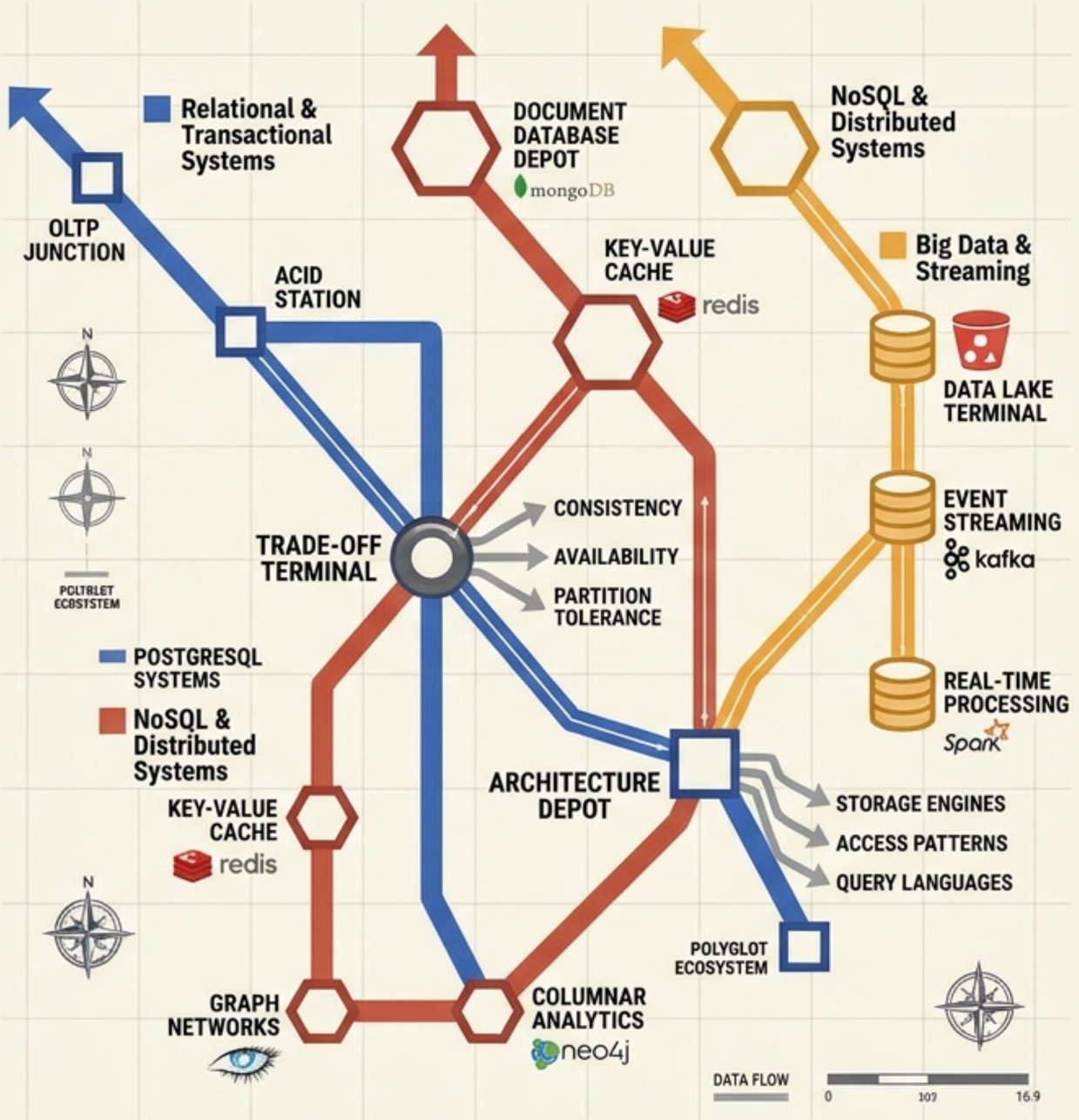
Modern Data Stores

Series Overview (Episode E00) – Navigating the Polyglot Landscape

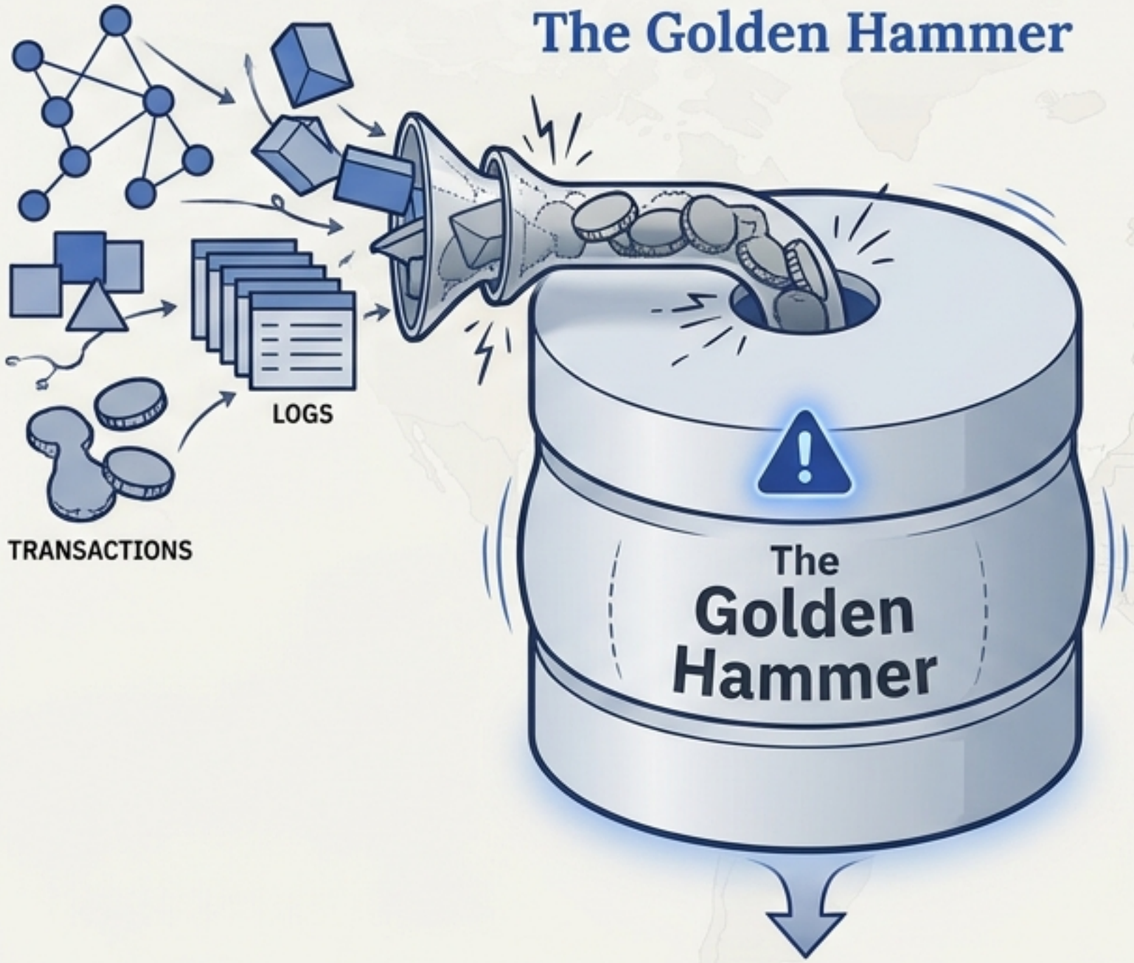
Choosing the right data store isn't about finding the 'best' database. It's about navigating strict trade-offs, understanding storage engines, and matching the right architecture to your exact access patterns.



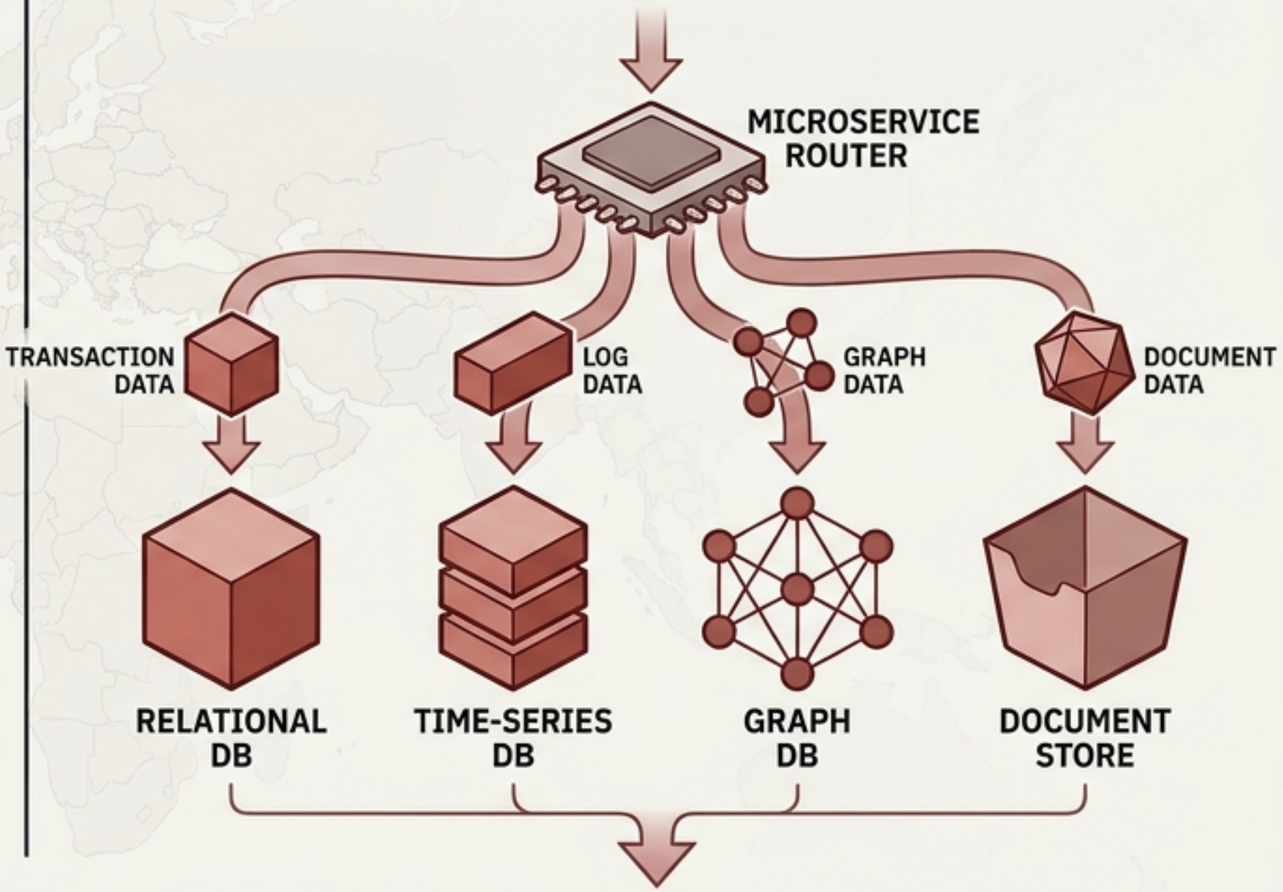
Gemini Notebook

The Era of the “One-Size-Fits-All” Database is Over

The Golden Hammer



Polyglot Persistence

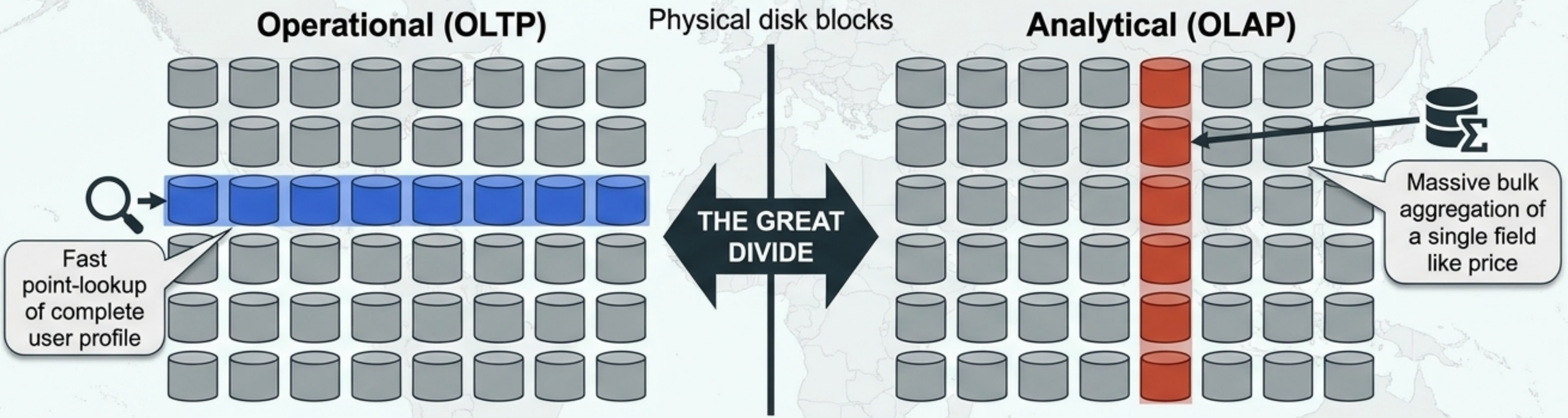


Approach	Primary Feature	Core Trade-off	It depends on...
“Monolith (Single DB)”	“Unified operational simplicity”	“Simplicity vs. Sub-optimal performance for niche queries”	“...whether operational simplicity outweighs the need for specialized query latency.”
“Polyglot (Multiple DBs)”	“Optimized access per data shape”	“Optimized access vs. Complex operational/sync overhead”	“...whether your engineering team has the maturity to monitor, backup, and govern multiple systems.”

Representative Implementations: N/A - Architectural Concept

Gemini Notebook

The Great Divide: How Are You Querying the Data?

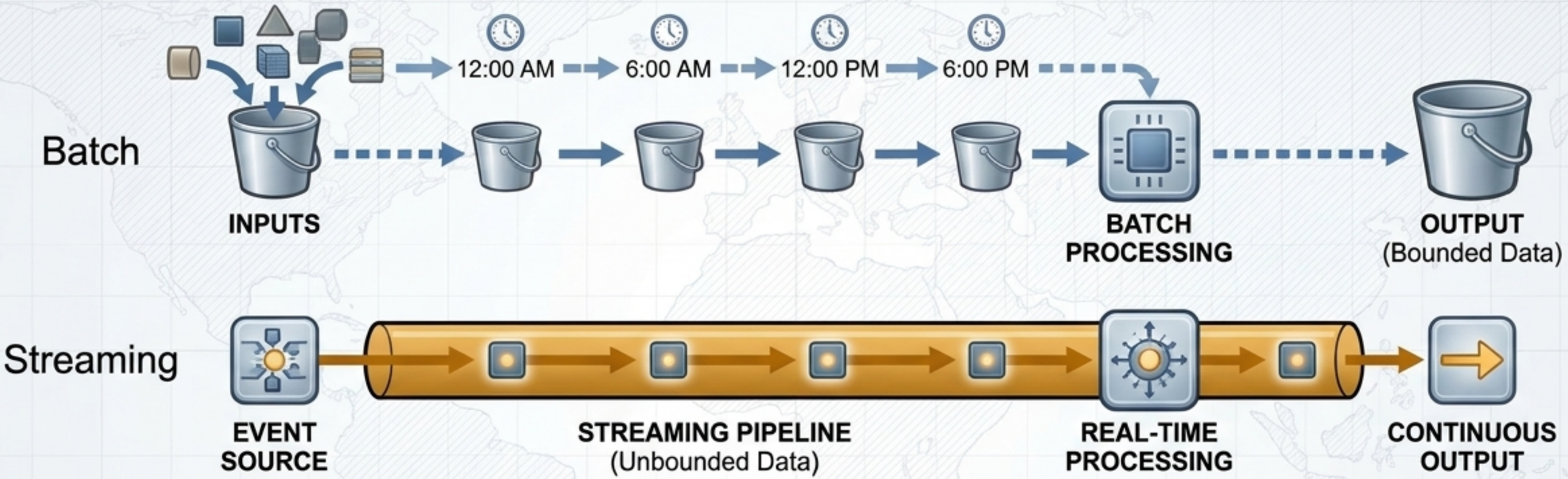


Dimension	Operational (OLTP)	Analytical (OLAP)	Trade-off	It depends on…
Goal	Strict, fast multi-entity transactions	Massive historical analytical scans	Point precision vs. Bulk aggregation	…whether the user is a customer checking out, or a data scientist finding trends.
Data Layout	Row-oriented	Column-oriented	Fast writes/updates vs. Fast selective reads	…whether your read concurrency outstrips your write concurrency.

Representative Implementations (Not Endorsements): Postgres, MySQL (OLTP) / ClickHouse, Databricks (OLAP)

Gemini Notebook

Processing Paradigms: When Does the Data Move?



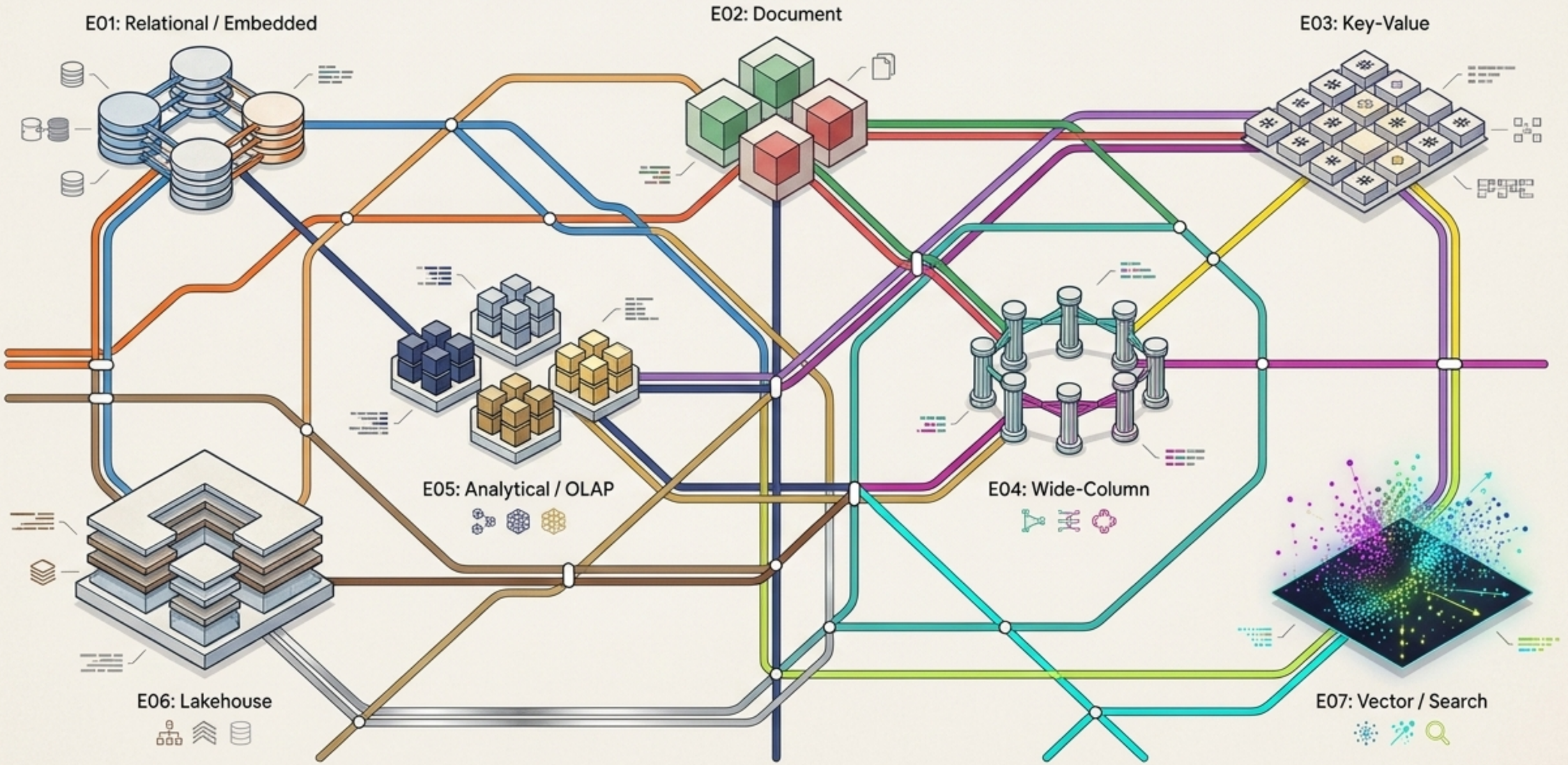
Paradigm	Characteristic	Core Trade-off	It depends on…
Batch	Scheduled, bounded processing (e.g., MapReduce)	High efficiency at scale vs. High data staleness	…whether downstream consumers can tolerate hours of latency for completeness.
Streaming	Continuous, unbounded event logs (e.g., Samza, Kafka)	Real-time reaction vs. Complex state/window management	…whether the business value of a sub-second reaction justifies the architectural complexity.

Representative Implementations (Not Endorsements): Hadoop MapReduce (Batch) / Apache Samza, Kafka (Streaming)

Gemini Notebook

Mapping the Modern Data Landscape

Over the next 7 episodes, we will dissect each family. Today, we survey the entire map to understand the territory.

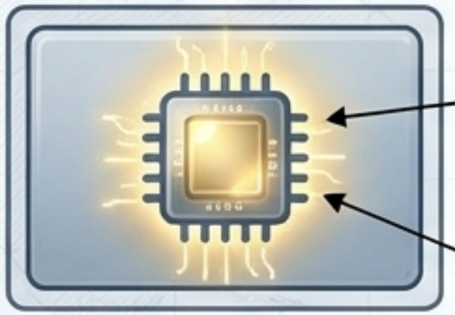


Gemini Notebook

Relational & Embedded: The Default Starting Point

Architectural Comparison

Embedded



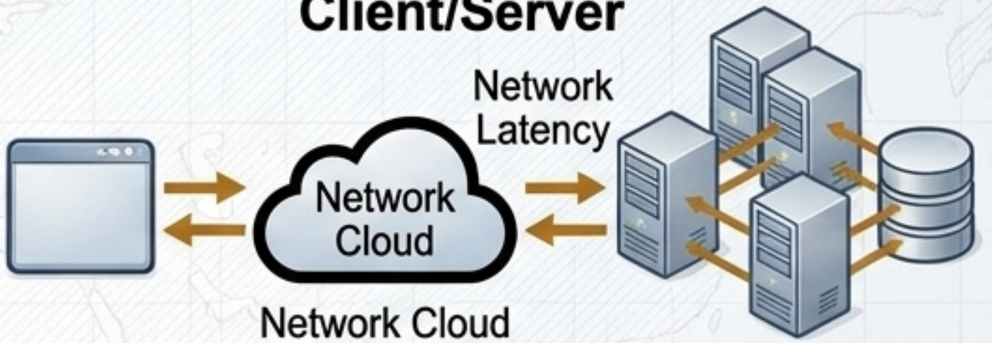
In-process embedded database

Seamless integration

Single Application Code Boundary

Representative Implementations (Not Endorsements): SQLite (Embedded)

Client/Server



Representative Implementations (Not Endorsements): PostgreSQL (Client/Server)

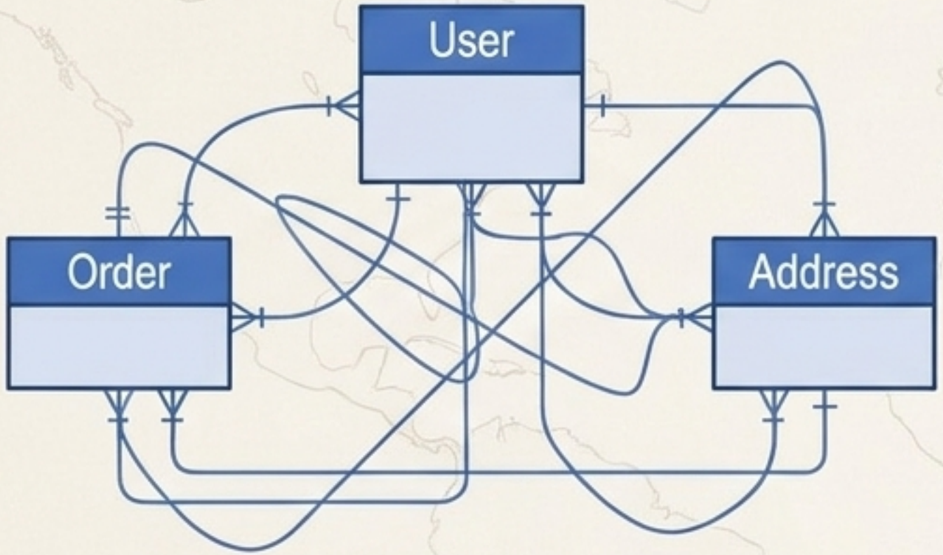
Architecture	Primary Strength	Core Trade-off	It depends on…
Embedded	Zero-config, runs directly in-process	Absolute simplicity vs. Single-writer concurrency limits	…whether the data is separated from the application by a network. If not, embedded is faster.
Client/Server	High write concurrency, centralization	High scalability vs. Network latency & admin overhead	…whether you have multiple concurrent writers or data creeping into the terabyte range.

Representative Implementations (Not Endorsements): SQLite (Embedded) / PostgreSQL (Client/Server)

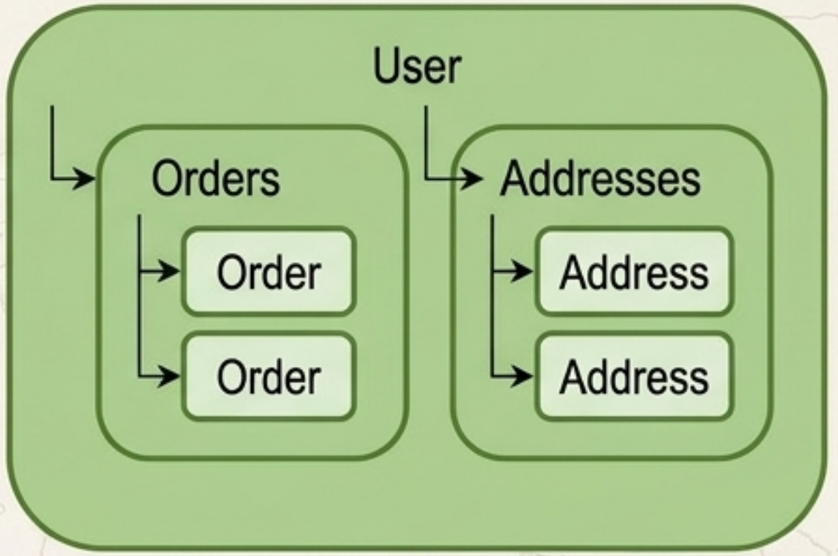
Gemini Notebook

Document Stores: Schema Flexibility and Aggregate Roots

Relational (Normalized ERD)



Document (Hierarchical JSON Tree)

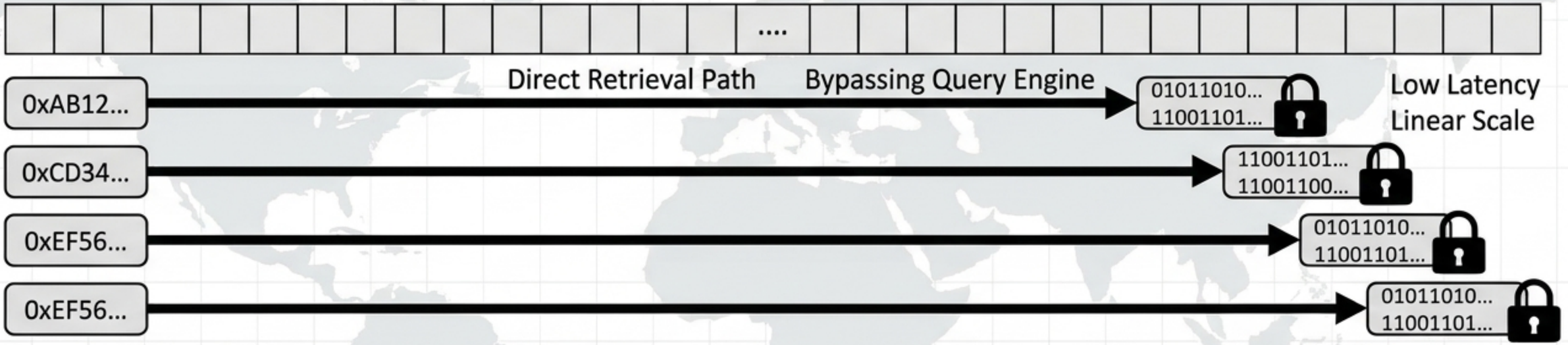


Aspect	Document	Relational	Trade-off	It depends on…
Schema	Schema-on-read	Schema-on-write	Rapid application evolution vs. Strict database-level data integrity	…whether your data structure changes frequently and aggregate objects are naturally accessed together.
Relations	Denormalized (Embedding)	Normalized (Foreign Keys)	Read performance (no joins) vs. Cost of partial updates/anomalies	…whether your application's data access requires deep, unpredictable relationship traversals.

Representative Implementations (Not Endorsements): MongoDB (Document)

Gemini Notebook

Key-Value Stores: Extreme Latency and Scale



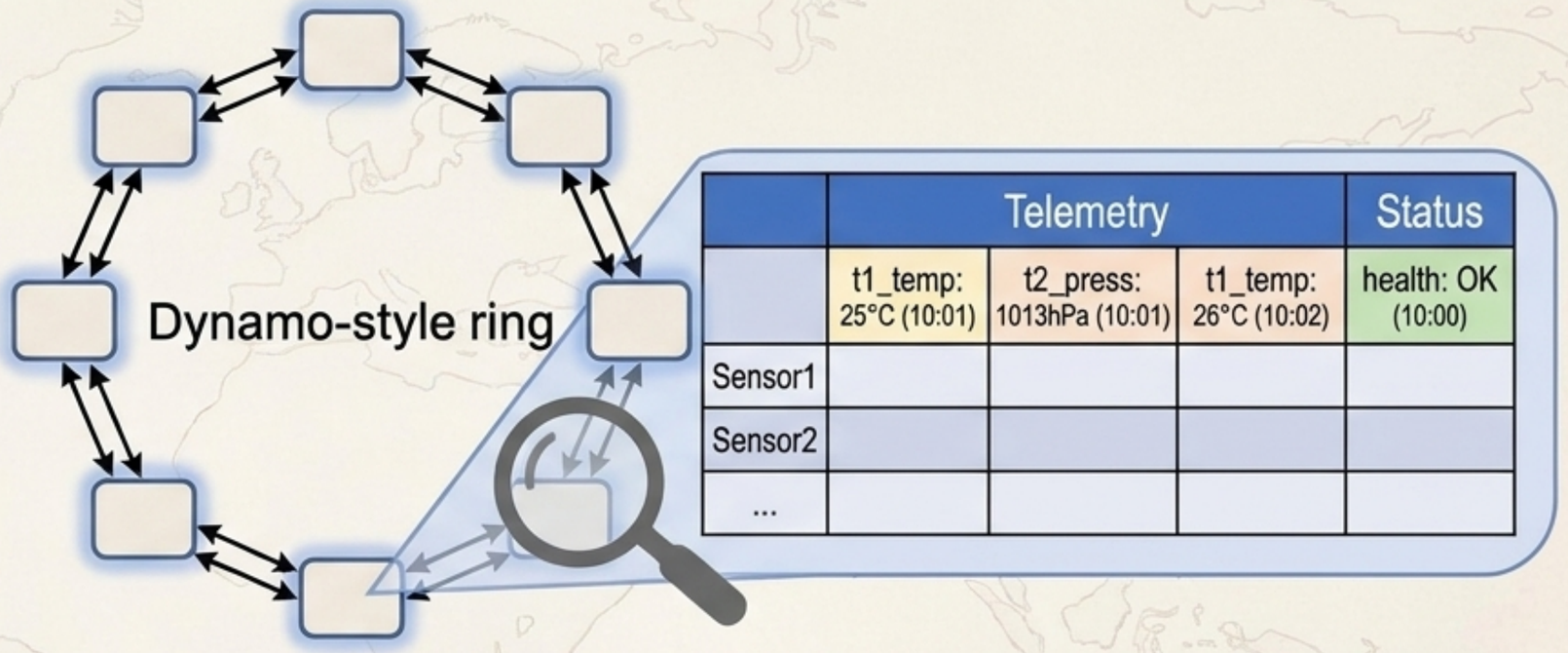
Architectural Comparison

Aspect	Key-Value	Document	Trade-off	It depends on...
Data Shape	Opaque blob accessed by a unique key	Structured, queryable fields	Extreme <1ms latency vs. Query expressiveness	...whether your workload strictly requires looking up data by a known ID (e.g., caching, sessions).
Updates	Full value overwrite	Partial field updates	Linear scale & simplicity vs. Network payload overhead on small edits	...whether you frequently mutate small parts of very large values.

Representative Implementations (Not Endorsements): Redis, Amazon DynamoDB (Key-Value)

Gemini Notebook

Wide-Column Stores: High-Ingest and Decentralization

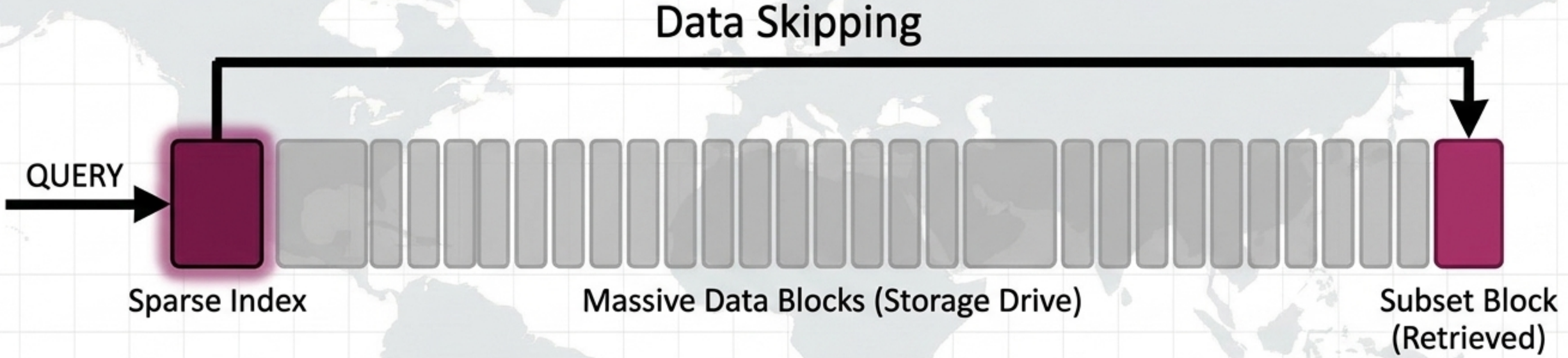


Aspect	Wide-Column	Relational	Trade-off	It depends on...
Topology	Decentralized, peer-to-peer ring	Primary / Replica	High availability & partition tolerance vs. Global strict consistency	...whether you must survive an entire datacenter outage without losing write availability.
Data Model	Sparse, dynamic column families	Rigid schemas	Optimized wide scans & write throughput vs. Ad-hoc join capability	...whether your queries are known in advance and rely on time-series/event-style tall data.

Representative Implementations (Not Endorsements): Apache Cassandra, Google Bigtable (Wide-Column)

Gemini Notebook

Columnar OLAP: Speeding Up Aggregations

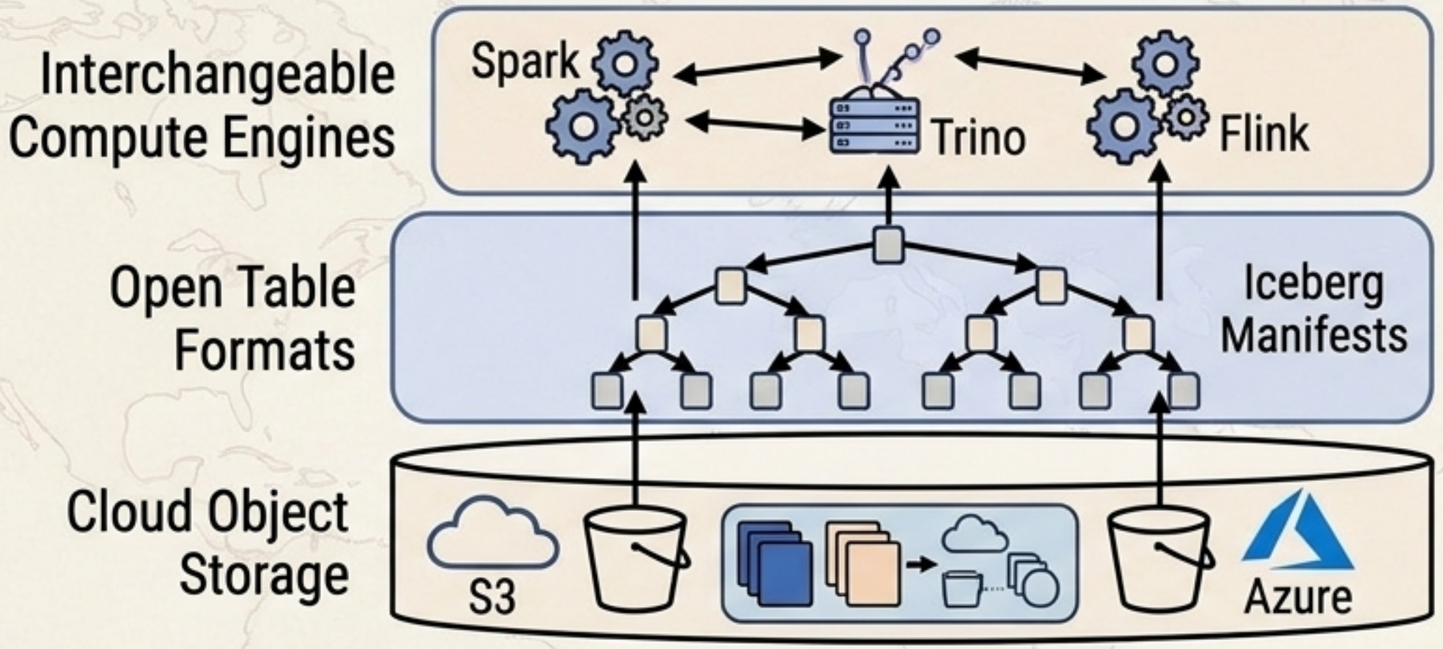


Aspect	Columnar OLAP	Traditional OLTP	Trade-off	It depends on...
Compression	Specialized codecs per column type	Generic block compression	Low disk footprint & CPU efficiency vs. Severe update penalty	...whether your workload consists of appending immutable analytical facts.
Indexing	Sparse indexes (Data skipping)	B-Trees (Point lookups)	Lightning-fast aggregations vs. Slow single-row retrieval	...whether a query retrieving a single specific row is an accepble anti-pattern for your app.

Representative Implementations (Not Endorsements): ClickHouse (Columnar OLAP)

Gemini Notebook

The Lakehouse: Unifying Lakes and Warehouses



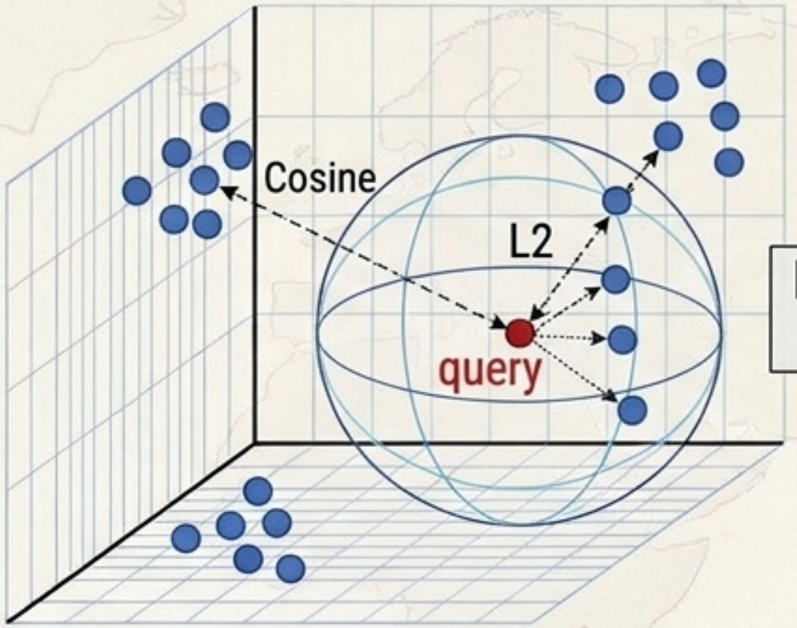
Aspect	Lakehouse	Data Warehouse	Trade-off	It depends on...
Storage	Open direct-access formats (Parquet)	Proprietary, managed storage	Engine flexibility (No vendor lock-in) vs. Turnkey performance optimizations	...whether you need multiple different processing engines to read the exact same data without copying it.
Metadata	Adaptive metadata trees (Manifests)	Internal catalog	Scalable small commits & time-travel vs. Setup complexity	...whether you require ACID transactions and row-level deletes directly on your data lake.

Representative Implementations (Not Endorsements): Apache Iceberg, Apache Parquet (Lakehouse)

Gemini Notebook

Vector Stores: Similarity and AI Context

3D Vector Space Graph



K-Nearest Neighbors (KNN) Search

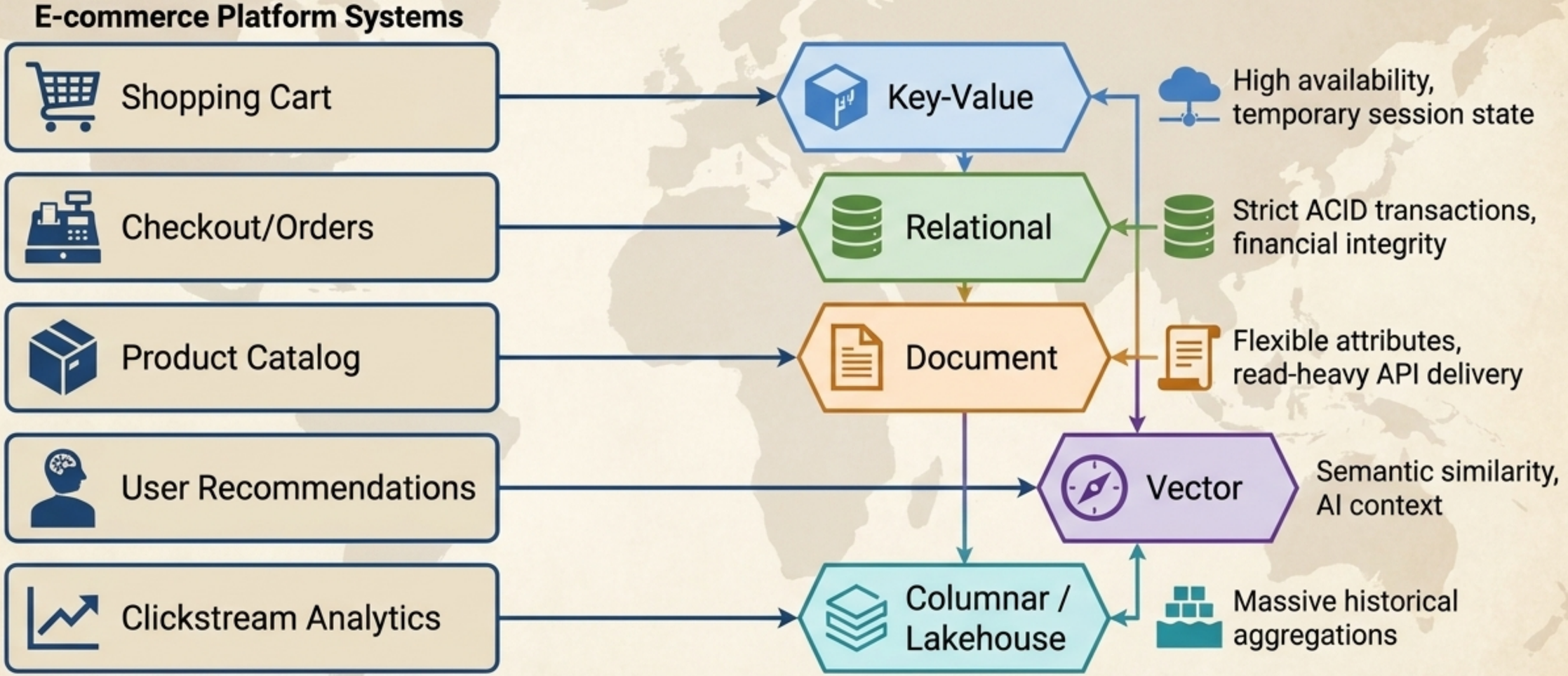
Index Type	Strategy	Core Trade-off	It depends on...
Exact NN	Scans every vector to find absolute closest match	Perfect recall vs. High query latency at scale	...whether your dataset is small enough that a full sequential scan meets your SLA.
Approx. NN (HNSW/IVFFlat)	Graph-based or clustered vector subsets	Millisecond search speed vs. Slight loss of accuracy (Recall)	...whether returning 'good enough' semantic matches instantly is more important than absolute perfection.

Representative Implementations (Not Endorsements): pgvector (Vector Store)

Gemini Notebook

Worked Example: Polyglot Persistence in Action

Do not build this on day one. Combine models only when access patterns or lifecycles clearly diverge and outgrow a single system's operational capacity.



Representative Implementations (Not Endorsements): Redis, PostgreSQL, MongoDB, Pinecone, Databricks Lakehouse

Gemini Notebook

The Jepsen Warning: Distributed Systems Realities

Network partitions will happen. Hardware will fail. Understand your store's consistency model before deploying.



Approach	Primary Guarantee	Core Trade-off	It depends on...
Strong Consistency (Linearizability)	Reads always return the latest write	Safety/Predictability vs. Total loss of availability during network partitions	...whether showing stale data to a user results in severe financial or legal consequences.
Eventual Consistency	System remains available during partitions	High Availability vs. Application complexity (handling aborted reads/stale data)	...whether you can design your application logic to gracefully tolerate eventual convergence.

Representative Implementations (Not Endorsements): Jepsen (Testing Tool)

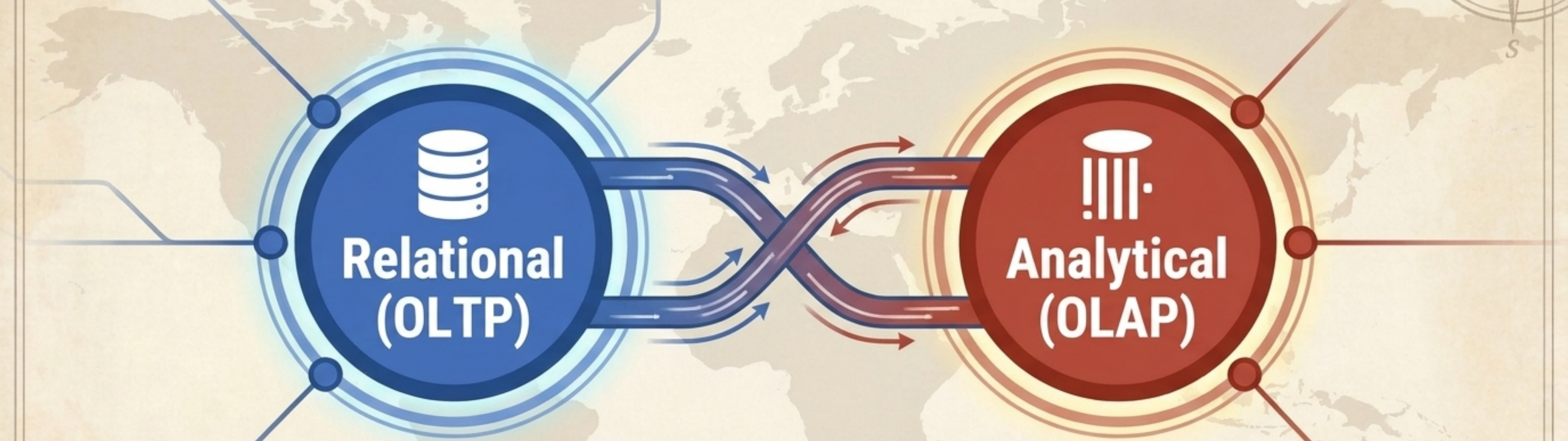
Gemini Notebook

The Data Landscape Matrix

Family	Primary Strength	Core Trade-off (What you give up)	It depends on...	Rep. Tech
Relational	Strict Transactions (ACID)	Rigid schema & horizontal scale limits	...needing data integrity above all else.	Postgres
Document	Developer Agility & Hierarchy	Complex join performance	...polymorphic data shapes.	MongoDB
Key-Value	Sub-millisecond Latency	Query expressiveness	...simple point-lookups at massive scale.	Redis
Wide-Column	Massive Write Throughput	Ad-hoc query flexibility	...decentralized, high-ingest telemetry.	Cassandra
Col. OLAP	Fast Bulk Aggregations	Single-row update speed	...scanning billions of immutable facts.	ClickHouse
Lakehouse	Open Format Interoperability	Fully managed, turnkey indexing	...decoupling your storage from compute engines.	Iceberg
Vector	Semantic/Similarity Search	Traditional keyword exact-match	...AI context and semantic proximity.	pgvector

Gemini Notebook

Coming Up: E01 – OLTP vs. OLAP Deep Dive



We've mapped the territory. Next time, we dive deep into the fundamental split in data engineering.

We'll explore internal storage engines, row-vs-column disk layouts, and when to bridge the gap between operational safety and analytical speed.

End of Episode E00. Prepare your SQL for Episode E01.

Gemini Notebook