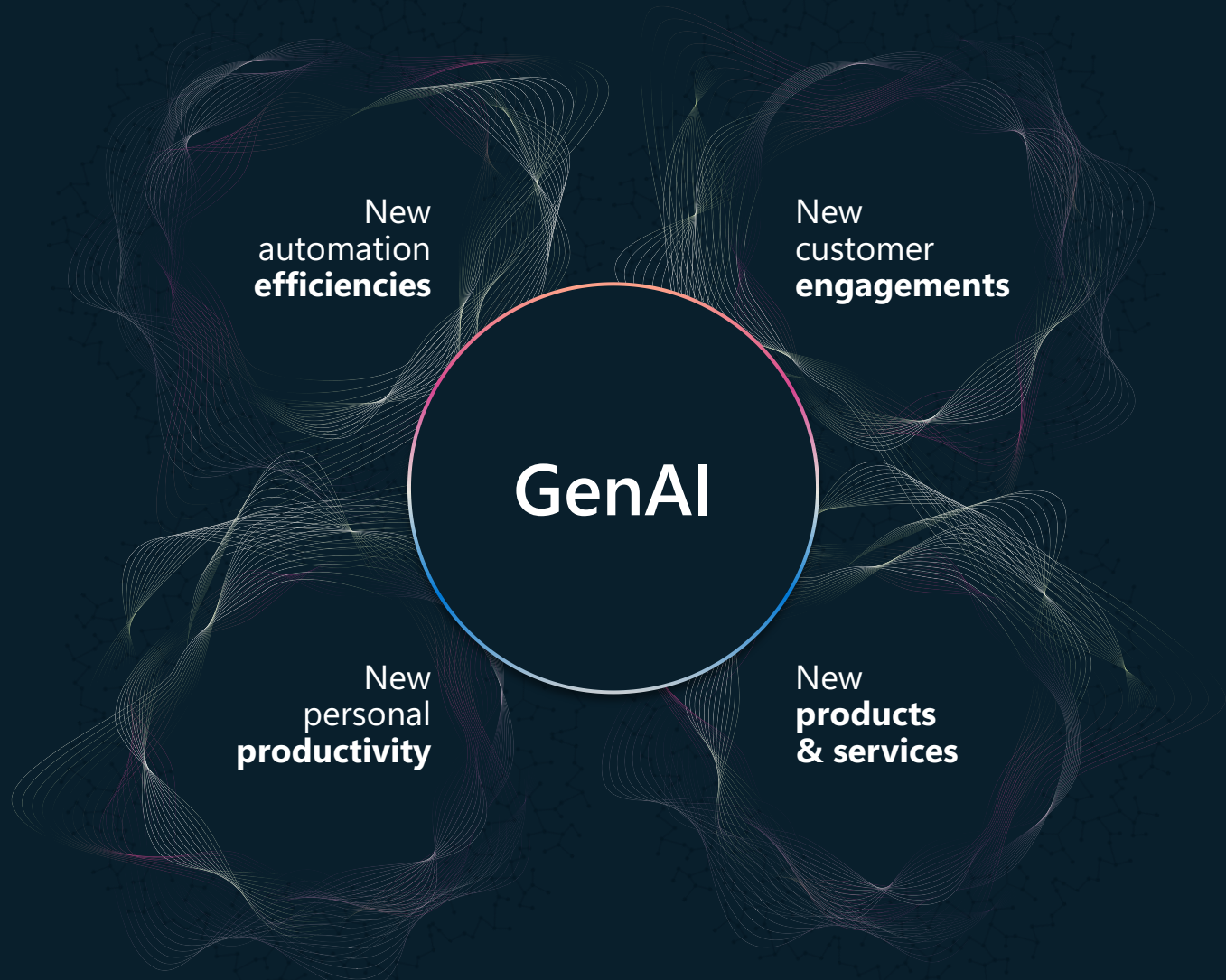
FAILURES?

Generative AI is the greatest force multiplier we've ever had at our fingertips



Today's databases need to be ready to power AI apps

To power intelligent apps, databases need…

To scale to any size

Built-in AI capabilities

Ability to deliver real-time experiences

Enterprise-grade security

No matter where you are in your cloud journey, you can modernize and build new intelligent apps with Azure

1. Connected smart products
2. Transaction processing at scale
3. Real-time fraud detection
4. Service & support bot
5. Information & product discovery
6. Personalize & recommend

Build your own copilots

Bringing domain knowledge to LLMs

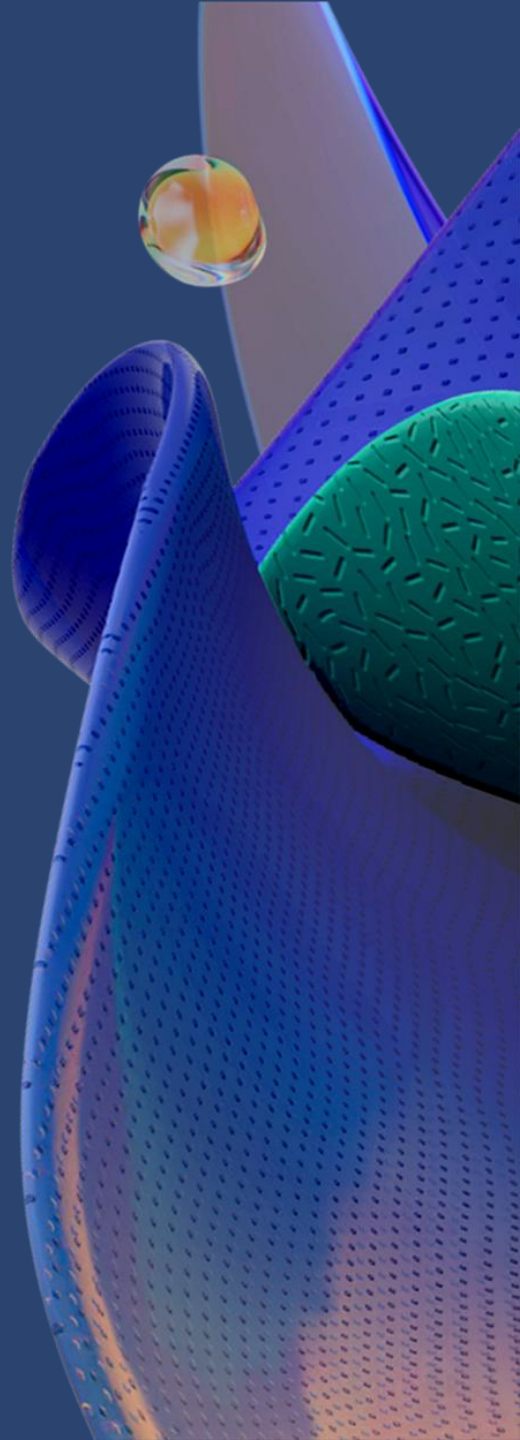
Retrieval augmentation

Learn new facts

Fine tuning

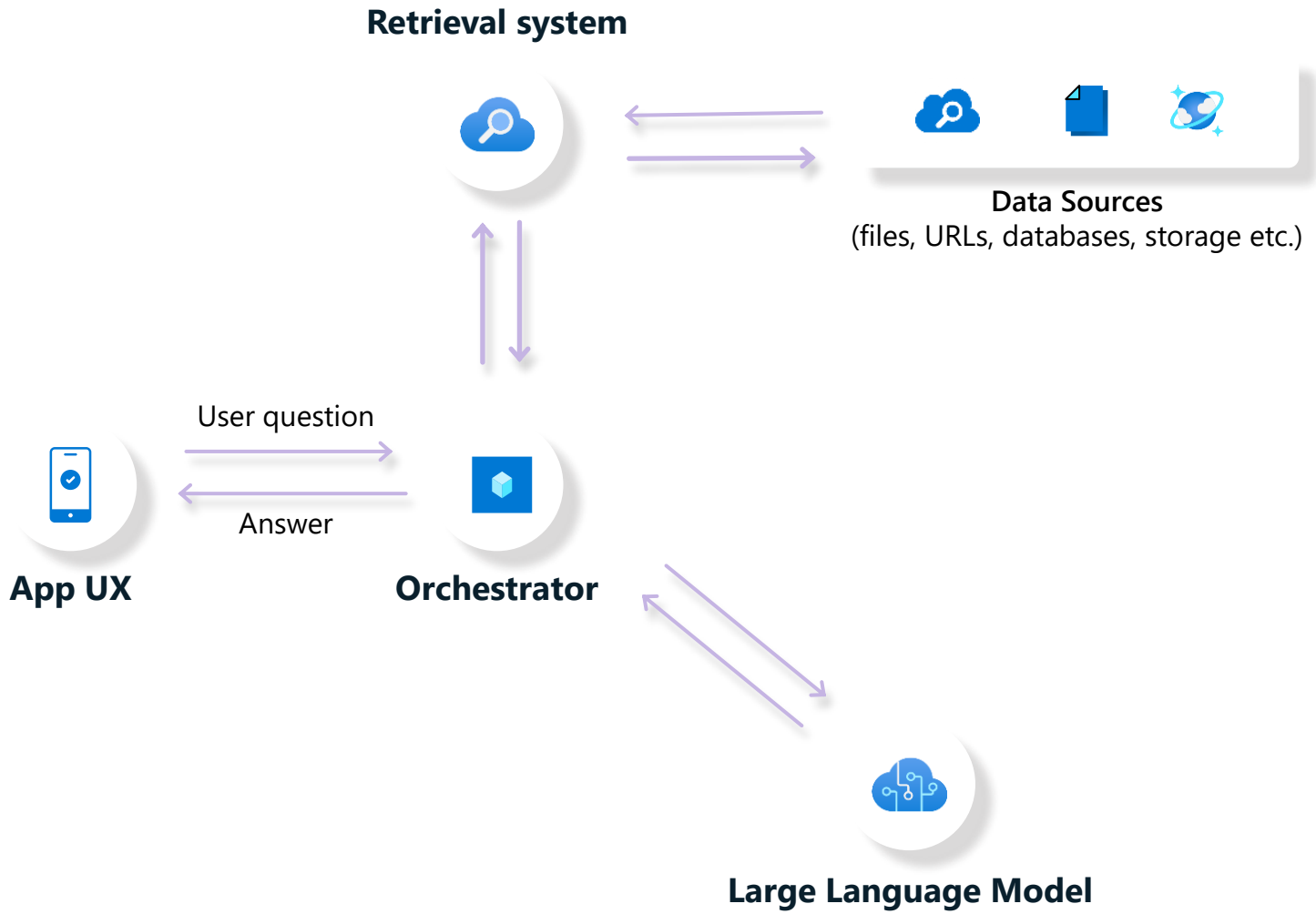
Learn new skills

RAG refresher



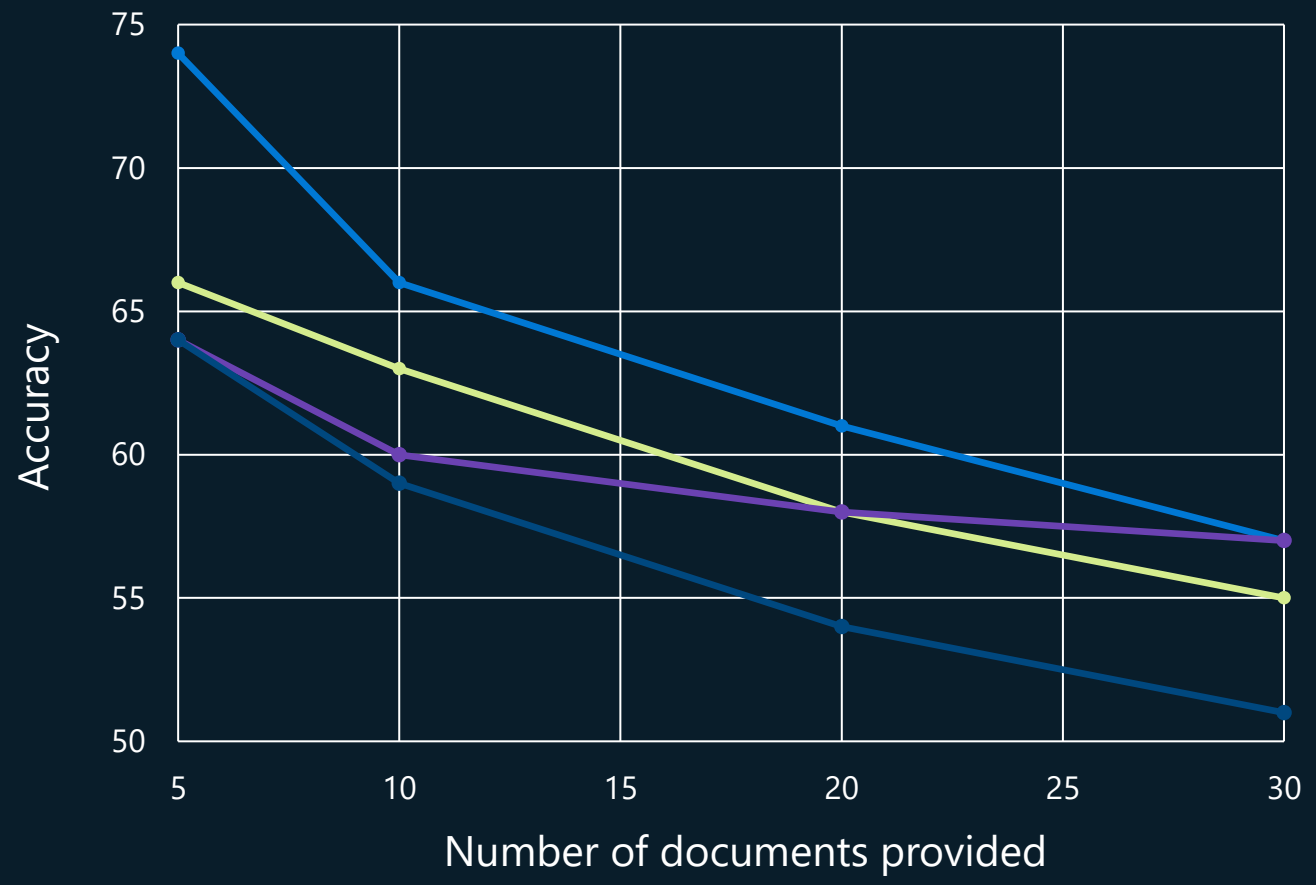
Retrieval – augmented generation

Anatomy of the workflow



Your retrieval strategy matters

More information ≠ better results



Source: Lost in the Middle: How Language Models Use Long Contexts, Liu et al. arXiv:2307.03172

Retrieval strategies

Keyword search

- **For exact, plain text matches**
- “Vocabulary gap” in Q&A systems like Copilot

Vector search

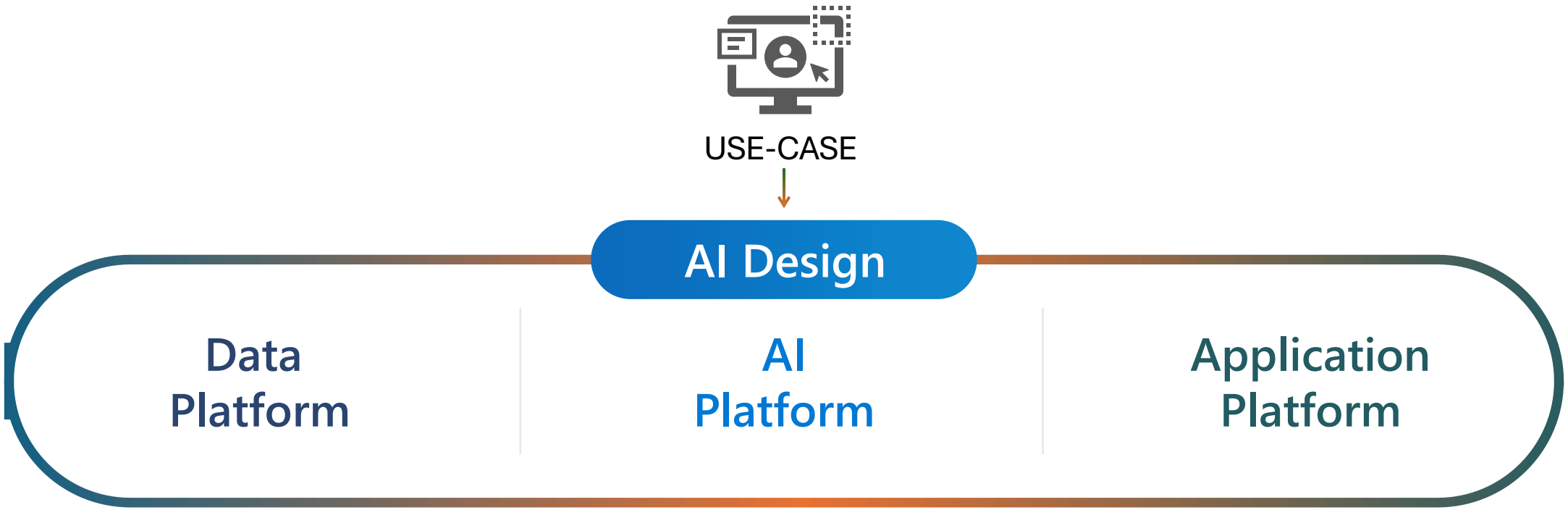
- **For conceptual similarity, or underlying meaning**
- Weak performance on exact matches (like a product ID or code)

Hybrid search

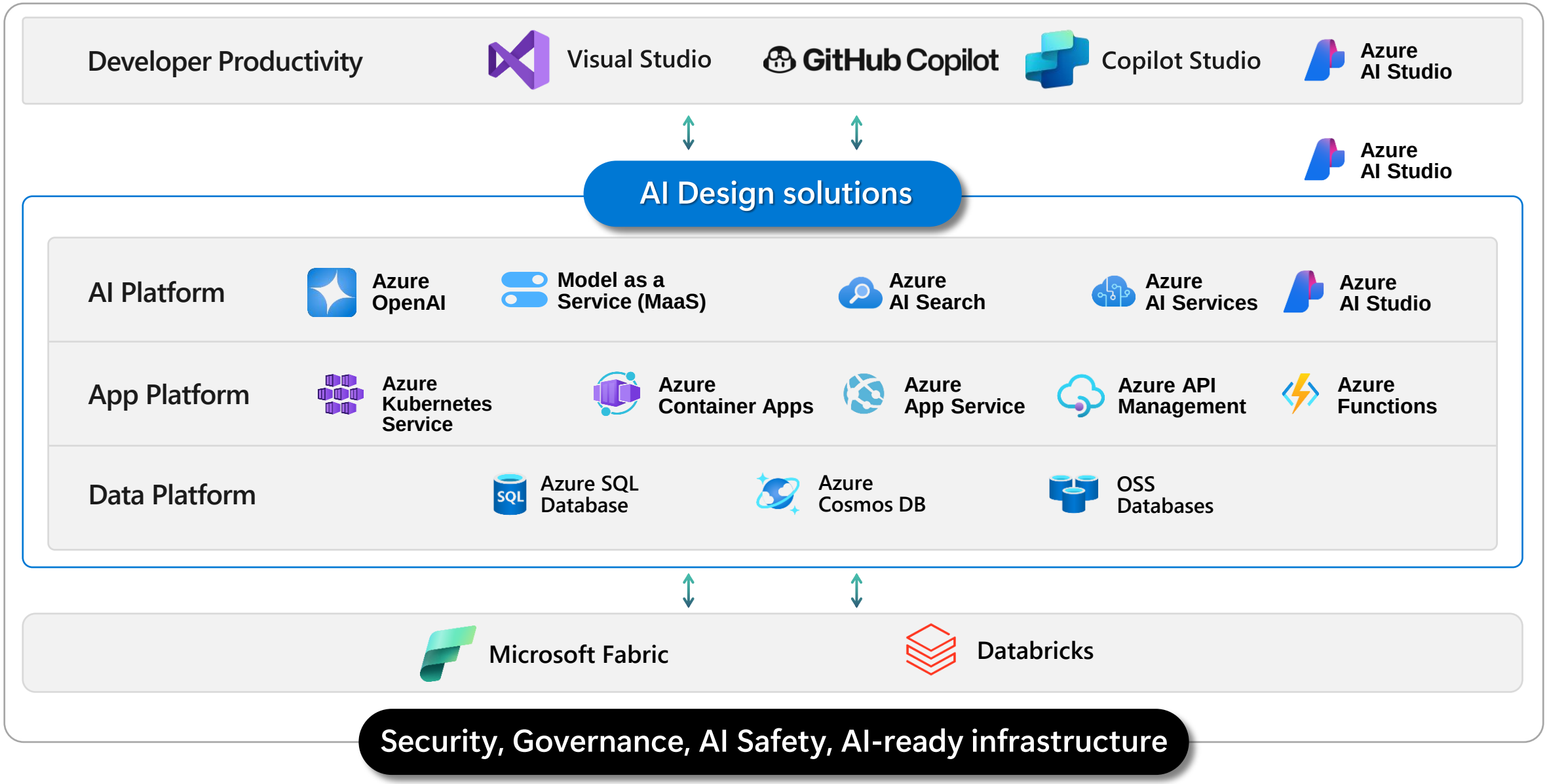
- **Best of both vectors and keywords**
- Brings more accurate responses across various scenarios

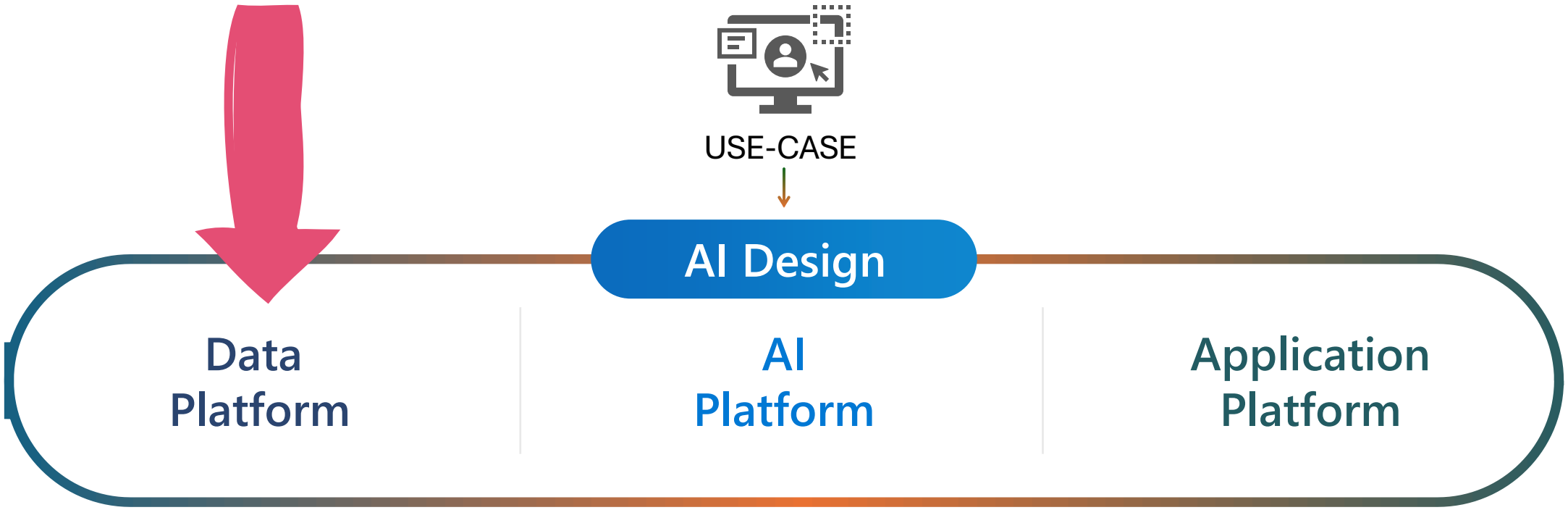
Search re-ranking

- **Scores and ranks all retrieved documents by relevance**
- Reranking runs after performing search strategy (can’t retrieve information)

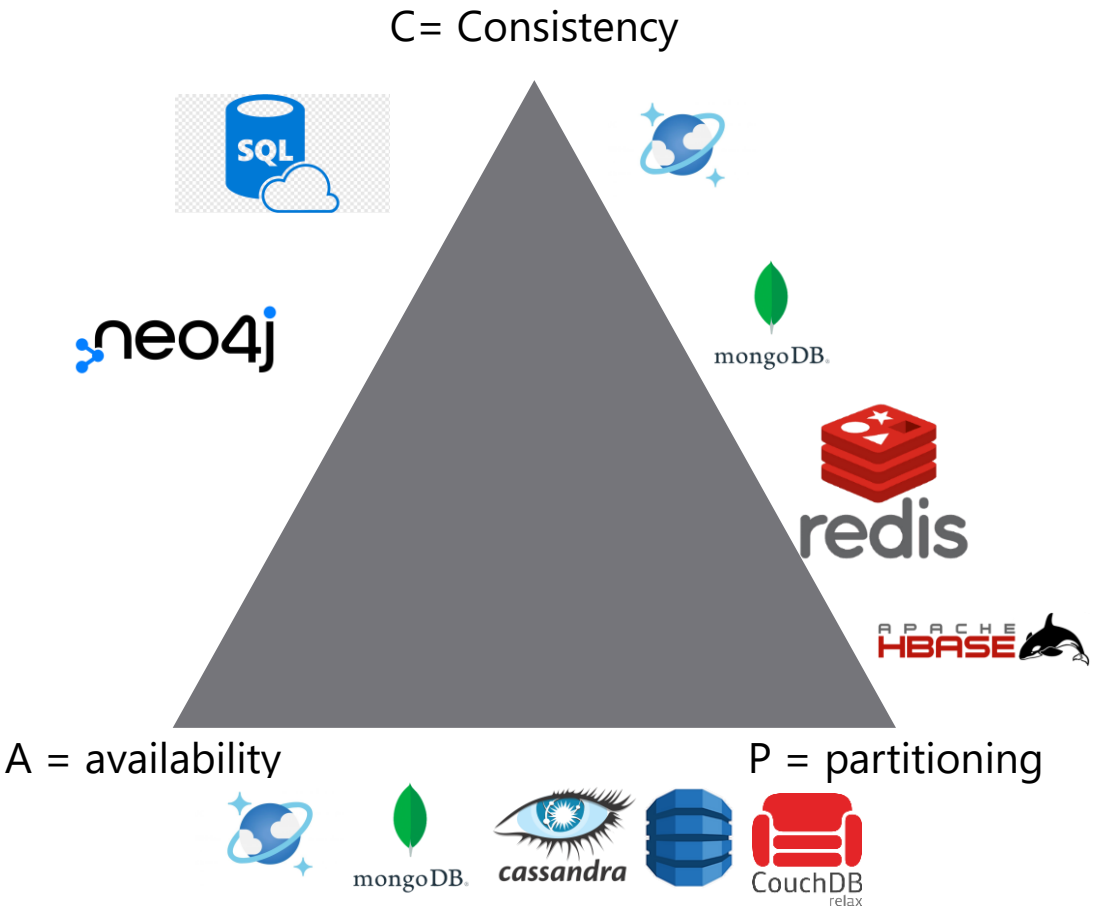

USE-CASE
AI Design
Data Platform
AI Platform
Application Platform

Anatomy of AI Design




USE-CASE
AI Design
Data Platform
AI Platform
Application Platform

CAP Theorem



The CAP theorem, also known as Brewer's theorem, states that in a distributed database system, it is impossible to simultaneously achieve all three of the following guarantees:

1. Consistency: Every read receives the most recent write or an error.
2. Availability: Every request receives a response, without guarantee that it contains the most recent write.
3. Partition-tolerance: The system continues to operate despite arbitrary partitioning due to network failures.

According to the CAP theorem, a distributed database can only provide two out of these three guarantees at any given time. This means that database designers must make trade-offs between consistency, availability, and partition-tolerance based on the specific requirements of their applications.

*Cosmos DB and MongoDB can be configured either way as CP or AP, depending on consistency models

Why would your intelligent app rely on a database?

Database+ Generative AI - scenarios

What	Why	When
Semantic Caching	Reduce latency Saves on token Consumption	Not so dynamic content FAQ, policies, …
Chat history	Conversational context UX improvements LLM optimizations Auditing	A MUST for chat sessions Improving cost & performance
Retrieval Augmented Generation (RAG)	Personalize LLM on your data Cheaper than fine tuning Faster iteration on new data	Any GenAI app
Vector + operational database	No ETL Consistent data Reduced complexity and cost	Proximity search

Powering a new era of apps
that run anywhere, at any scale

Managed

Ease your life and focus on what's important

Integrated

Modern, AI apps on an open, flexible, and common platform

Trusted

Available, scalable, safe and compliant. Unified support

Intelligent

AI-powered management and performance at any scale

Database native vector search OR/AND Azure AI search ?

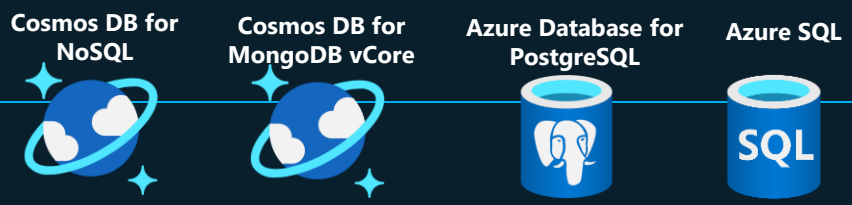
Use Database native capabilities when you…

Want to have an **intelligent app** based on your **dynamic operational data**.

Augment your operational data (e.g. Real-time location-based search on inventory)

Need a **simple architecture**, you do not need anything else.

Need **always on, low latency, guaranteed throughput, geo-replication**, etc.

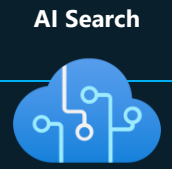


Use AI search if you…

Need to search across **multiple sources**, not only Cosmos DB or PostgreSQL, also pdfs, etc.

Content **does not change often**.

You need **additional capabilities** such as e.g. fuzzy, autocomplete, multi-language, meta-filtering, beyond vector search.



Vector Search in Azure Cosmos DB for NoSQL

Store data + vectors together

Reduced complexity & cost
Transactional data & vectors
Vector indexing and search in a single service

Vector Search + Query Filters

Combine with equality, range & spatial filters
Optimize query

Flexible Indexing

Flat, quantized flat, and DiskANN indexing available

Azure Cosmos DB for NoSQL Capabilities

Serverless or provisioned throughput
Built-in multitenancy
Instant & dynamic autoscale
<10ms point-reads
Globally-replicated
Industry-leading 99.999% SLA

The DiskANN Advantage

Creates a graph-based index that stores compressed vectors in memory, and a full-fidelity graph on SSDs. Advantages:

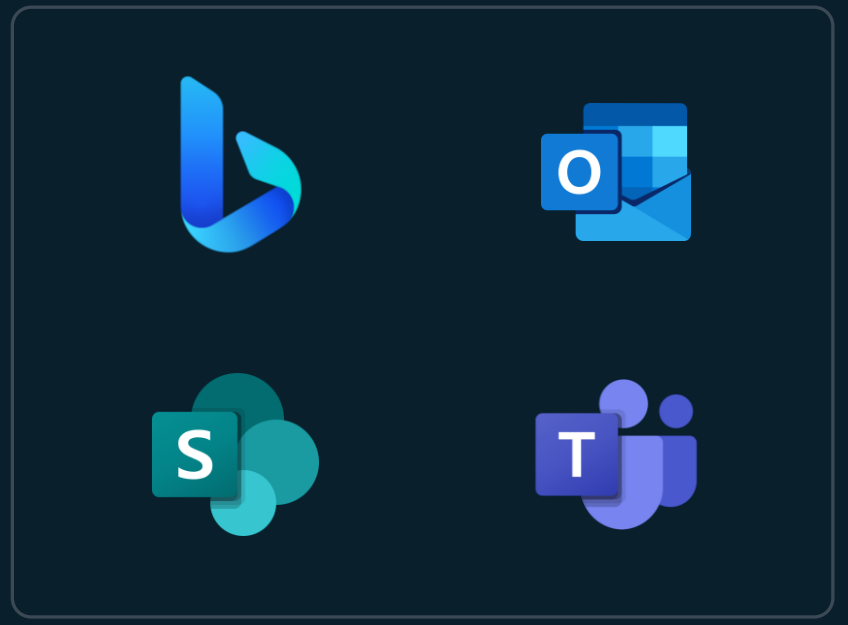
Low latency

High accuracy

Low cost

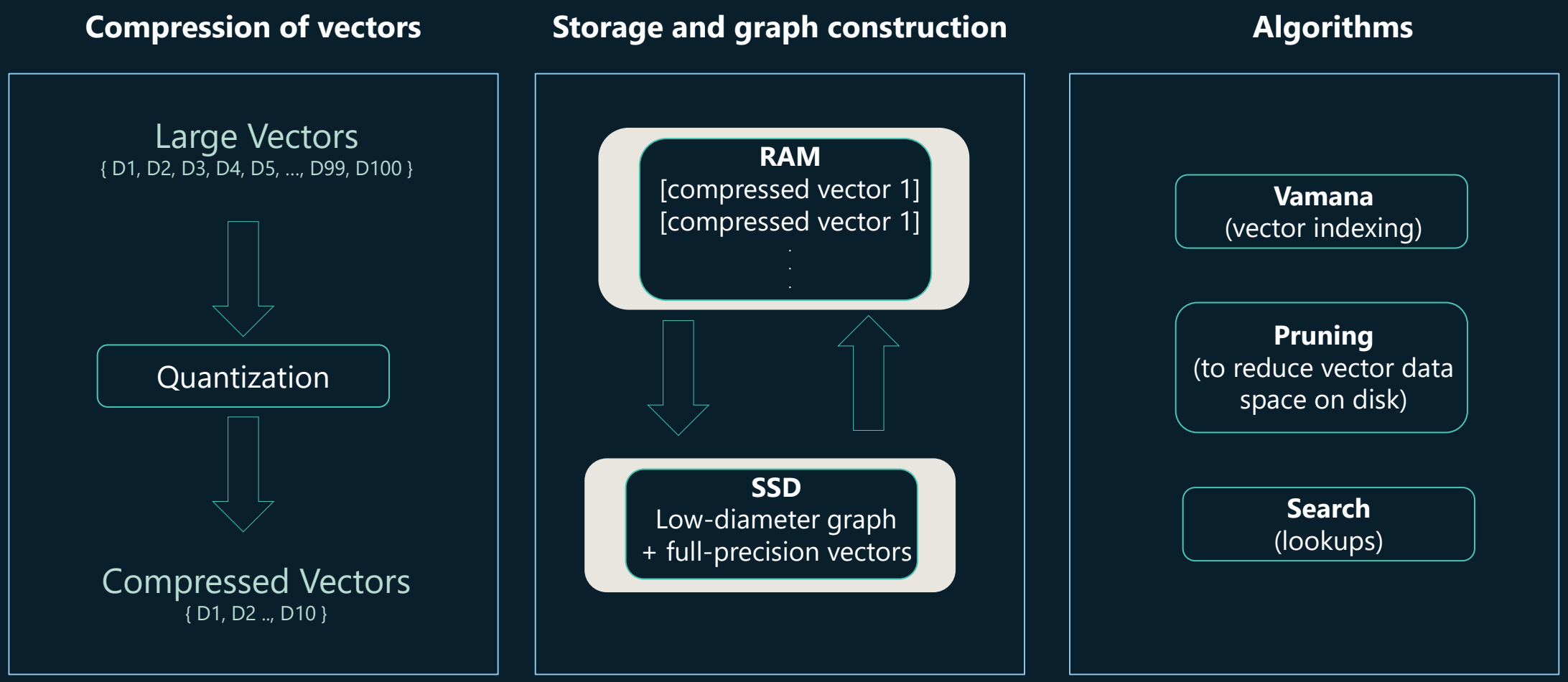
Robust to data changes

DiskANN powers semantic search across Microsoft



- Indexes with up to 400B vectors, trillions of points
- Mean query latency <5ms, >90% recall

DiskANN


Compression of vectors
Large Vectors
{ D1, D2, D3, D4, D5, ..., D99, D100 }
Quantization
Compressed Vectors
{ D1, D2 .., D10 }
Storage and graph construction
RAM
[compressed vector 1]
[compressed vector 1]
SSD
Low-diameter graph
+ full-precision vectors
Algorithms
Vamana
(vector indexing)
Pruning
(to reduce vector data space on disk)
Search
(lookups)

DiskANN

What is similar to this?

Vectors are stored as a property in the same logical unit as the data they represent.

DiskANN index information sits beside each partition.

RU consumption is very low because DiskANN creates a graph of reduced dimensions (low memory usage) which acts as a pointer to a full-precision graph on SDD to retrieve info on nearest neighbors

Document Ingestion & Processing

Extracts text from documents

PDFs, office docs, images, text files

Chunks & vectorizes

Create text chunks, then generate vectors using Azure OpenAI models

Makes it searchable in Azure Cosmos DB

Insert data items and index for efficient search

Solution Accelerator: https://aka.ms/doc2cdb

Natural language to Cosmos DB for NoSQL

Turn your natural language data questions in NoSQL queries

Integrated into Data Explorer at no charge

(Public Preview)

Copilot in Azure capability for multi-turn conversations

(Coming Soon)

Unlocking the power of Open AI and pgvector with Azure Database for PostgreSQL

Simplified LLMs with direct use of PostgreSQL data

Industry leading AI for building intelligent applications

[1.5, -0.4, ..., 20.2]

Native Vectors

Azure AI Integrated

Copilot Powered

Industry leading AI for building intelligent applications

[1.5, -0.4, ..., 20.2]

Native Vectors

Store data + vectors together

Reduced complexity & cost

Vector indexing and search in a single service

Azure AI Integrated

Copilot Powered

Industry leading AI for building intelligent applications

[1.5, -0.4, ..., 20.2]

Native Vectors

Azure AI Integrated

Copilot Powered

SQL Interface with Azure Open AI

Create embeddings from the DB with SQL statements

SQL Interface with Azure AI Language/Vision Services

Invoke ML models from within the DB

Industry leading AI for building intelligent applications

[1.5, -0.4, ..., 20.2]

Native Vectors

Azure AI Integrated

Copilot Powered

Chat-based interface for querying or debugging (coming soon)

Native Vector Search

- **pgvector** extension
- Storing and indexing vectors alongside relational data
- Various indexing & retrieval strategies
- Combine vector queries with metadata filters

DiskANN now in public preview for better latency and accuracy!!!

DiskANN Vector Index in Azure Database for PostgreSQL (microsoft.com)

Generative AI apps

RAG (**Retrieval Augmented Generation**) **apps**

Retrieve private data to ground LLM model responses

Recommendation/Semantic Search

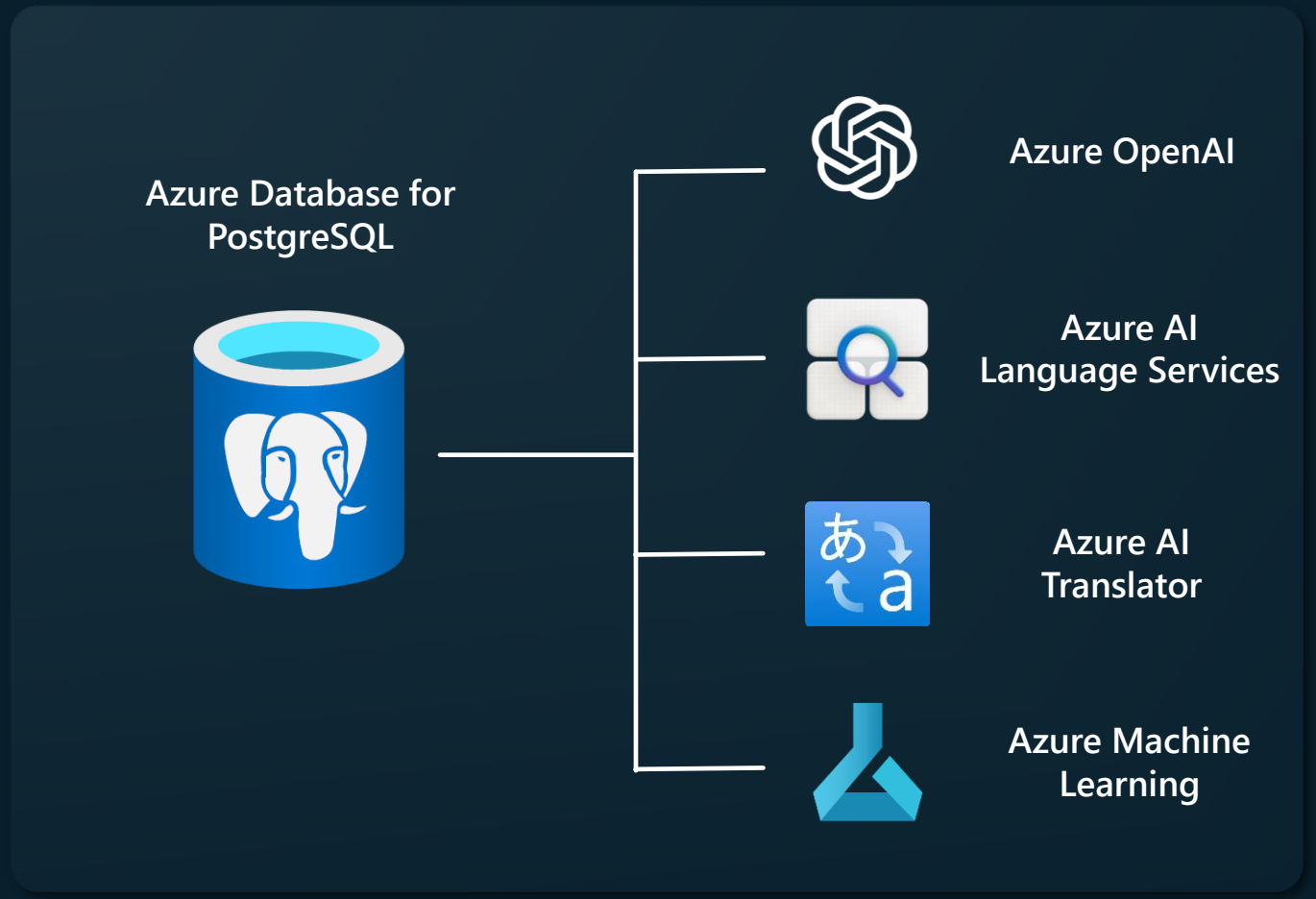
Retrieve similar documents by distance between vectors

Hybrid Search

Combine vector search, row filtering, and full-text search

AI Services integrated into Azure Postgres

Make remote calls directly from PostgreSQL with the azure_ai extension



Enable vector support
Create embeddings
Store vectors & indexes

Analyze sentiment
Extract key phrases
Linked entities (disambiguate)
Recognize entities
Recognize PII entities
Summarize abstractive and extractive

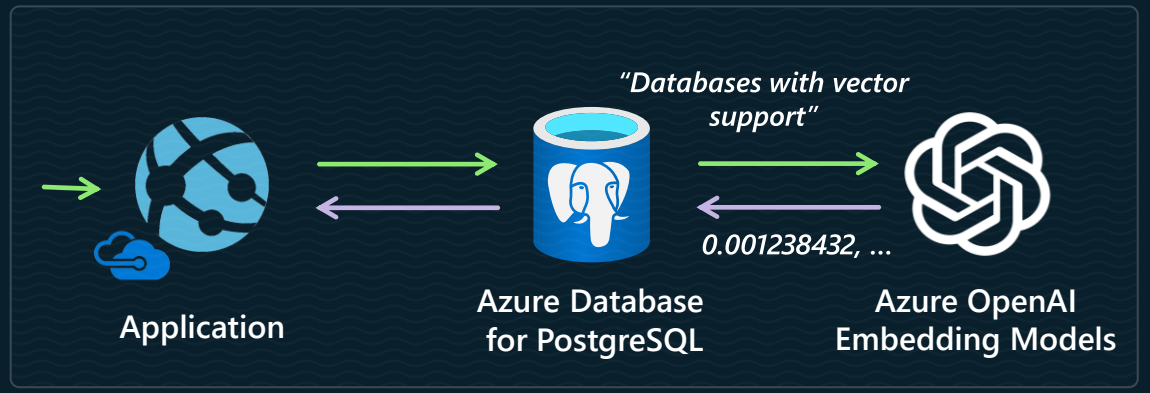
Detect language
Translate

Inference: send data to a deployment name to obtain a result.

Vectors generation with SQL statement

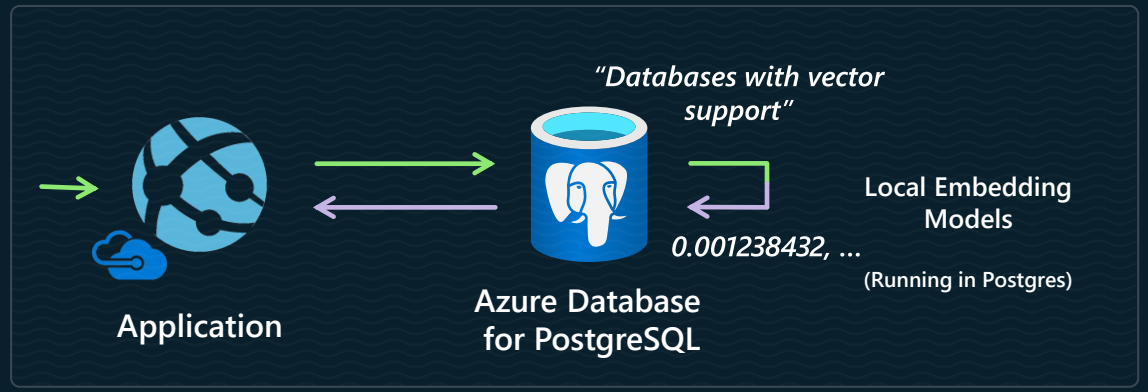
Remote Embedding Models

```
SELECT * FROM <table>
ORDER BY
database_description <->
azure_openai.create_embeddings(
'text-embedding-ada-002',
'Databases with vector support')
```



In-Database Embedding Models
(Preview, azure_local_ai extension)

```
SELECT * FROM <table>
ORDER BY
recipe_embedding <#>
azure_local_ai.create_embeddings(
'multilingual-e5-small:v1',
'Databases with vector support')
```



Data Privacy, data stays in database
90% faster and less costly (no data transmission, no additional external services used)

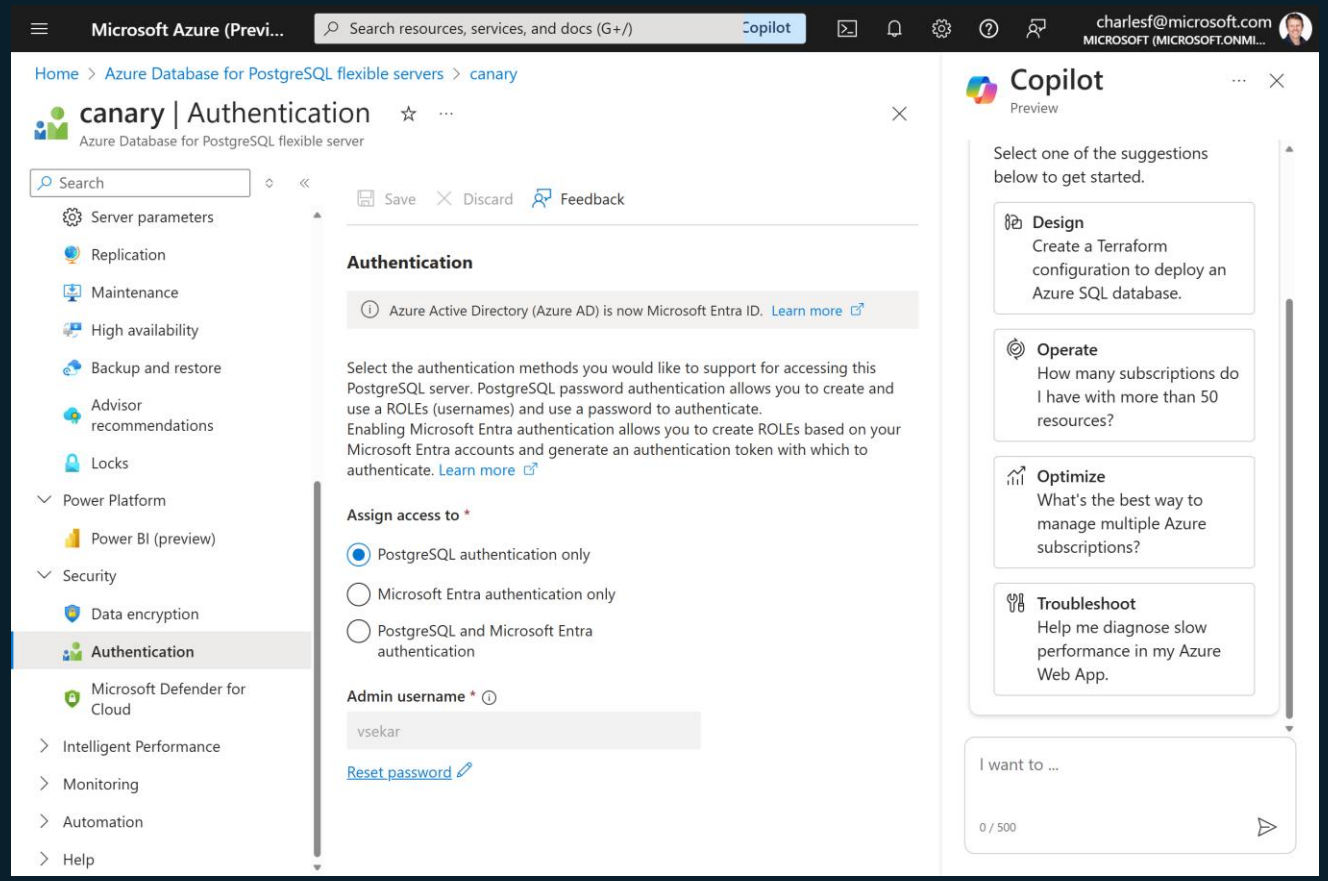
Coming soon

Azure Copilot for PostgreSQL Flexible Server

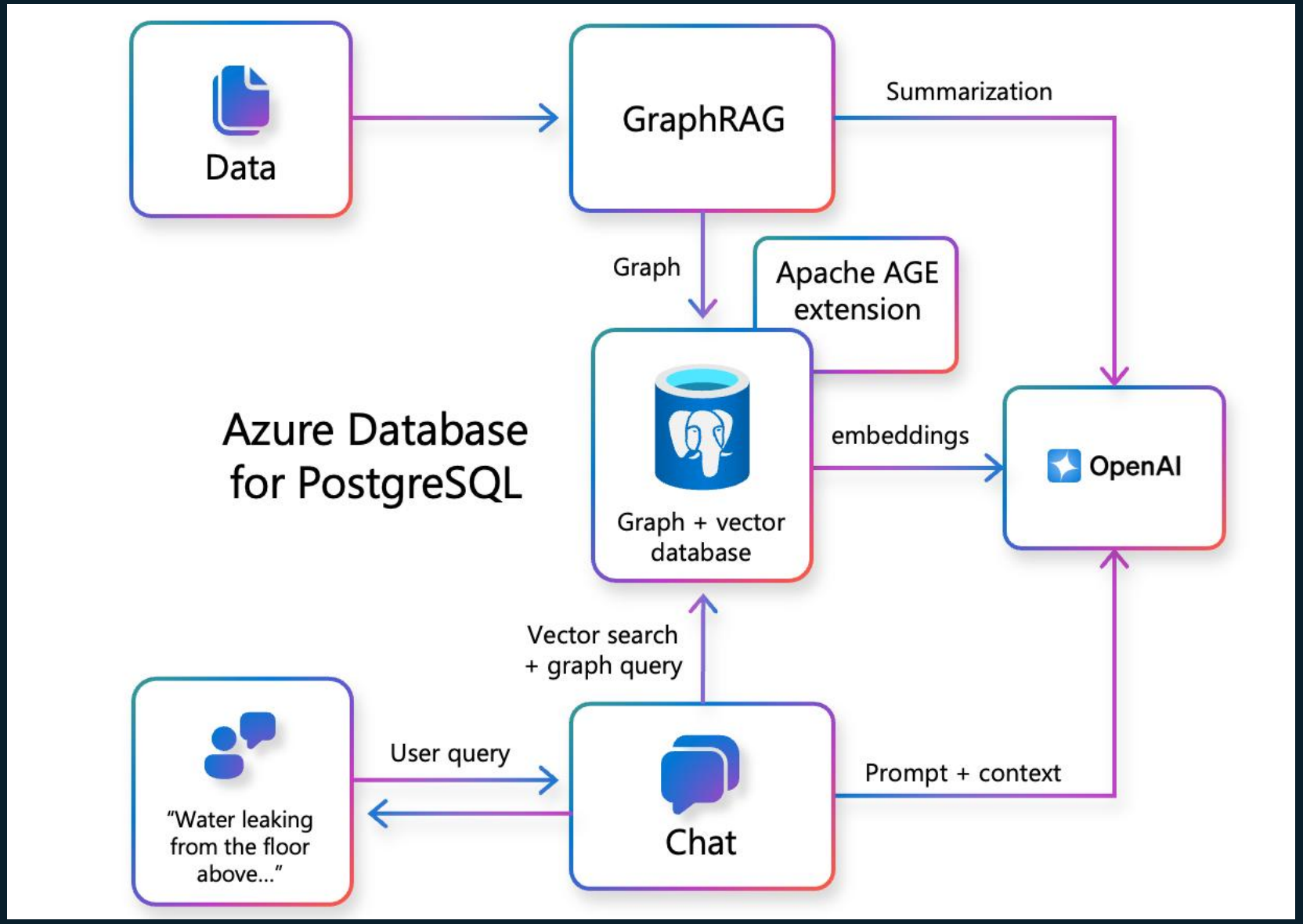
Chat-based interface for querying or debugging

Contextual awareness of the individual server provides the most relevant responses on how to troubleshoot vs. generic documentation

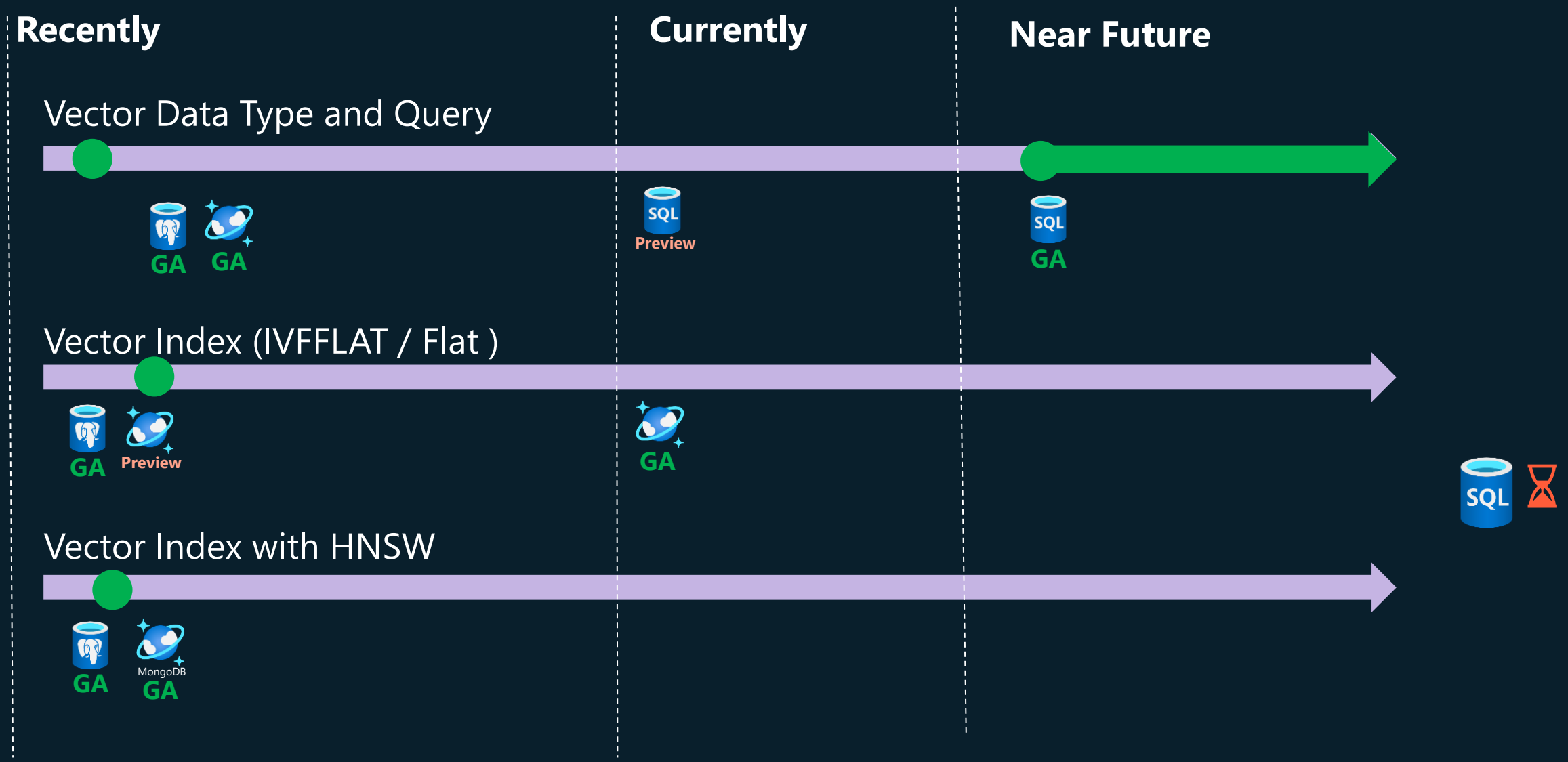
Read-only scenario initially, followed by actions where the Copilot can apply the recommended changes to the server itself with user approval

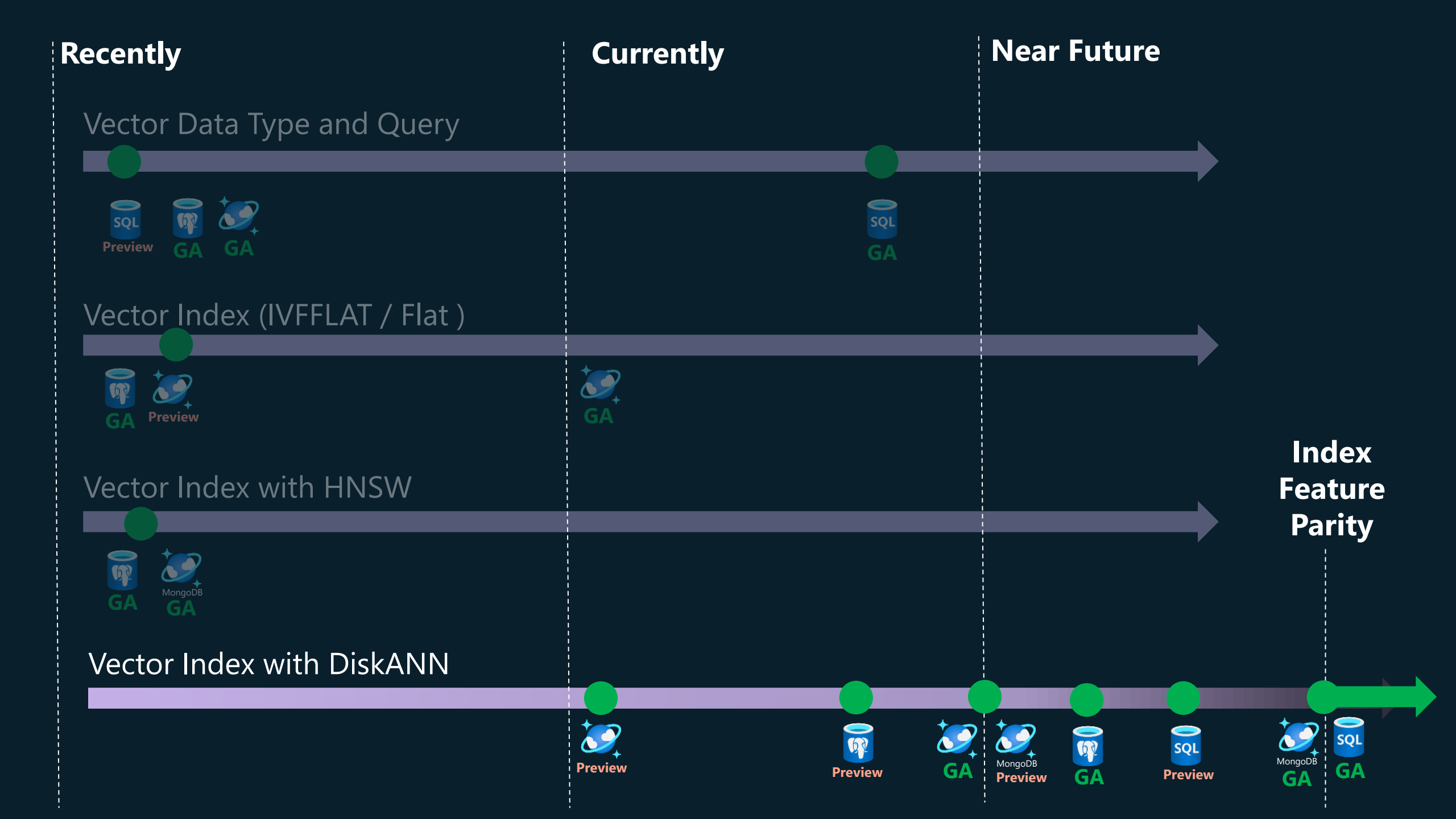


GraphRAG with PostgreSQL



Solution Accelerator:
https://github.com/Azure-Samples/graphrag-legalcases-postgres


Recently
Currently
Near Future
Vector Data Type and Query
GA
GA
SQL
Preview
SQL
GA
Vector Index (IVFFLAT / Flat)
GA
Preview
GA
SQL
Vector Index with HNSW
GA
MongoDB
GA


Recently
Currently
Near Future
Vector Data Type and Query
SQL
Preview
GA
GA
SQL
GA
Vector Index (IVFFLAT / Flat)
GA
Preview
GA
Vector Index with HNSW
GA
MongoDB
GA
Index Feature Parity
Vector Index with DiskANN
Preview
Preview
GA
MongoDB
Preview
GA
SQL
Preview
MongoDB
GA
SQL
GA

Database native vector search Comparison

	Cosmos DB for NoSQL	Cosmos DB for MongoDB vCore	Azure Database for PostgreSQL	Azure SQL	AI Search
Technical features	•Native vector search support with DiskANN. •Full text search (BM25) and Hybrid search (RRF) (in preview).	• Native vector search (DiskANN, HNSW, IVF) •Filtered vector search (preview)	•Native support through the pgvector and pg_diskann (preview) extension. •Semantic Ranking through azure_ai extention and Azure ML. •GraphRAG	•Native vector type (preview) •Vector index through DiskANN (coming soon) •Hybrid Search (preview)	•Native vector search capabilities. •Integration with OpenAI embeddings and cognitive skills. •Full hybrid search support and Semantic Ranker
Use-cases	•Scalable document storage with key-value and document models. •Useful **for fast lookup operations** in metadata-heavy vector search	•Suitable for applications **already using MongoDB,** seeking Azure-managed infrastructure.	•Ideal for applications needing **relational capabilities** alongside vector search. •PostgreSQL extensions (like pgvector) enable seamless vector search integration.	•Ideal for applications needing **relational capabilities** alongside vector search.	•Purpose-built for **search applications**, including hybrid search •Excellent for content-based recommendation systems, enterprise search, and ML-driven search interfaces.

Database native vector search Comparison

	Cosmos DB for NoSQL	Cosmos DB for MongoDB vCore	Azure Database for PostgreSQL	Azure SQL	AI Search
Technical features	•Native vector search support with DiskANN. •Full text search (BM25) and Hybrid search (RRF) (in preview).	• Native vector search (DiskANN, HNSW, IVF) •Filtered vector search (preview)	•Native support through the pgvector and pg_diskann (preview) extension. •Semantic Ranking through azure_ai extention and Azure ML. •GraphRAG	•Native vector type (preview) •Vector index through DiskANN (coming soon) •Hybrid Search (preview)	•Native vector search capabilities. •Integration with OpenAI embeddings and cognitive skills. •Full hybrid search support and Semantic Ranker
Use-cases	•Scalable document storage with key-value and document models. •Useful **for fast lookup operations** in metadata-heavy vector search	•Suitable for applications **already using MongoDB,** seeking Azure-managed infrastructure.	•Ideal for applications needing **relational capabilities** alongside vector search. •PostgreSQL extensions (like pgvector) enable seamless vector search integration.	•Ideal for applications needing **relational capabilities** alongside vector search.	•Purpose-built for **search applications**, including hybrid search •Excellent for content-based recommendation systems, enterprise search, and ML-driven search interfaces.
Cost **Case: 100M vectors (1536 dimensions), 100QPS**	•Pay for *provisioned throughput* (RU/s) *30K RU/s, 1TB storage* *~2,400 EUR/month*	•Pricing based on *vCores and storage capacity* *M200 cluster, 64vcores, 1TB* *~7,200 EUR/ month*	• Pricing based on *vCore* and storage tiers (for DiskANN). *E64bsv5, 64 vcores, 1TB SDD v2* *~6,000 EUR/month*	•Tiered pricing based on *performance* (DTUs or vCores). *Hyperscale Premium series, 64vCores* *• ~10,600 EUR/month*	•Pricing based on *query volume and storage*. *S3 (5 Partitions, 5 Replicas 25 Units)* *• ~49,000 EUR/month*