GENERATIVE AI: Data Architecture with Google Cloud- AlloyDB & AI

Prasad Venkatachar

Sr. Director Products & Solutions

@Pliops

SNIA COMPUTE, MEMORY, AND STORAGE SUMMIT

Solutions, Architectures, and Community

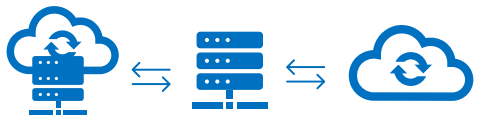
VIRTUAL EVENT, MAY 21-22, 2024

Topics

- Data Platform Considerations
- Gen AI – Enterprise Verticals & Horizontal
- Google Cloud AlloyDB Omni Solution
- How do I use it for Business Applications
  - Operational Store, Analytics, Gen AI
- Retrieval Augmented Generation Concepts & Implementation
- AlloyDB AI Features

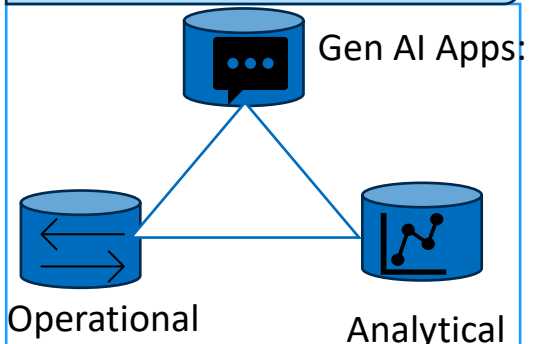
Next Data Platform Consideration

Seamless Data Movement



Edge – Core- Cloud

One Technology

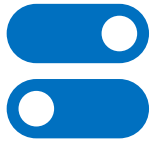


Robust Support & Compliance

Enterprise Support

Compliance

Open Source or Compatible



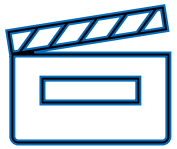
Flexibility to Switch

"71% of respondents in the Data and AI Trends Report plan to use databases integrated with gen AI capabilities."

Gen AI Opportunity Everywhere

Industry Verticals

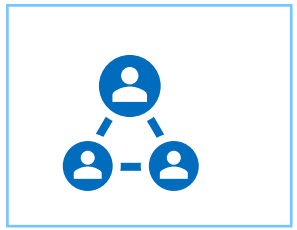
Content & Media



Banking & FSI

Retail

Horizontal Functions



HR & Finance

Marketing

Customer Support

One Platform:

Transactions/Analytics/Gen AI

E-Commerce Applications

Real Time Biz Analytics

Gen AI Chatbots

Transaction Database

Columnar Engine

Vector Database

AlloyDB Omni

AlloyDB

AccelDB PLiOPS

Lenovo

XDP Data Accelerator + SSDs

SSD SSD

Lenovo Edge Server

XDP Data Accelerator + SSDs

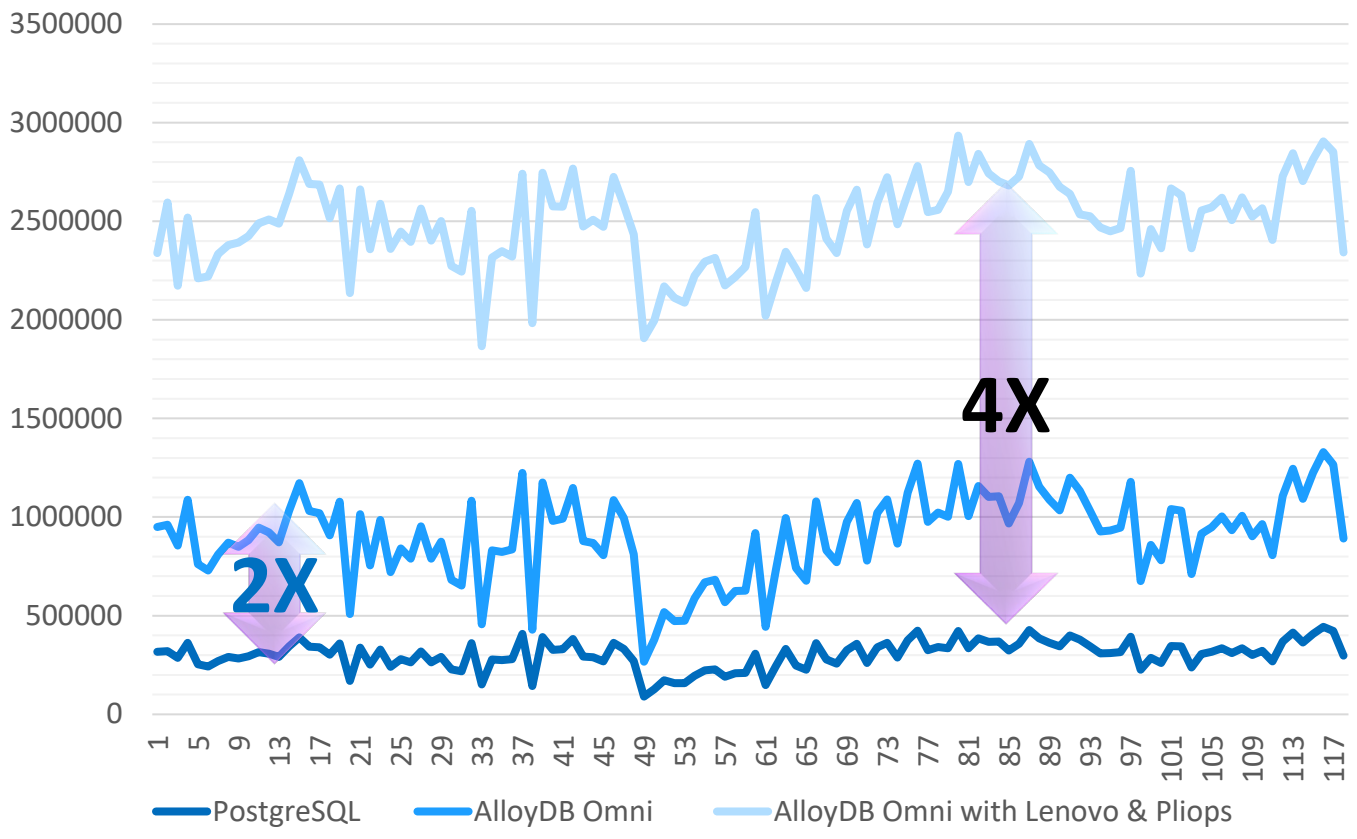
SSD SSD SSD SSD

Lenovo Datacenter Server

Google Cloud

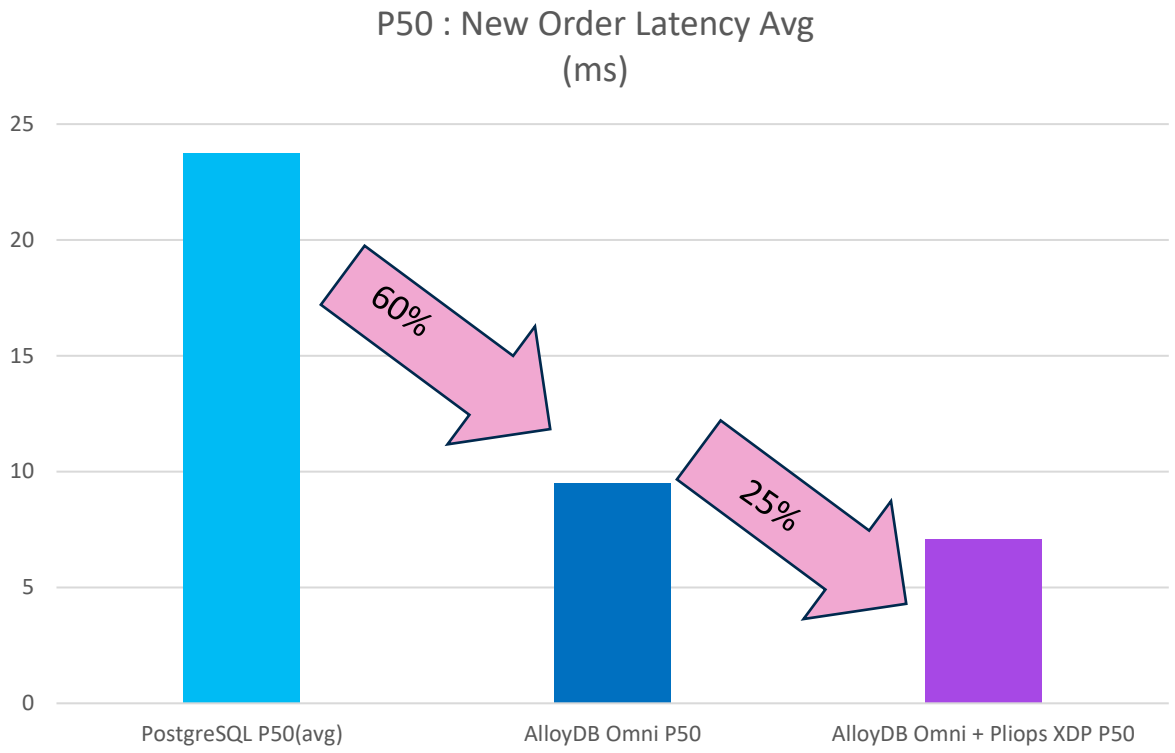
4X PostgreSQL Performance:
Transactional Workloads

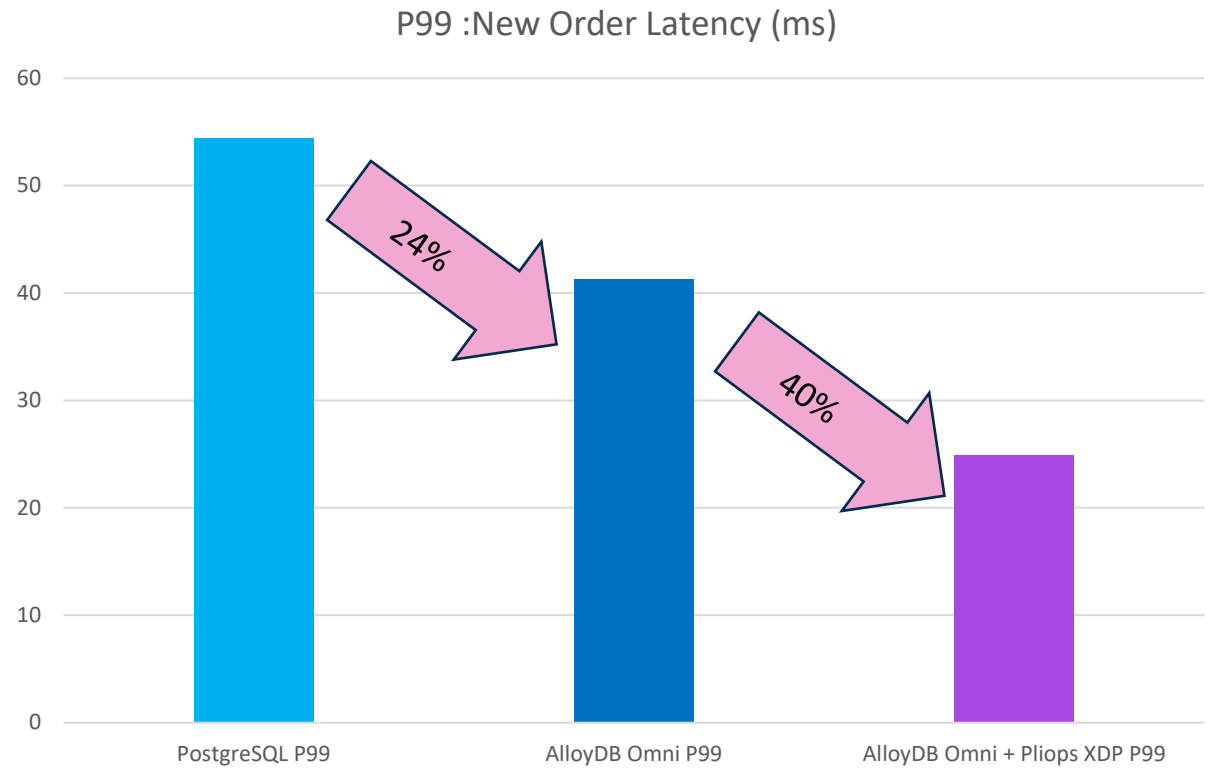
- 2X Higher Transaction served with AlloyDB Omni compared to PostgreSQL
- Up to 4X Transaction served from PostgreSQL to AlloyDB Omni with Pliops & Lenovo
- Serve more Web and Mobile users transaction Requests to meet demand
- Seamlessly scale database while maintaining high performance.



Application User Experience:

Significant Average & Tail Latency Reduction

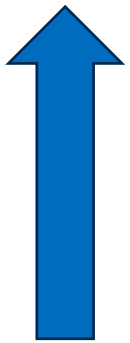


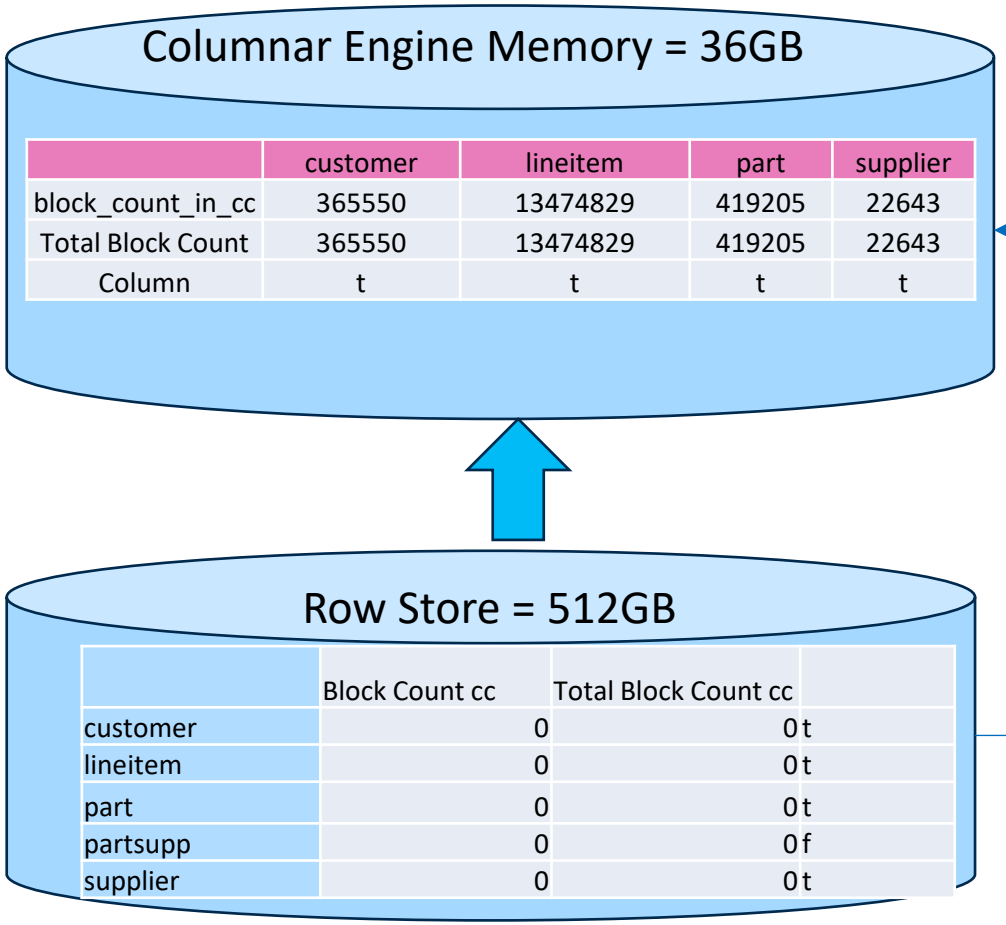


Analytics: AlloyDB Columnar Engine

Implementation & Benefits

Real-time business insights





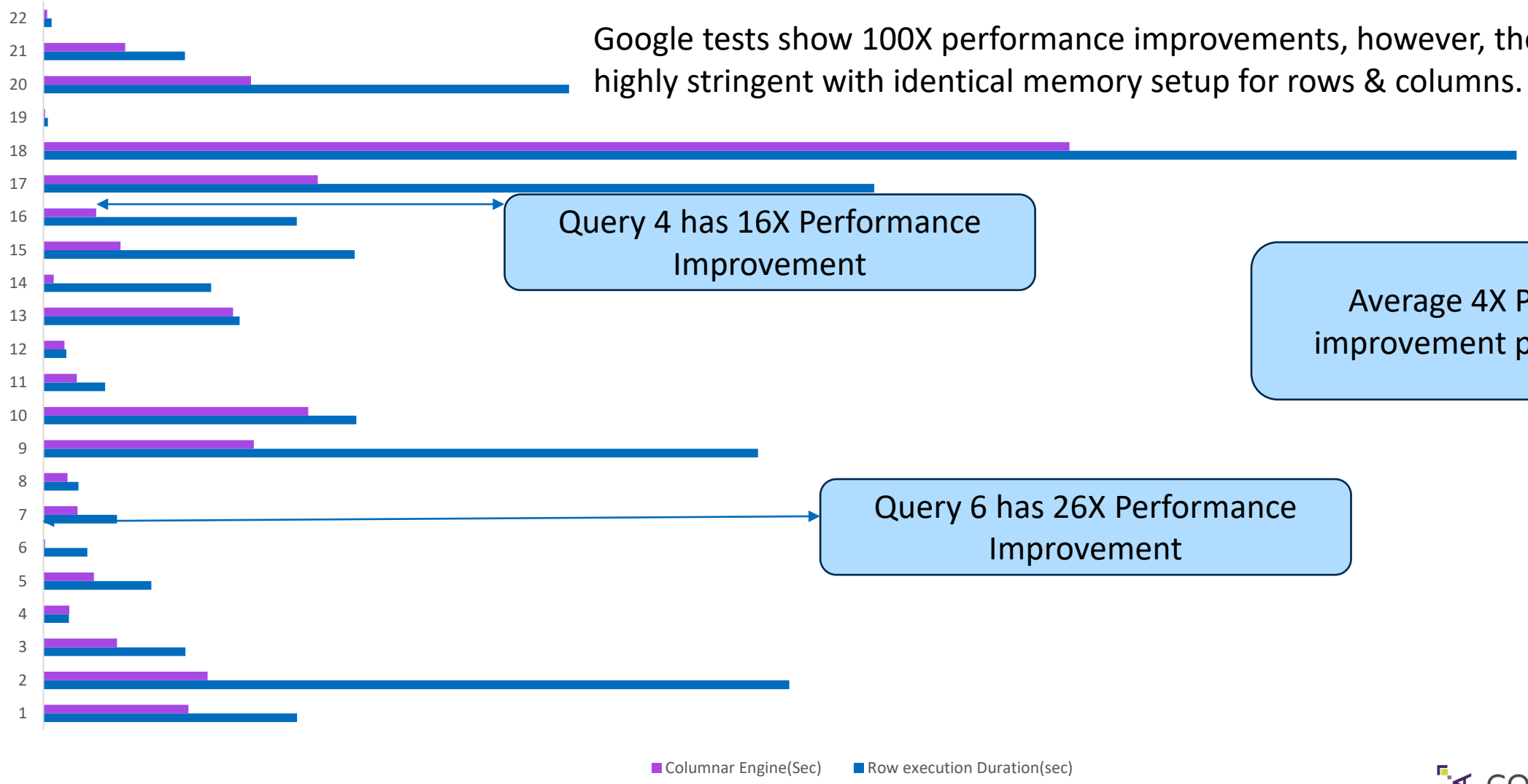
google_columnar_engine.relations=' TPC-H. Customer,,,,,,,

Populating Tables

Recommend Col Memory Sizing

columnar_engine.memory_size_in_mb=36GB

Row vs Columnar Execution



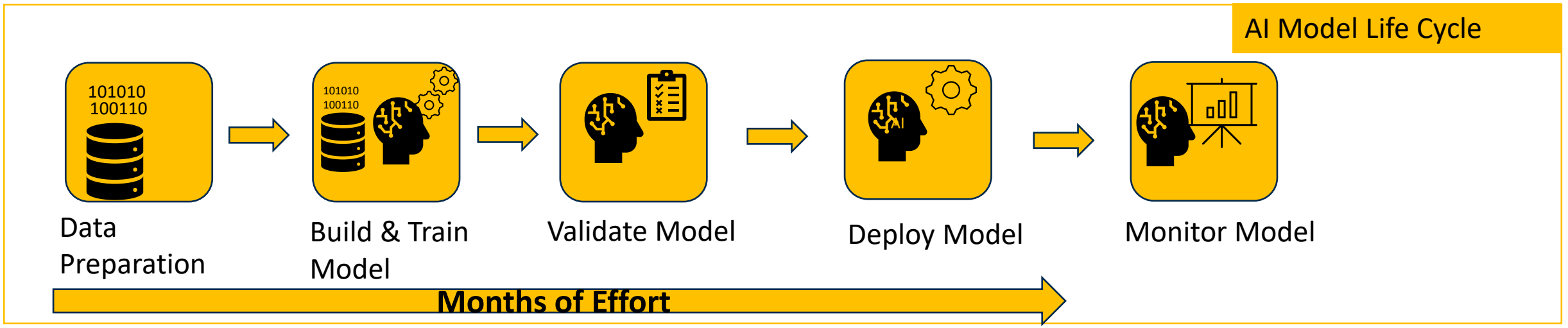
Google tests show 100X performance improvements, however, these tests are highly stringent with identical memory setup for rows & columns.

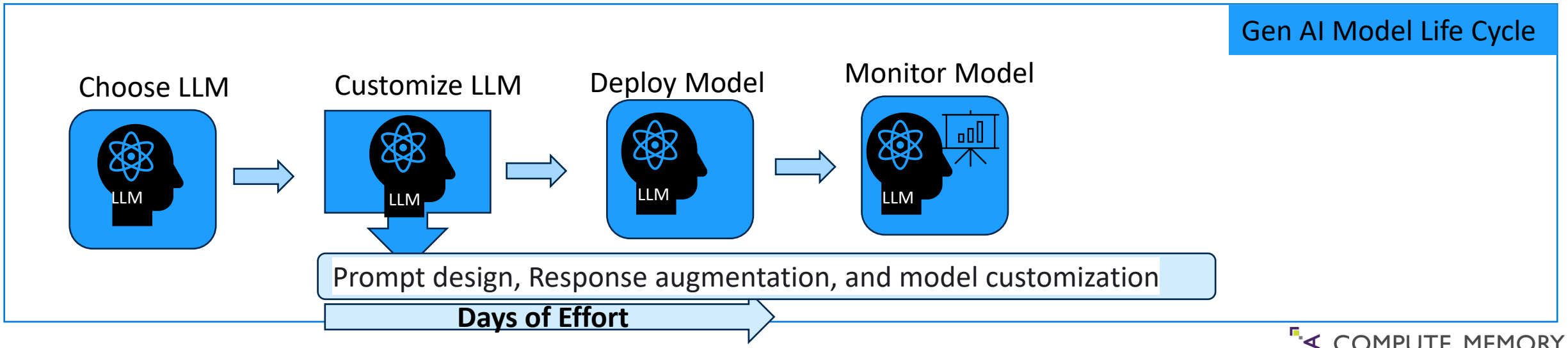
Query 4 has 16X Performance Improvement

Average 4X Performance improvement per Query level

Query 6 has 26X Performance Improvement

AI VS Gen AI Life Cycle





RAG Use cases & Advantages

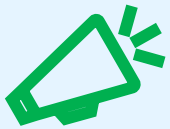
Retrieval Augmented Generation Use Cases

- Chatbots
- Searching for similar content (Text, Image, Video)
- Personalized recommendation
- Document Summarization

Identify the right use case is the Key

Retrieval Augmented Generation Advantages

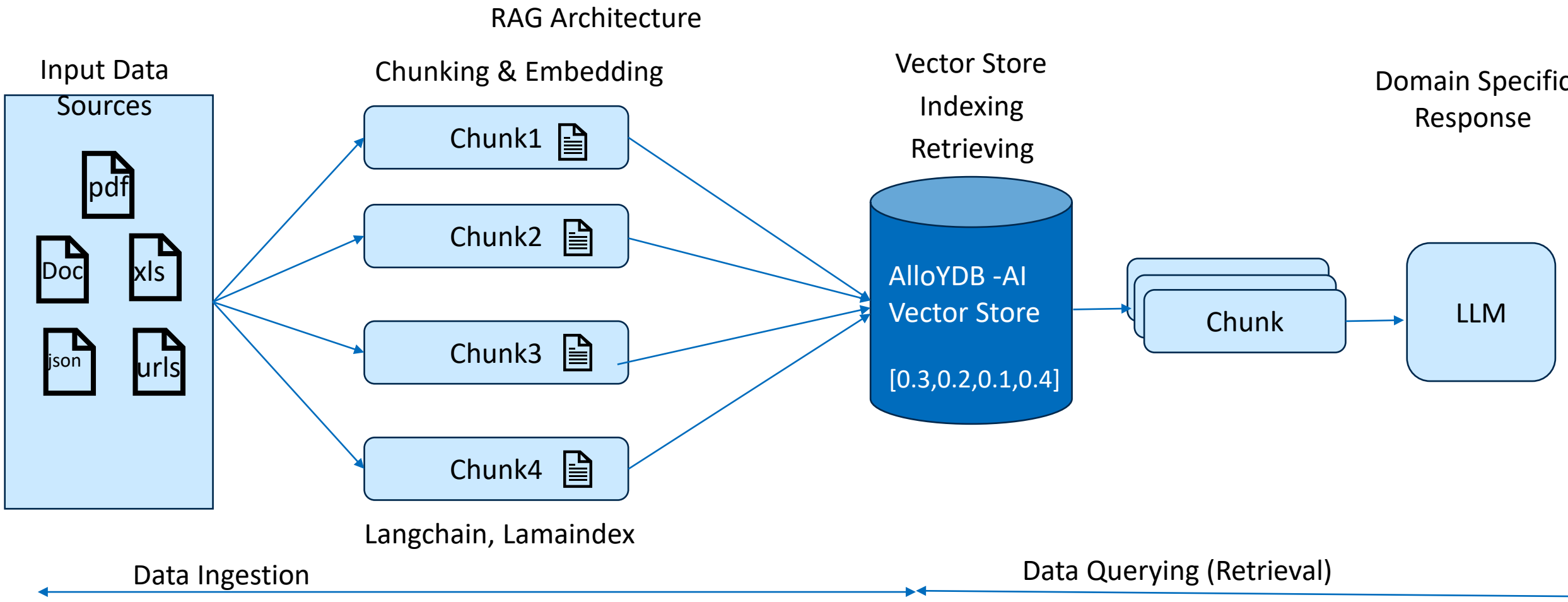
- Reduce Hallucinations



- Upto -Date Information for user response

- Enable LLM to cite Source

AlloyDB AI – Vector Database



AlloyDB AI Implementation

- Add google_ml_integrations & PgVector
- Register & Integrate the Embedding model endpoint - Preview Mode
- Generate Embedding using embedding() function within AlloyDB
- Store the vector form of data in AlloyDB

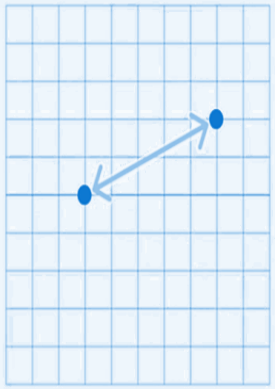
```
CREATE EXTENSION google_ml_integration;

CREATE EXTENSION vector;

GRANT EXECUTE ON FUNCTION emedding to user;

SELECT embedding ('textemedding-geck@001', 'AlloyDB is high performance Postgres database for Enterprises to build Operational, Analytics & Generative AI Applications);
```

Technology Preview

Vector Search

Find Most Similar Embeddings

L2_distance(vector1, vector2)

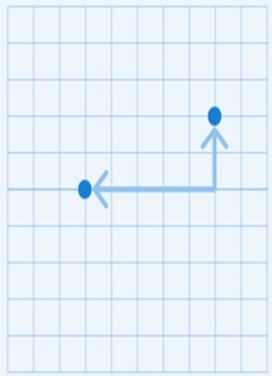


Squared Euclidean

(L2 Squared)

$$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

L1_distance(vector1, vector2)

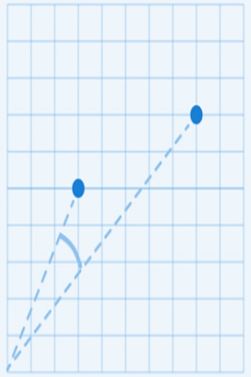


Manhattan (L1)

$$\sum_{i=1}^{n}|x_i - y_i|$$

AlloyDB Database

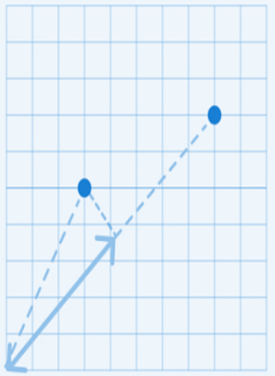
Cosine_Distance(vector1, vector2)



Cosine Distance

$$1 - \frac{A \cdot B}{||A||\ ||B||}$$

Inner_product(vector1, vector2)



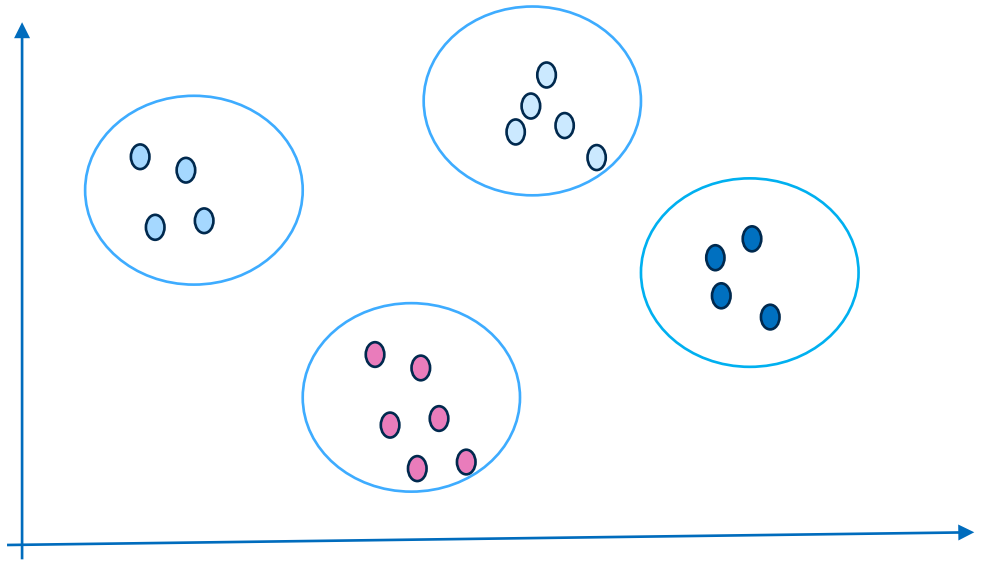
Dot Product

$$A \cdot B = \sum_{i=1}^{n} A_i B_i$$

Source: Deep Learning. Ai

Postgres Vector Indexes

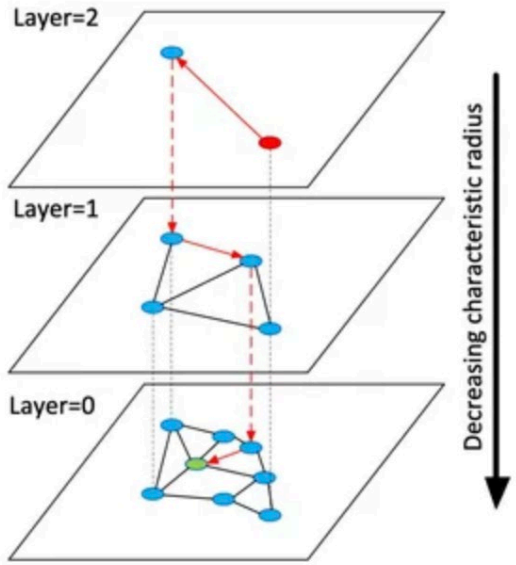
IVFFlat (Inverted File with Flat Compression)



Number/size of the lists

Search: Number of lists to be verified

Hierarchical Navigable Small Worlds(HNSW)

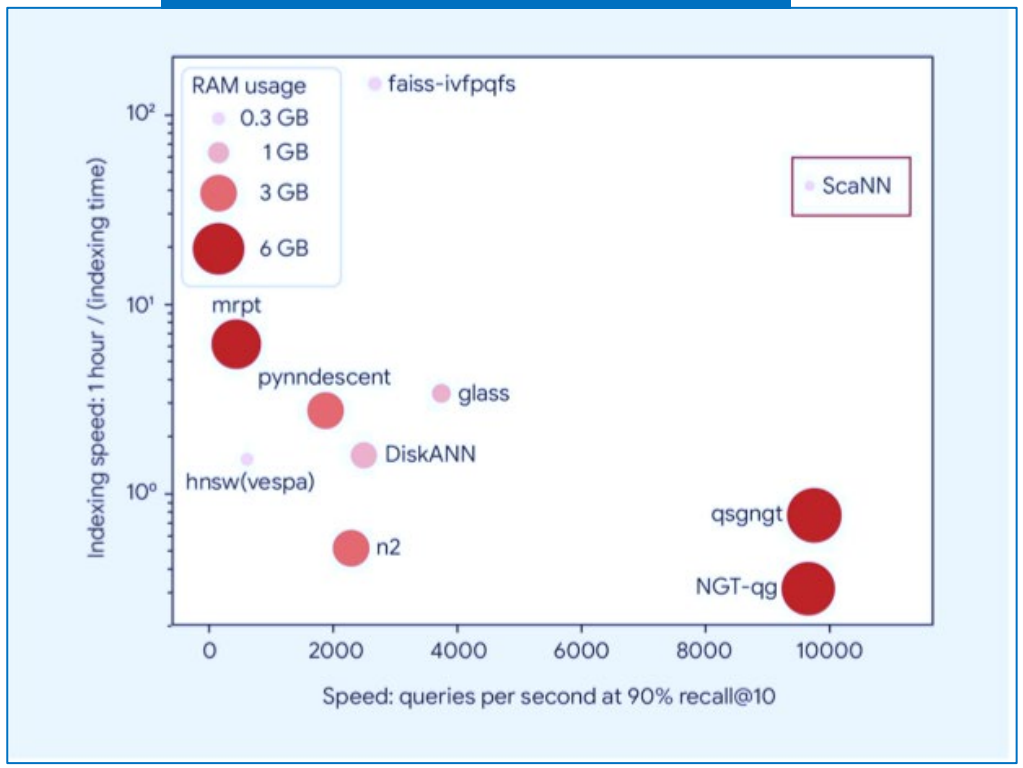


m: Maximum number of connections per layer

Ef_construction : Size of Dynamic list for construction graph

ScaNN Index Benefits over HNSW

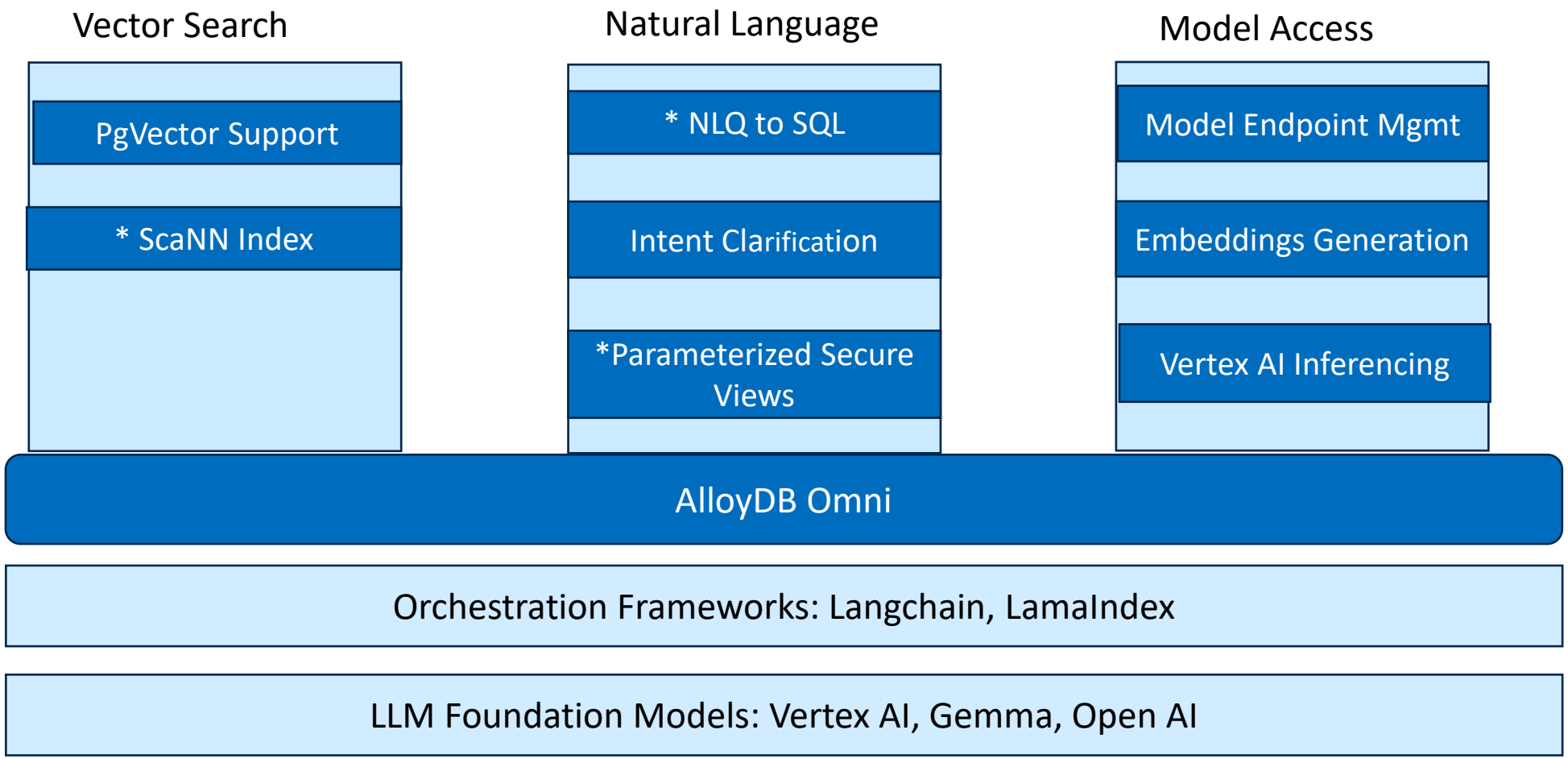
ScaNN Index for AlloyDB



Method	ScaNN for AlloyDB	HNSW
Index size	Tree overheads are typically smaller. Quantization further reduces size.	Graphs have more edge connections therefore bigger index overhead.
Index time	Tree training and indexing are faster. The operation is of O(#vectors * #centroids).	Graph construction fundamentally requires many vector–vector comparisons.
Memory access	Vectors in tree leaves are stored contiguously. Memory access is more continuous and friendly to SIMD acceleration.	Graph walk beam search requires random memory accesses. NGT partially solves this by duplicating data in nodes but leads to bloated indices.
Latency	Constant search speed across queries. Typically configured to search a fixed number of leaves.	Varies more per query. Easy queries terminate early as nothing is left in the working set of beam search. But tail latency for harder queries could be higher.

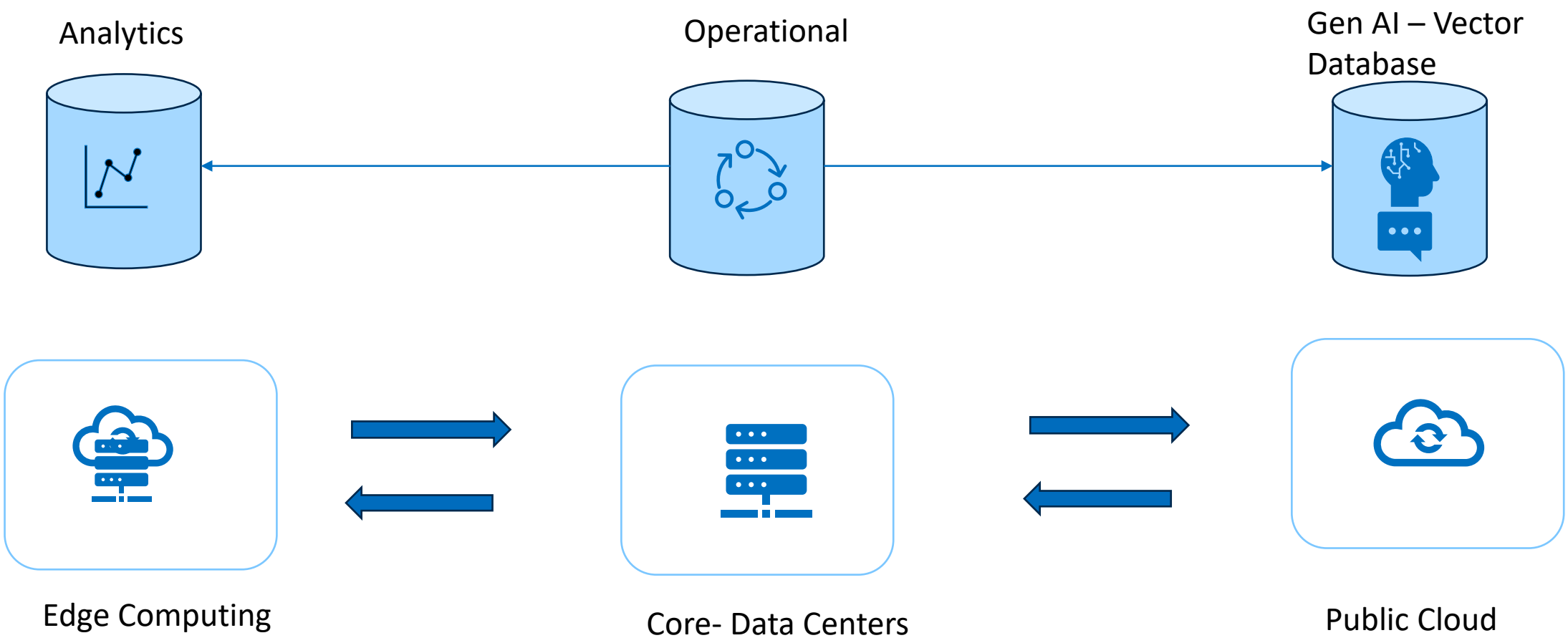
ScaNN : Technology Preview Mode

AlloyDB AI Enhancements



* Technology Preview Mode

Unified Infra & Data Strategy : AlloyDB Omni Solution



Please take a moment to rate this session.

Your feedback is important to us.