associated indexes. **Automated backup** processes incorporate vector-specific safeguards, including index consistency verification and embedding lifecycle integrity checks. Recovery validation procedures confirm the ability to fully restore vector databases along with all metadata, configuration states, and performance parameters.
- **Real-time monitoring and Alerting frameworks** provide end-to-end operational visibility for vector workloads. Metrics collected include **query performance trends, index health indicators, and capacity utilization analytics**. Automated alerts drive proactive intervention, mitigating risks such as performance degradation or resource saturation before they affect production environments.

EDB PG AI's performance optimizations are only as valuable as their ease of adoption. The next section provides a step-by-step guide to implementing pgvector inside **AI Factory**, enabling teams to start building intelligent, vector-powered applications immediately.

6. Implementation guide

6.1. Getting started with pgvector in AI Factory

Step 1: Launch your AI Factory environment

1. Log in to your AI Factory account.
2. From the dashboard, select Create New Project → AI Workload.
3. Pick your preferred cluster (e.g., EDB Postgres Advanced Server or EDB Postgres Extended).
4. Enable pgvector during setup by checking AI Extensions → pgvector.

✅Tip: You can also add pgvector later using:

```
CREATE EXTENSION IF NOT EXISTS vector;
```

Step 2: Create your first vector table

Define a schema to store text embeddings. For example:

```
CREATE TABLE documents (
     id SERIAL PRIMARY KEY,
      content TEXT,
     embedding VECTOR(1536)
);
```

This structure stores raw content alongside its AI embedding vector.

Step 3: Generate and insert embeddings

You can create embeddings using OpenAI, Hugging Face, or NVIDIA NeMo Retriever models integrated with AI Factory. Example Python snippet:

```
import psycopg2
import openai
conn = psycopg2.connect("dbname=ai_factory user=postgres
password=secret")
cur = conn.cursor()
text = "EDB Postgres AI accelerates enterprise AI adoption."
embedding = openai.Embedding.create(input=text, model="text-embedding-
3-small")["data"][0]["embedding"]
cur.execute("INSERT INTO documents (content, embedding) VALUES (%s,
%s)", (text, embedding))
conn.commit()
```

Step 4: Query using vector similarity

Retrieve semantically similar content using cosine similarity:

```
SELECT id, content
FROM documents
ORDER BY embedding <-> '[vector values here]'
LIMIT 3;
```

This enables powerful use cases such as semantic search, context retrieval, and intelligent recommendations.

Step 5: Monitor and optimize performance

With AI Factory, you get:

- **2x–3x throughput improvements** for data embedding tasks
- **1.5x–2x faster retrieval performance**
- **Software-defined optimization** via NVIDIA AI microservices (NIM, NeMo Retriever)

Use the **performance dashboard** in AI Factory to view latency, query times, and resource utilization in real time.

Step 6: Scale and integrate

- **Deploy in production:** Self-hosted on Kubernetes or fully managed with EDB PG AI Hybrid Manager.
- **Integrate** with LLMs through REST APIs or LangChain connectors.
- **Maintain sovereignty** by keeping your data and vectors secure within your infrastructure.

Step 7: Explore next steps

- Try the **hands-on labs** in AI Factory.
- Experiment with **hybrid search (Full-text + Vector)**.
- Explore the **NVIDIA NIM + pgvector integration** for advanced RAG use cases.

6.2. To summarize

By combining **pgvector** with **AI Factory**, you get an end-to-end environment for building sovereign, high-performance, and scalable GenAI applications—right inside Postgres.

AI Factory isn't just Postgres with vectors—it's a complete software-optimized path to AI production.

Once deployed, AI applications must operate within a secure, compliant, and governable framework. The next section outlines how EDB PG AI ensures enterprise-grade data sovereignty, access control, and operational governance across all environments.

7. Enterprise governance, security, and unified AI architecture in EDB PG AI

EDB PG AI redefines enterprise readiness for vector and AI workloads by merging data sovereignty, compliance, and security with a unified data and AI platform built directly on Postgres. It eliminates the silos and limitations of standalone vector databases through an open, software-optimized foundation that combines transactional integrity, analytical scalability, and vector intelligence—all within a single, governed platform.

7.1. Data sovereignty and governance

EDB PG AI ensures that all data, embeddings, and model interactions remain fully within the enterprise's sovereign environment. Vector and relational workloads operate in the same Postgres instance—on-premises, in the cloud, or in hybrid configurations—providing complete ownership and control.

Comprehensive **audit trails** capture user queries, embedding operations, and AI inference, ensuring transparency, compliance, and traceability across the full AI lifecycle.

This sovereign design allows enterprises to meet stringent frameworks such as **GDPR**, **HIPAA**, **PDPA**, and **RBI/DPDP (India)** while maintaining freedom from proprietary, vendor-controlled infrastructures.

7.2. Open standards and freedom from vendor lock-in

Built entirely on open Postgres standards, EDB PG AI's architecture avoids the pitfalls of closed ecosystems. Embeddings are **queryable via SQL, interoperable with the wider Postgres ecosystem, and portable across environments**—cloud, hybrid, or on-premises.

This flexibility ensures long-term scalability and independence, allowing organizations to migrate or federate vector workloads without reengineering pipelines or compromising compliance.

7.3. Security and access control

EDB PG AI delivers **multi-layered enterprise security** spanning data protection, identity management, and network isolation.

- **Data protection:** AES-256 encryption at rest and TLS 1.3 in transit safeguard relational and vector data alike, integrating with enterprise key management systems (KMS) and hardware security modules (HSM) for secure key management.
- **Identity and access:** RBAC, fine-grained permissions, and multifactor authentication (MFA) for sensitive operations ensure that only authorized users can interact with AI pipelines or vector data.
- **Deployment security:** Secure by default, the platform supports VPC isolation, service mesh–based interservice encryption, and preconfigured firewall templates for consistent protection across all environments.

Unlike standalone vector stores that require dual security layers, EDB offers a unified SQL and vector security model, enabling secure hybrid queries such as:

```
SELECT id, description
FROM documents
ORDER BY embedding <-> $1
LIMIT 5;
```

This single governance framework ensures compliance without adding operational complexity.

7.4. Postgres-native vector integration

At the core of EDB PG AI is pgvector, embedded natively within Postgres. This eliminates external vector databases and API layers, enabling direct SQL access to embeddings stored alongside relational data.

- **Shared storage engine:** Embeddings and structured data share a single schema for real-time hybrid analysis.
- **Unified query planner:** Postgres optimizes vector and scalar operations together for consistent performance.
- **Transactional consistency:** ACID compliance ensures synchronization between vectors and base data—critical for accuracy and recoverability.

This integrated approach minimizes data movement, simplifies architecture, and ensures end-to-end consistency across the AI stack.

7.5. Unified platform for all workloads

Unlike purpose-built vector engines that introduce additional silos, EDB PG AI **unifies OLTP, OLAP, and AI/ML workloads** within one platform.

Enterprises can process transactions, run analytics, and perform semantic search using the same database—reducing integration overhead, latency, and total cost of ownership.

This convergence delivers faster insights and simpler governance for real-time, AI-driven business operations.

7.6. Deployment flexibility and observability

EDB PG AI's **Kubernetes-native architecture** ensures consistent deployment across **cloud**, **on-premises**, **hybrid**, and **edge** environments. Its **Hybrid Manager** centralizes orchestration and monitoring, providing unified visibility into distributed workloads.

Performance metrics, index health, and embedding lifecycle data are tracked in real time, with automated index optimization, cleanup, and alerting to maintain peak performance and uptime.

These operational insights enable teams to scale securely while maintaining compliance and performance consistency across environments.

7.7. Differentiation beyond industry standards

EDB PG AI extends beyond traditional database functionality through a combination of open architecture, enterprise hardening, and AI lifecycle integration via **AI Factory**.

Differentiator	EDB Advantage	Why It Matters
Postgres-native vector engine	Built directly into Postgres via pgvector—no separate vector store or API layer	Simplifies architecture and ensures ACID compliance across workloads
Unified structured + unstructured data	SQL joins across vector and relational data	Enables hybrid AI use cases such as RAG + BI
Sovereign and hybrid deployments	Identical functionality across on-prem, cloud, or hybrid	Meets data residency and compliance mandates
AI Factory integration	Embedding generation, retriever creation, and model orchestration	End-to-end AI lifecycle in one ecosystem
External data access for unified analytics	Query data across different formats and locations without ETL	Enables cross-domain AI analytics
Enterprise-grade security	Encryption, RBAC, audit logging, and compliance alignment	Trusted for sensitive workloads in BFSI, healthcare, and government

7.8. The hardware-optimized path to AI production

The collaboration between EDB PG AI and Supermicro delivers a hardware-optimized foundation for building and deploying intelligent, high-performance GenAI applications at scale while maintaining complete data sovereignty. High-performance, high-efficiency Server Building Block Solutions combined with expert database tuning create an optimized environment for enterprise AI workloads that remain fully within organizational control—on premises, in hybrid environments, or at the edge.

Together, EDB PG AI and Supermicro are redefining the performance and value baseline for enterprise AI infrastructure:

- **6x transaction throughput** at peak level versus out-of-the-box community Postgres installations
- **Up to 90% better price/performance** compared to running EDB on Amazon EC2 (on-demand pricing)
- **Up to 83% better price/performance** compared to Amazon EC2 with three-year savings plans[5]

This hardware-optimized approach eliminates the traditional cost and complexity barriers of AI infrastructure. By combining enterprise-grade server hardware with a unified transactional, analytical, and vector data engine, organizations achieve superior performance while maintaining complete control over infrastructure, compliance, and data sovereignty.

As strategic partners, EDB and Supermicro work together to ensure that customers can deploy EDB Postgres AI on optimally configured hardware, enabling them to harness the full power of Postgres for mission-critical GenAI applications—whether deployed on-premises, in hybrid environments, or at the edge.

> *"Running EDB Postgres AI on Supermicro hardware enables enterprises to break free from cloud vendor lock-in while achieving superior performance, value, and faster time to production. This partnership delivers the sovereignty and control enterprises demand without sacrificing the performance their GenAI applications require."*
>
> ***—Nancy Hensley, Chief Product Officer, EDB***

7.9. To summarize

EDB PG AI delivers more than secure and compliant data infrastructure—it provides a **software-optimized**, **Postgres-native foundation** for building and operationalizing AI at enterprise scale. By unifying transactional, analytical, and vector workloads within a single governed environment, organizations gain the agility to innovate without sacrificing control, compliance, or performance.

With this unified architecture, enterprises can move confidently from **AI experimentation to production**, accelerating deployment cycles while maintaining sovereignty over data, models, and infrastructure.

The following section, **From capabilities to impact: Real-world use cases**, demonstrates how EDB PG AI translates these architectural advantages into measurable business outcomes—powering real-time intelligence, automation, and innovation across industries such as financial services, healthcare, and manufacturing.

8. From capabilities to impact: Real-world use cases

Building on the unified foundation described above, EDB PG AI translates technical innovation into measurable business outcomes. By combining transactional integrity, analytical scalability, and vector intelligence within a single platform, organizations can operationalize GenAI applications where their data already resides, securely and at scale. The following use cases illustrate how this vector-enabled architecture enables production RAG applications, AI agents, and conversational AI systems across financial services, healthcare, manufacturing, and beyond.

8.1. Industry-specific use cases

The following use cases illustrate how organizations across financial services, healthcare, and manufacturing are leveraging EDB PG AI to deploy RAG applications, AI agents, and conversational AI systems that combine the power of LLMs with enterprise data while maintaining data sovereignty, regulatory compliance, and operational efficiency.

- **Financial services:** EDB PG AI supports financial institutions in handling the unique demands of banking, insurance, and fintech: high transaction volumes, stringent compliance, high availability, latency-sensitive analytics, and risk management.

  Scenarios:

  - **Intelligent document processing:** For RAG-based systems that analyze loan documents, contracts, and regulatory filings, answer questions such as, "What are the risk factors in this mortgage portfolio?" by combining structured loan data with unstructured document analysis.
  - **Conversational banking assistants:** AI agents built with GenAI Builder help customers with account inquiries, transaction disputes, and financial planning—all while maintaining PCI compliance and data sovereignty.
  - **Regulatory compliance automation:** AI agents continuously monitor transactions and communications against regulatory requirements, generating compliance reports by querying both structured transaction data and unstructured communications.

- **Healthcare and life sciences:** Healthcare and life sciences organizations face huge volumes of structured and unstructured data (clinical records, trial data, imaging) under high regulatory burden (HIPAA, data residency requirements). EDB PG AI provides a platform capable of handling relational and vector workloads in a compliant, secure manner.

  Scenarios:

  - **Clinical AI assistants:** RAG applications help physicians by retrieving relevant patient history, recent research, and treatment protocols—combining EHR data with medical literature embeddings.
  - **Patient care coordination:** AI agents automate care plan creation by analyzing patient records, lab results, and clinical notes—all within HIPAA-compliant infrastructure.
  - **Medical research accelerator:** GenAI Builder workflows enable researchers to query across clinical trials, genomic data, and published literature without moving sensitive data outside enterprise boundaries.

- **Manufacturing and supply chain**: Manufacturers rely on predictive maintenance, logistics optimization, sensor analysis, and quality evaluation—each requiring the integration of structured data with unstructured sources such as images, logs, and IoT feeds. EDB PG AI powers these vector-driven use cases across a unified data environment.

  Scenarios:

  - **Maintenance knowledge assistants:** RAG systems help technicians troubleshoot equipment by querying maintenance manuals, sensor data, and historical repair logs in natural language.
  - **Supply chain intelligence:** AI agents monitor supplier communications, logistics data, and market signals to predict disruptions and recommend alternatives.
  - **Quality assurance copilots:** GenAI applications assist quality inspectors by analyzing defect images, comparing against historical patterns, and suggesting root causes.

8.2. Other cross-sector and high-level use cases

Beyond those sector-specific examples, here are additional use cases supported by AI Factory:

- **Cognitive AI (semantic search and recommendation):** Organizations embed unstructured content (documents, images, logs) as vectors and run semantic search, recommendations, and RAG inside Postgres.
- **Virtual expert/conversational AI agents:** Use AI Factory's tools (e.g., GenAI Builder, Agent Studio) to build natural-language interfaces and knowledge-base assistants with access to vector indexes of enterprise content.
- **Agentic analytics:** AI agents autonomously query business data and generate insights using natural language. Analysts ask, "Which product lines showed margin compression and why?" and agents combine structured data with unstructured reports (stored as vector embeddings) to deliver contextual answers. Agents proactively monitor key performance indicators, detect anomalies, and alert stakeholders—democratizing data access without SQL expertise.
- **Agentic microservice management:** AI agents monitor microservice health, logs, and dependencies in real time, automatically correlating issues across distributed systems. When incidents occur, agents trace error patterns, query historical incident embeddings, and recommend remediation. Engineers ask, "Why is the payment service slow?" and receive root cause analysis with resolution suggestions—reducing mean time to resolution.

From foundational architecture to production deployment, EDB PG AI delivers the performance, governance, and flexibility needed for enterprise AI success.

9. Conclusion

The future of data and AI is already here.

The convergence of AI and data infrastructure represents the most significant shift in enterprise technology since the internet. Organizations that successfully integrate vector capabilities into their data platforms will define the next decade of innovation.

EDB PG AI makes this transformation accessible today—not through incremental improvements but through a fundamental reimagining of what a database can be. By unifying transactional, analytical, and AI workloads on a single platform, we eliminate the complexity that has held enterprises back from realizing AI's full potential.

The numbers speak for themselves:

- **4.22x faster query performance** via optimized indexing and intelligent query orchestration.[1]
- **18x greater storage efficiency** through advanced compression and optimization.[2]
- **67% reduction in development complexity** via automated AI pipelines and low-code GenAI Builder.[3]

But beyond the metrics lies a more profound truth: AI is not a separate workload—it's an integral part of every modern application. EDB PG AI recognizes this reality and provides the foundation for a future in which every query is intelligent, every application is adaptive, and every decision is informed by the full context of your data.

Unlock the power of intelligent data today

Discover how EDB PG AI can transform your data infrastructure; unify transactional, analytical, and AI workloads; and accelerate innovation. Start building applications in which every query is intelligent, every decision is data driven, and AI is seamlessly embedded at scale.

Explore EDB PG AI: Visit enterprisedb.com to learn more about:

- EDB Postgres AI Platform | Read an Overview
- AI Factory | Use Cases
- Explore EDB PG AI Demos
- Talk to an Expert Today

10. References

1. EnterpriseDB, *Solving Enterprise Generative AI and Analytics Challenges: Zooming into Our Q4 2024 Release*, December 2024.

2. EnterpriseDB Internal Benchmark Report, *EDB Postgres AI Object-Storage Integration and Compression Efficiency Tests*, conducted Q1 2025, internal testing data on file with EnterpriseDB.

3. McKnight Consulting Group, *EDB Postgres AI Significantly Outperforms MySQL, SQL Server, MongoDB, and demonstrates TCO Savings Against AWS Stack in New Benchmark Study*, February 19, 2025.

4. Shubham Singh, *Latest ChatGPT Users Stats 2025 (Growth & Usage Report)*, demandsage.com, October 7, 2025.

5. EnterpriseDB–Supermicro Benchmark Report, EDB Postgres AI and Supermicro Unlock 6x Higher Returns on Your Database Investment, October 2024.

EDB POSTGRES AI

EDB Postgres AI is the first open, enterprise-grade sovereign data and AI platform, with a secure, compliant, and fully scalable environment, on premises and across clouds. Supported by a global partner network, EDB Postgres AI unifies transactional, analytical, and AI workloads, enabling organizations to operationalize their data and LLMs where, when, and how they need them. For more information, visit www.enterprisedb.com