yugabyteDB

SOLUTION BRIEF

A Practical Guide to Building GenAI Apps on a PostgreSQL-Compatible Database

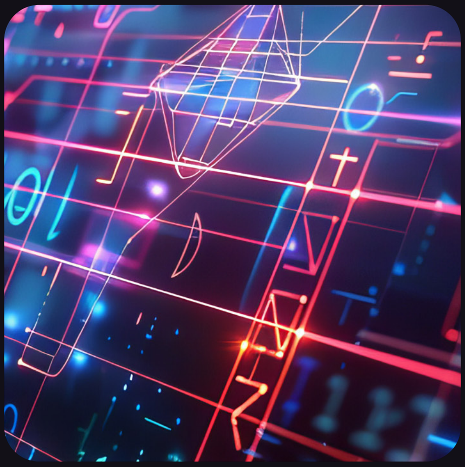
Table of Contents

In this solution brief you will discover:

- ↗ How YugabyteDB's powerful vector indexing capabilities support vector search through the pgvector extension
- ↗ The unique distributed architecture of YugabyteDB
- ↗ How YugabyteDB's distributed design enhances scalability and performance for AI workloads

This guide shares basic AI concepts, architectural considerations, and access to hands-on tutorials that demonstrate how to build your first GenAI application on various platforms.

An Introduction to Vector Embeddings



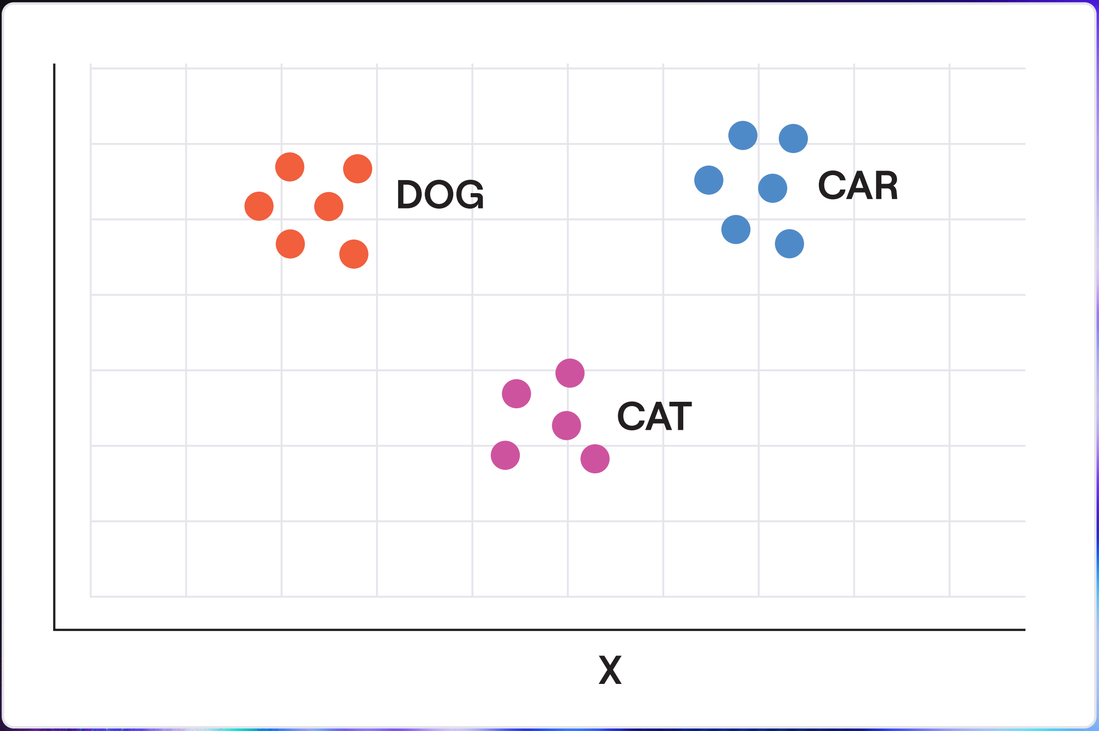
Vector search is crucial for AI applications. It enables efficient similarity searches in high-dimensional data, which is common in generative AI models.

Vector embeddings transform unstructured data—like text, images, or audio—into high-dimensional numeric arrays that capture semantic meaning. This allows systems to perform operations based on meaning, not just exact matches.

Vector embeddings are foundational to modern applications that require semantic search, recommendations, and generative AI.

Explore ultra-resilient, scalable vector search with YugabyteDB

Watch Now >



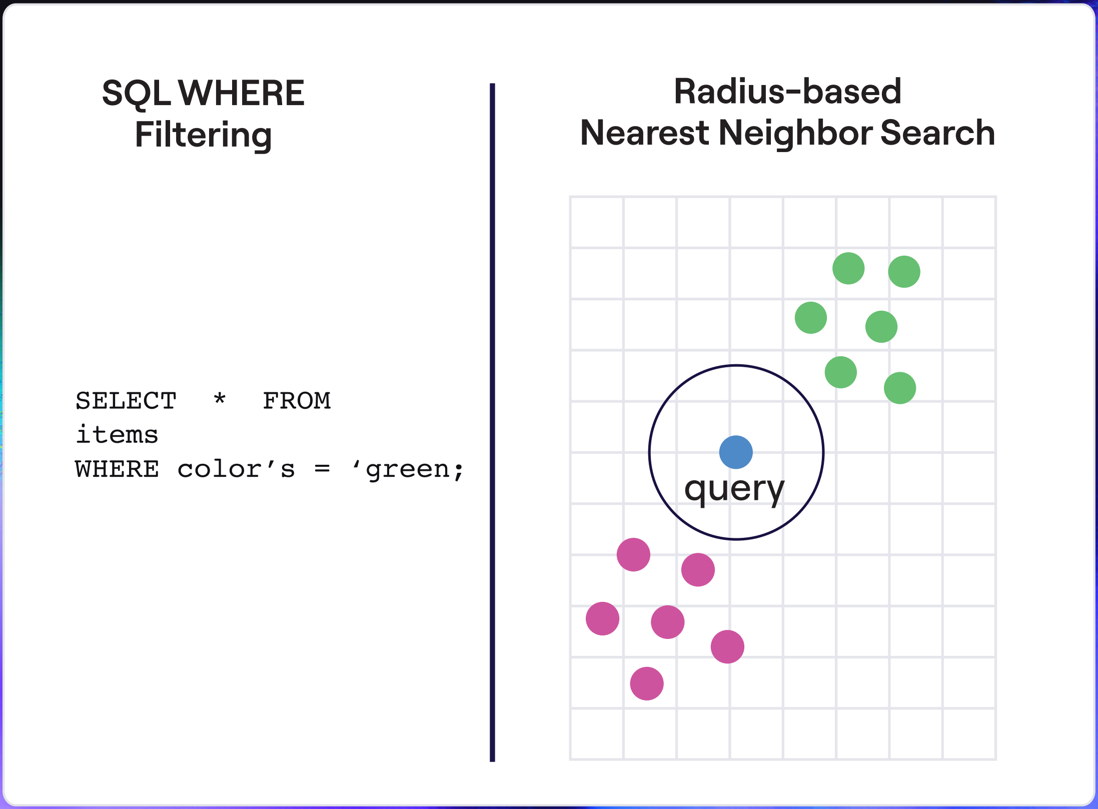
Vector Databases vs Traditional Databases

Traditional databases excel at exact-match lookups using structured filters. By contrast, vector databases enable similarity searches over embeddings, finding items that are close in meaning without matching exact words or values. This unlocks use cases like natural language search and personalized recommendations.

The pgvector PostgreSQL extension provides vector similarity search capabilities (without the need for a dedicated vector database) and allows you to store and query vectors for performing similarity searches.

Learn more about the pg_vector extension

Read More



```
SELECT  *  FROM
items
WHERE color's = 'green;
```

Similarity Searches and Nearest Neighbors

Similarity searches identify vectors closest to a query vector, using distance functions such as cosine similarity or Euclidean distance (L2).

Approximate nearest neighbor (ANN) algorithms like HNSW make this process scalable to billions of vectors, while maintaining sub-second query times.

PostgreSQL pgvector: getting started and scaling

Read More

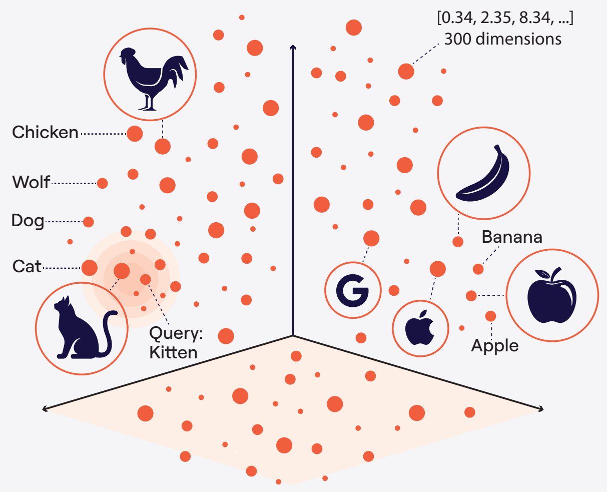
Explore similarity search using Google Vertex AI

Learn How

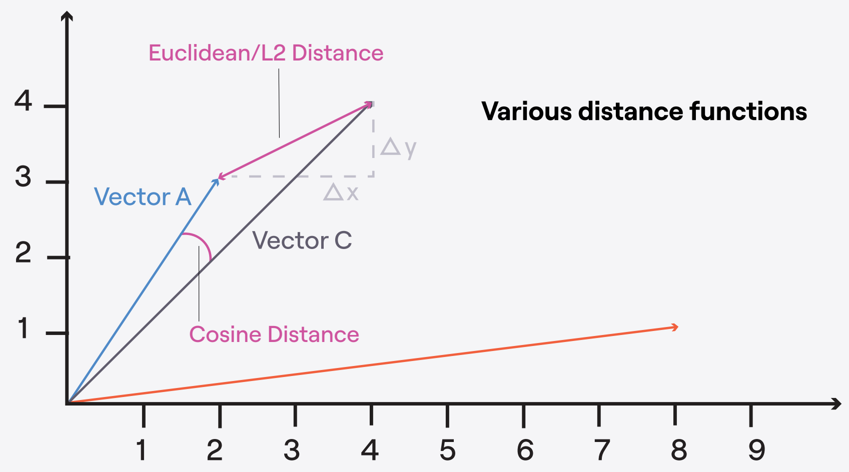
Explore similarity search using Azure OpenAI

Learn How

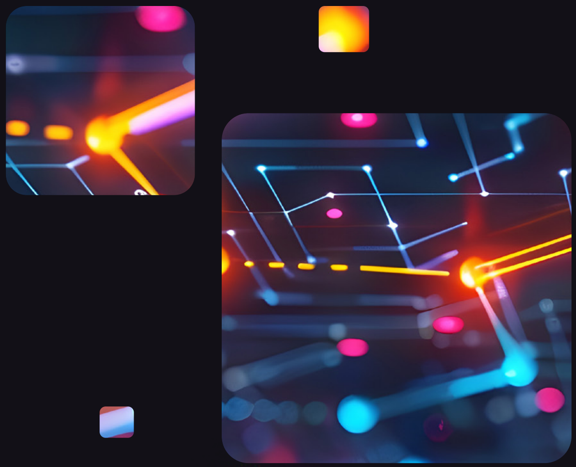
Vector Search



Various distance functions



Vector Embeddings in Practice



The typical development lifecycle for GenAI applications starts by passing, at data load time, a collection of data (documents, images, sound clips, etc.) through an embedding model (like OpenAI, Ollama, or Hugging Face), which outputs a numerical vector for each item.

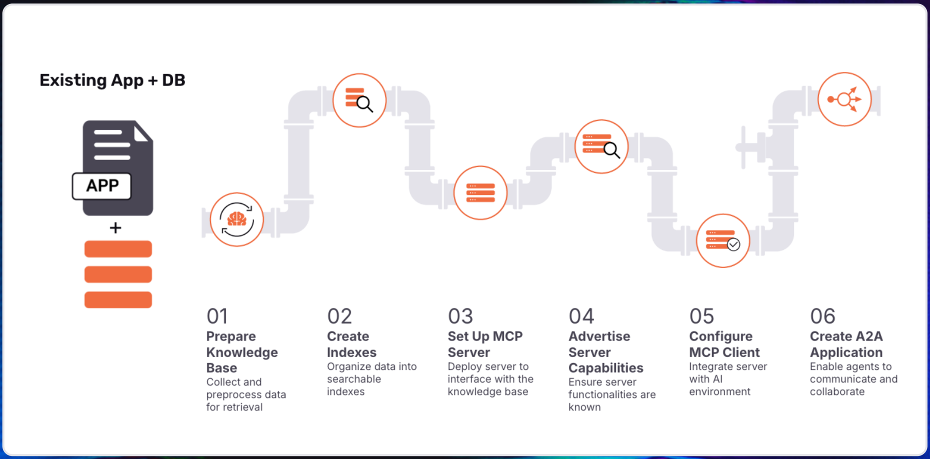
These vectors are stored in a database with related metadata (like date, type, category). Later, when a result is needed at runtime, data items most relevant to a user's query are efficiently retrieved and passed to an LLM for use in formulating a response.

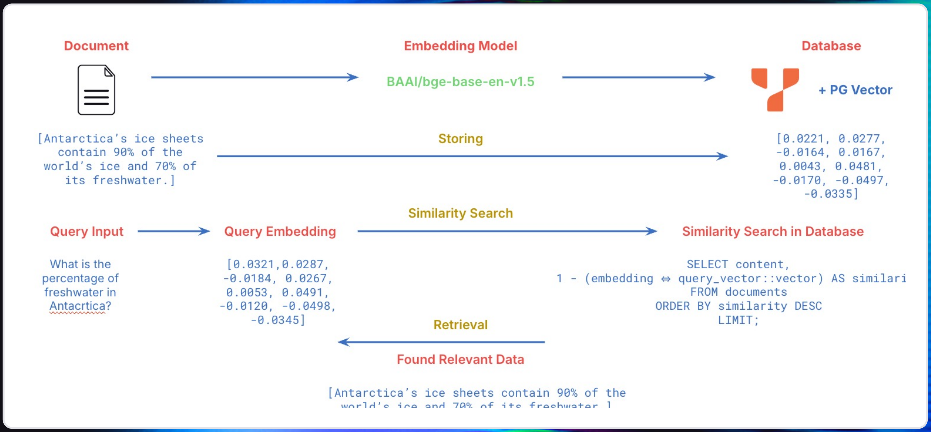
Use Ollama to generate text embeddings

Learn How >

Use LocalAI to create an interactive query interface

Learn How >


Existing App + DB
APP
+
01
Prepare Knowledge Base
Collect and preprocess data for retrieval
02
Create Indexes
Organize data into searchable indexes
03
Set Up MCP Server
Deploy server to interface with the knowledge base
04
Advertise Server Capabilities
Ensure server functionalities are known
05
Configure MCP Client
Integrate server with AI environment
06
Create A2A Application
Enable agents to communicate and collaborate



```
[Antarctica's ice sheets
contain 90% of the
world's ice and 70% of
its freshwater.]
```

```
[0.0221, 0.0277,
-0.0164, 0.0167,
0.0043, 0.0481,
-0.0170, -0.0497,
-0.0335]
```

What is the percentage of freshwater in Antacrtica?

```
[0.0321,0.0287,
-0.0184, 0.0267,
0.0053, 0.0491,
-0.0120, -0.0498,
-0.0345]
```

```
SELECT content,
1 - (embedding ⇔ query_vector::vector) AS similari
FROM documents
ORDER BY similarity DESC
LIMIT;
```

```
[Antarctica's ice sheets contain 90% of the
```

Introducing YugabyteDB

Traditional databases struggle with scaling vector workloads, making distributed databases like YugabyteDB essential.

PostgreSQL-compatible distributed YugabyteDB delivers strong consistency, ultra-resilience, and linear scalability. It seamlessly blends relational and vector workloads, making it a powerful foundation for AI-driven applications that require both traditional transactional operations and similarity search.

Explore YugabyteDB's Vector indexing architecture

Read More

Disadvantages of a Standalone Database

- API-based, prone to desync
- May offer proprietary indexing algorithms
- Lack full database features (ACID compliance, row-level security)
- Faster query times, but with potential network latency
- More cost-effective for smaller, proof-of-concept (POC) projects
- Requires additional effort for data synchronization and management

Advantages of Postgres as a Vector Database

- Built-in PostgreSQL data sync
- Supports multiple indexing algorithms (e.g., IVFFlat, HNSW)
- PostgreSQL's full ACID backup, and security
- Higher accuracy and often outperforms in QPS, especially with HNSW indexing
- More economical at scale, leverages existing PostgreSQL infrastructure
- Seamless integration with PostgreSQL data and familiar tools

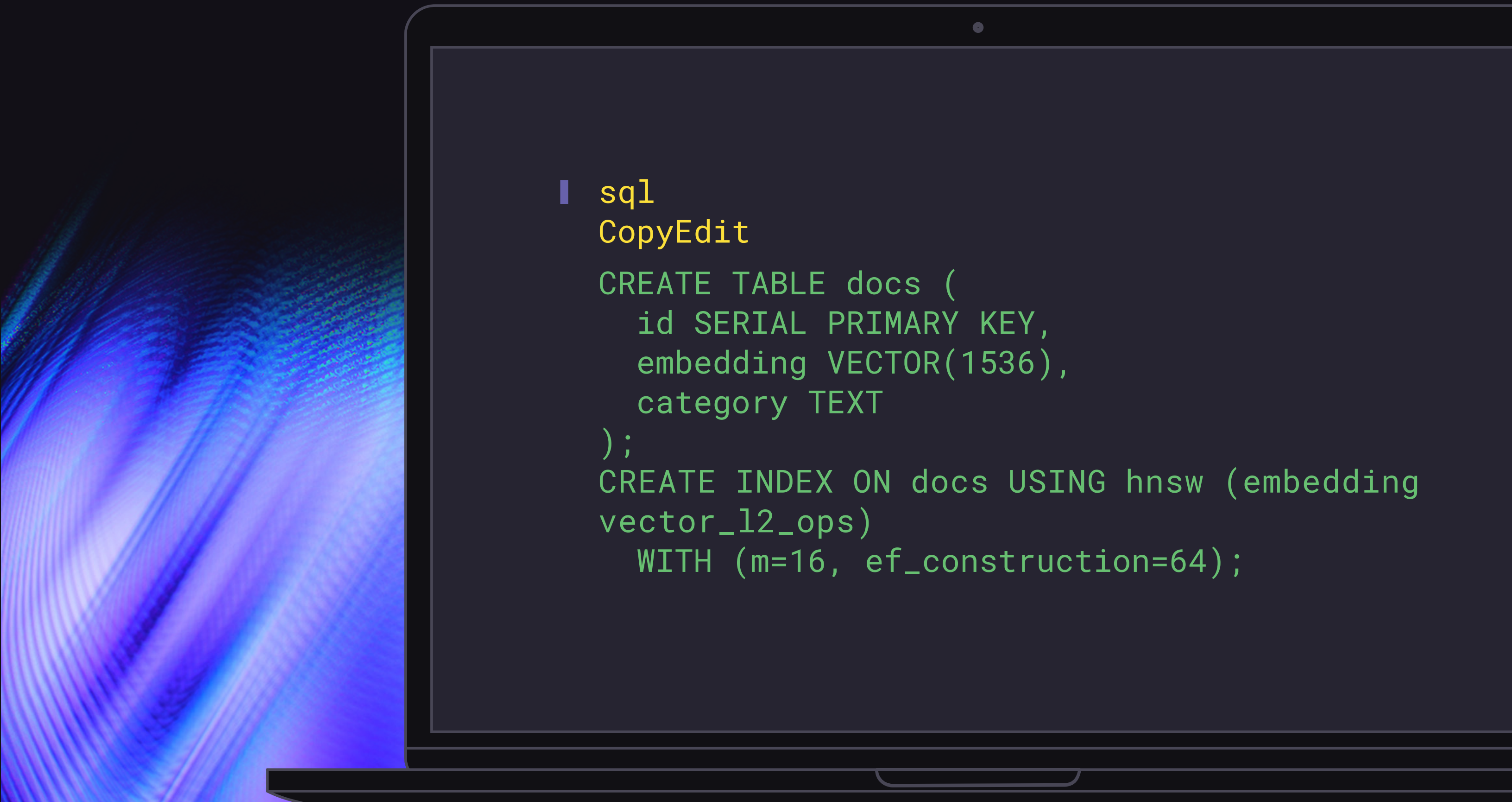
The pgvector Extension in YugabyteDB

YugabyteDB integrates the popular pgvector extension, enabling developers to store and index vectors directly in SQL tables.

Yugabyte provides an extensible indexing framework designed to support the seamless integration of state-of-the-art vector indexing libraries and algorithms, augmenting the capabilities offered by pgvector.
This allows you to create HNSW indexes on vector columns and combine similarity scoring with WHERE filters for rich hybrid queries.

Discover YugabyteDB's extensible Vector search

Read More



```
sql
CopyEdit
CREATE TABLE docs (
  id SERIAL PRIMARY KEY,
  embedding VECTOR(1536),
  category TEXT
);
CREATE INDEX ON docs USING hnsw (embedding
vector_l2_ops)
  WITH (m=16, ef_construction=64);
```

YugabyteDB's Distributed Vector Engine

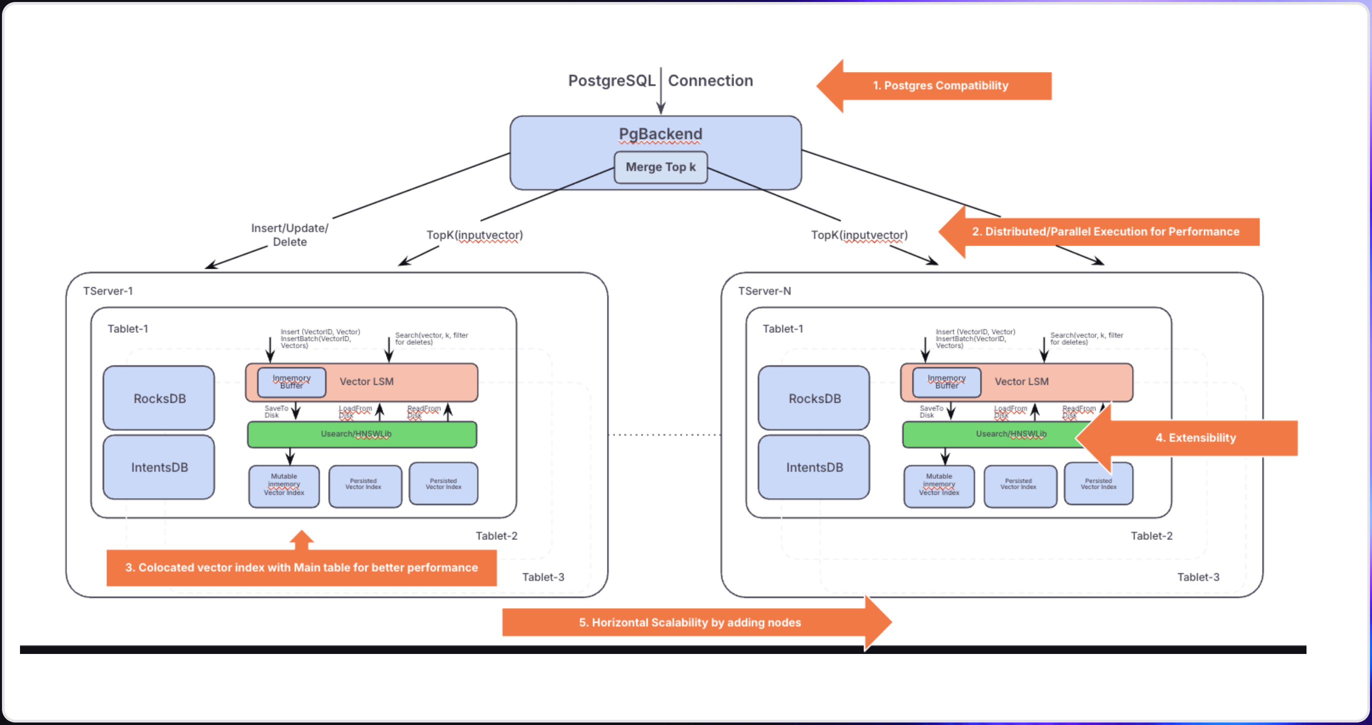
YugabyteDB leverages USearch for approximate nearest neighbor search alongside a multi-level storage engine (LSM) that keeps recent data in memory and moves older data to disk.

This architecture provides ultra-low latency for fresh data, massive scale, and resilience through replication.

YugabyteDB features an extensible indexing framework that supports the integration of state-of-the-art vector indexing libraries like USearch, HNSWLib, and Faiss. This ensures that applications can adapt to evolving vector search requirements and incorporate the latest algorithms.

Learn how to architect apps for ultra-resilience with YugabyteDB

Read More



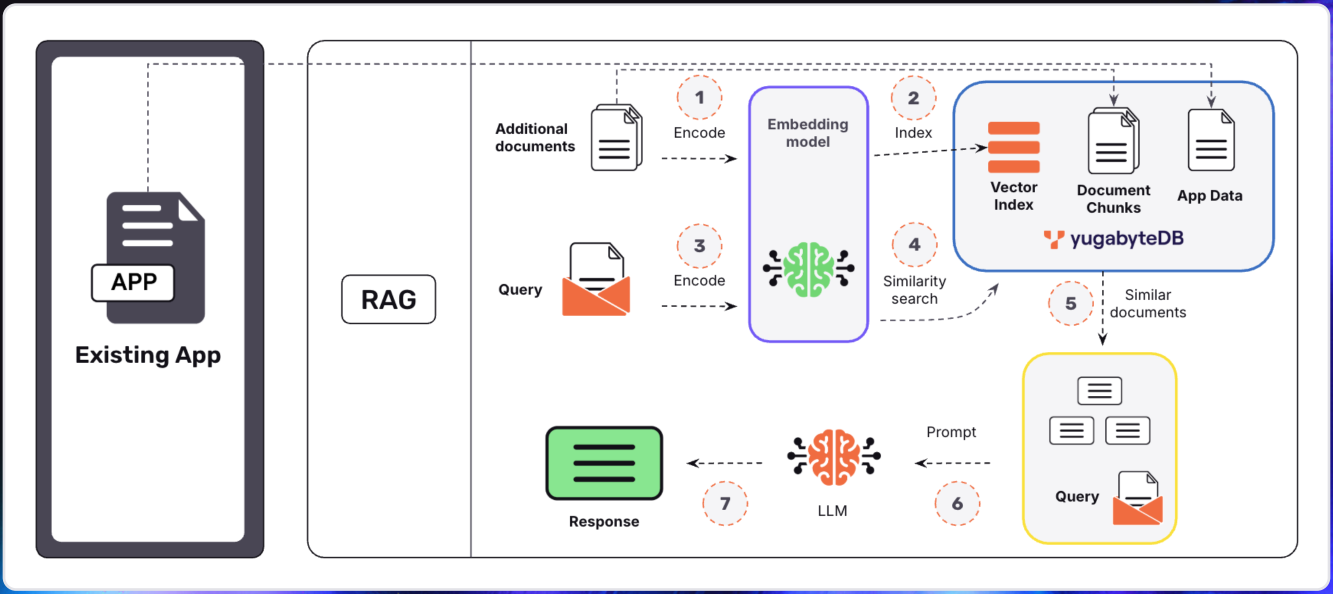
Retrieval-Augmented Generation (RAG)

RAG combines large language models (LLM) with external knowledge sources to produce more accurate and context-aware responses.

In RAG architectures, embeddings stored in YugabyteDB are retrieved based on similarity to an input query, then fed into an LLM to produce context-aware answers. This approach is essential for building AI copilots and enterprise search solutions that produce answers based on your private data.

Build a Retrieval-Augmented Generation (RAG) pipeline with YugabyteDB

Learn How >



Accelerate Vector Search with YugabyteDB's MCP Server

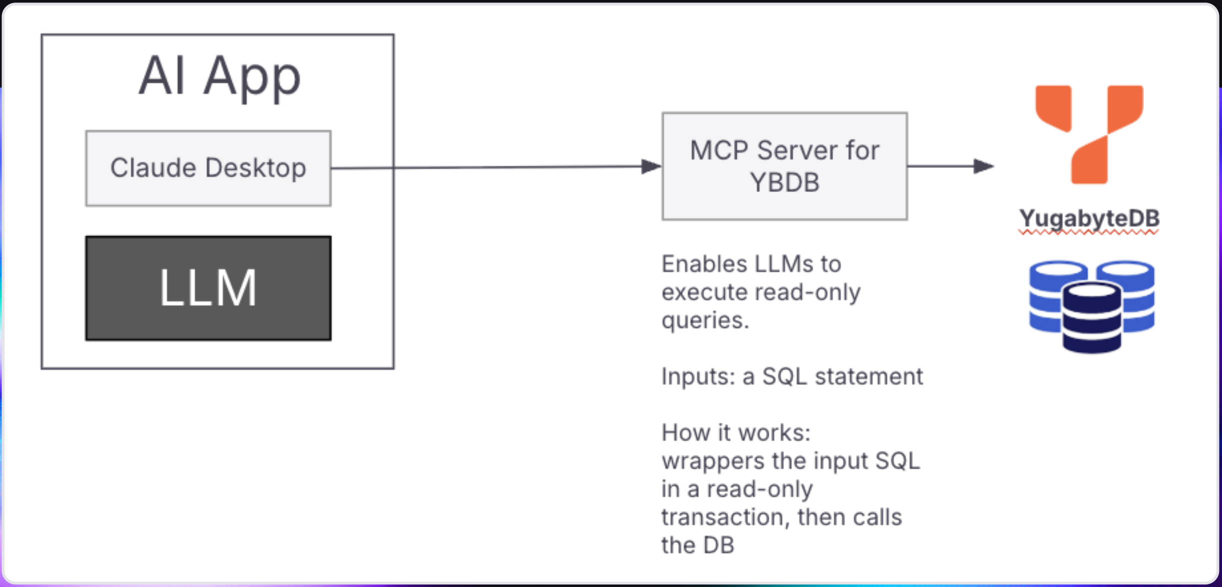
While vectors and RAG enable AI LLMs to utilize pre-loaded data, MCP Servers enable AI LLMs to directly access live data.

MCP Servers can enable AI LLMs to take action via direct access to live services. They are the glue that enables AI LLMs to become flexible "agents" with access to multiple data sources and services.
While an MCP-based approach typically executes slower, consumes more compute resources, and has less result predictability, it gives the LLM more power and yields greater flexibility.

Yugabyte's MCP Server accelerates AI application development providing AI LLMs live, direct access to massive datasets, yielding real-time, intelligent insights.

Explore the YugabyteDB MCP Server

Read More >


AI App
Claude Desktop
LLM
MCP Server for YBDB
Enables LLMs to execute read-only queries.
Inputs: a SQL statement
How it works: wrappers the input SQL in a read-only transaction, then calls the DB
YugabyteDB

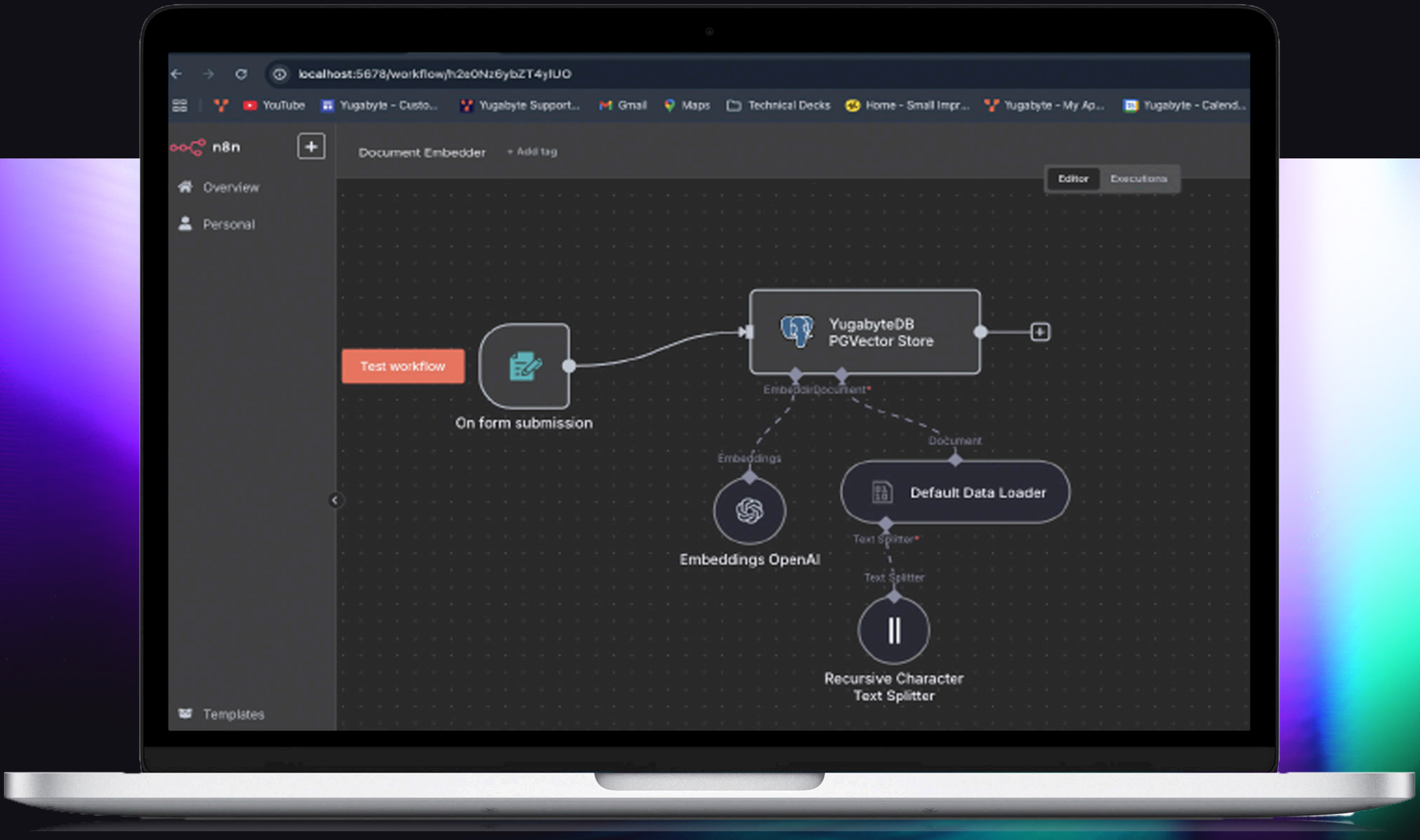
End-to-End AI Workflows with YugabyteDB

YugabyteDB fits naturally into orchestrated AI pipelines.

Tools like n8n can automate ingesting data, generating embeddings, storing vectors, and querying them alongside LLMs such as Ollama (local) or Vertex AI (cloud). This creates a complete stack for production-grade AI systems.

Explore end-to-end AI workflows using n8n and YugabyteDB

Read More



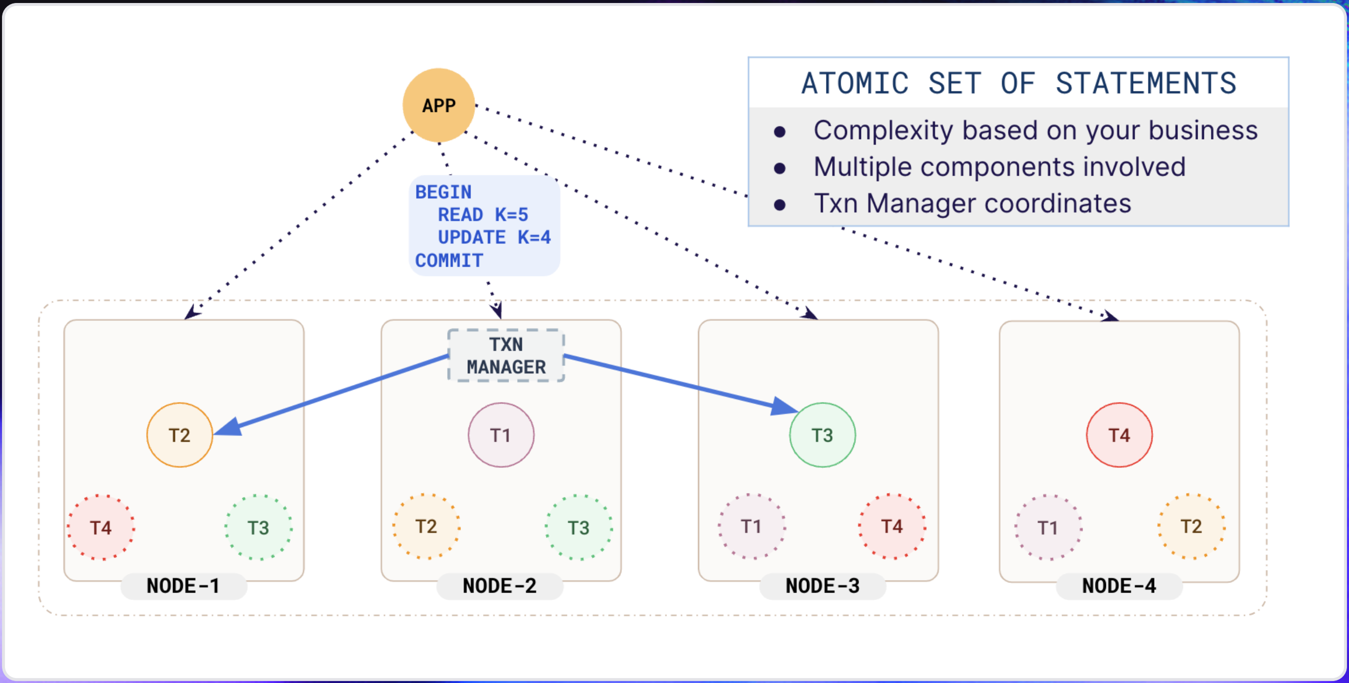
Scalability and Resilience

YugabyteDB's auto-sharded architecture supports hundreds of millions of vectors with active-active replication across global regions.

Architecture that supports 1 billion vectors and an active AI roadmap ensures that YugabyteDB evolves alongside the rapidly advancing AI ecosystem.

Learn how to scale transactions with YugabyteDB

Read More >



Summary

As AI continues to advance, databases will need to integrate more sophisticated vector indexing techniques and support for emerging AI frameworks. YugabyteDB's extensible architecture positions it well for the future.

As you progress on your AI journey, follow these best distributed database practices:

- Choose the right indexing algorithm: Select algorithms like HNSW for efficient query performance
- Monitor and maintain distributed clusters: Regularly check cluster health and leader placement
- Optimize data distribution: Use geo-partitioning for data locality and compliance

Next Steps:

- Get started with YugabyteDB for free
- Try building your own pipeline with YugabyteDB, pgvector, and a local or cloud LLM
- Explore AI tutorials or learn more about architecting GenAI and RAG apps with YugabyteDB
- Join the YugabyteDB open source community and explore the power of AI-ready distributed Postgres

yugabyteDB

For additional information contact a YugabyteDB expert at yugabyte.com/contact.

FOLLOW US