influxdata®

How to Choose the Right Database for Your Application

Charles Mahler, InfluxData

Agenda

- Overview of the current database landscape and trends
- What makes databases perform differently?
- Overview of different database types
- Future Database trends/features
- Q/A

Database trends for 2024

- Multi-model databases
- Semantic layer becomes more important
- Focus on interoperability, integration, and common standards
- ML/AI enhanced databases

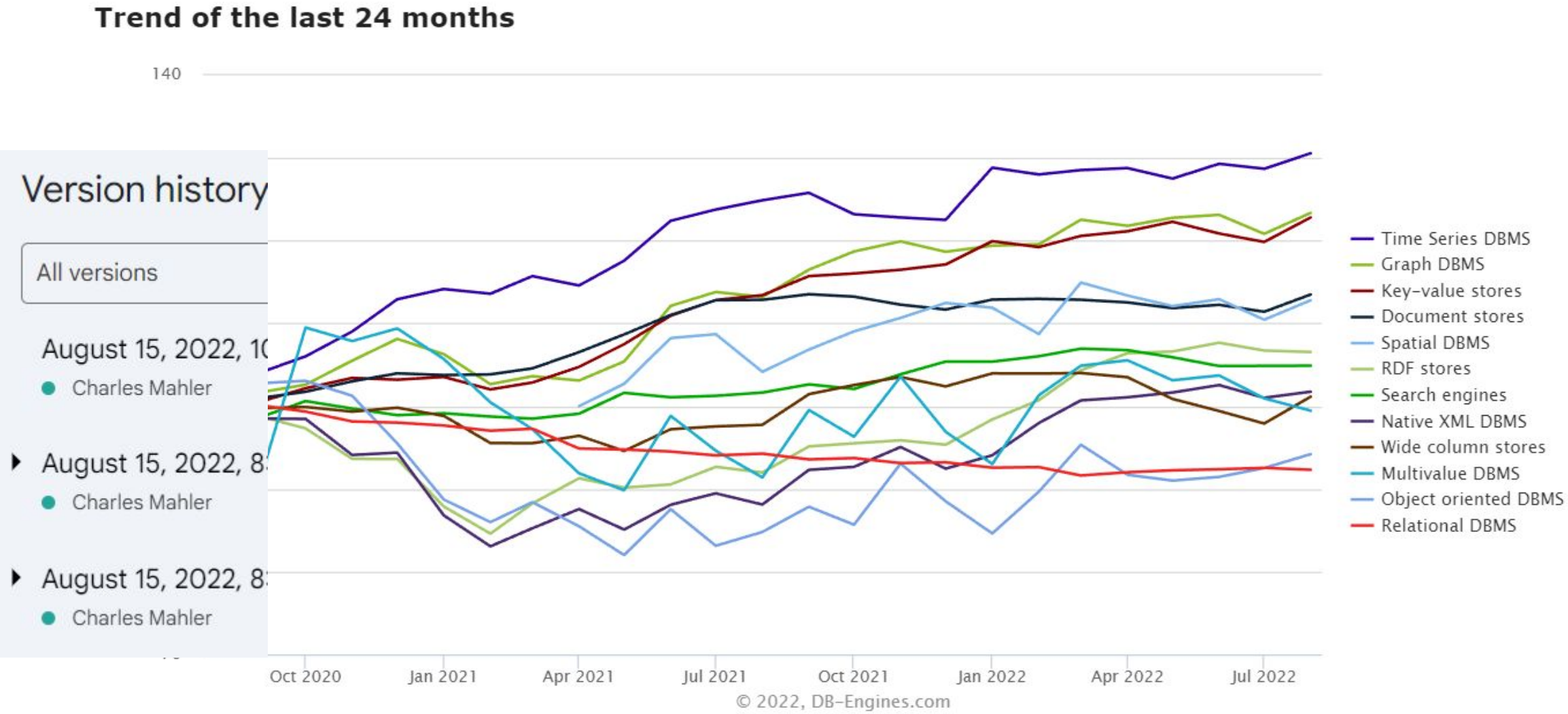
Things to keep in mind

- Don’t fall for hype- keep it simple
- Nothing is magical, always tradeoffs
- Many databases have overlap within categories and use cases
- Some generalizations being made for simplicity

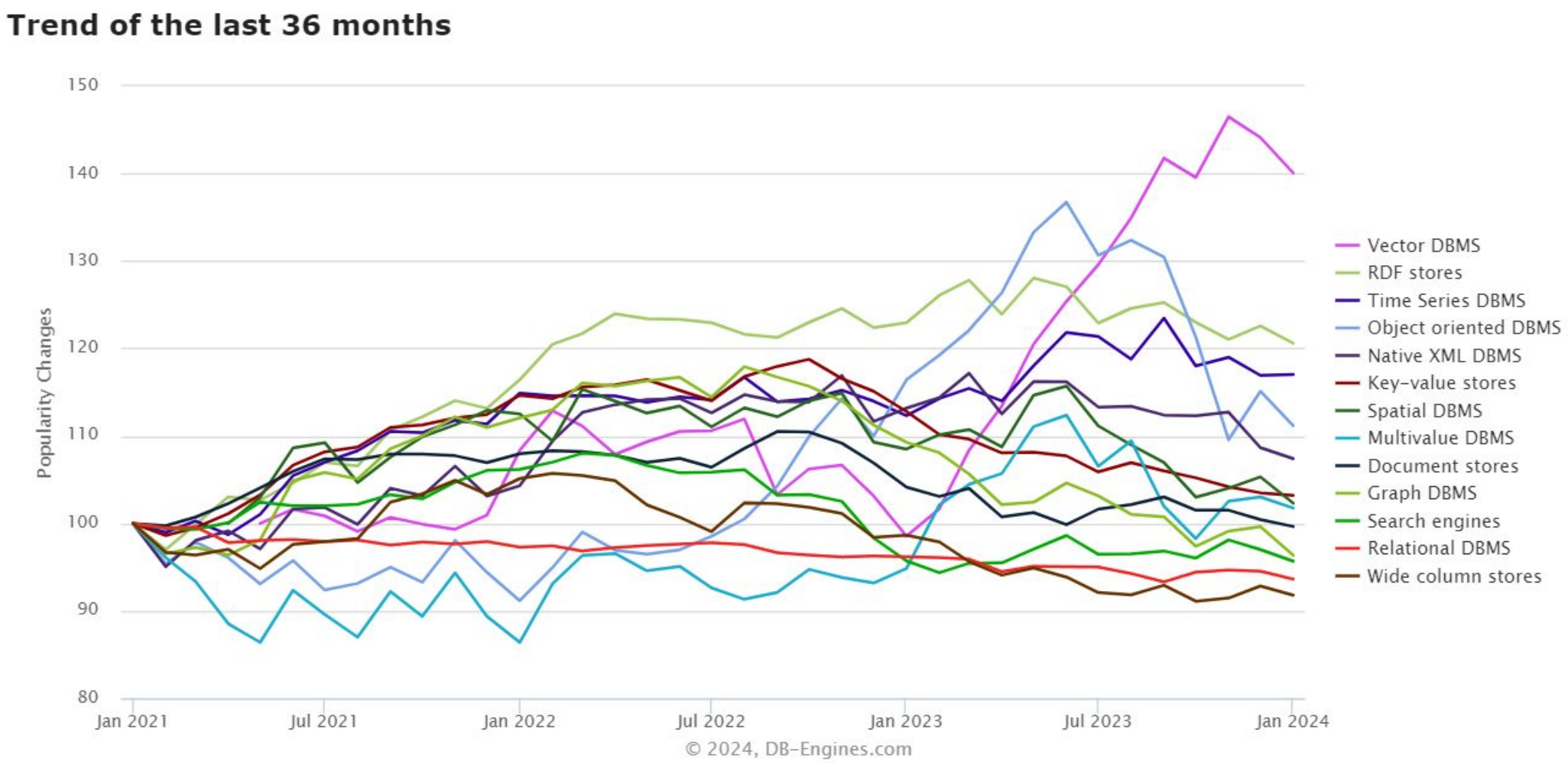
Types of Databases

Relational	Document
Key-Value	Graph
Time series	Columnar
In-Memory	Search
Vector	NewSQL

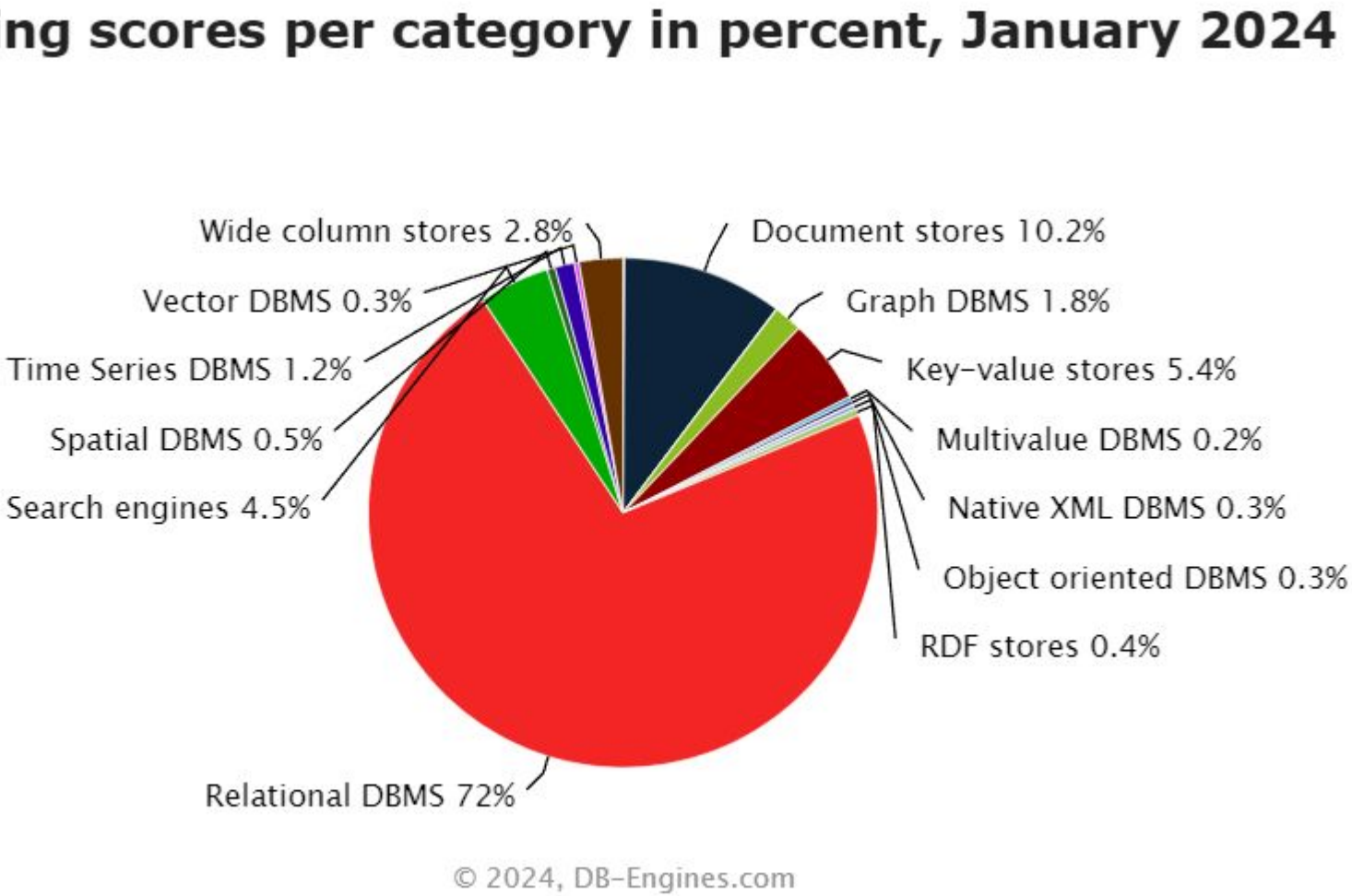
2022 version



Database Trends



Current Database Landscape



What makes databases perform differently?

Read vs Write

- The fundamental tradeoff in database performance

Database Design Considerations

- On disk data format
- Indexing data structure
- Disk-based vs memory-based
- Compression
- Hot vs Cold data
- Durability/Recovery

How to Choose the Right Database

- Data access patterns
- Transaction isolation requirements
- Scalability requirements
- Business stage
- In-house knowledge

Relational vs NoSQL characteristics

- B-tree index
- Table based structure
- Defined schema
- Strong consistency- ACID
- Vertical scaling

- LSM tree index
- Multiple types of storage
- Flexible schema
- Consistency tradeoffs- BASE
- Horizontal scaling

2024 trends - Multi-model databases

- Relational with column storage extensions
- Relational/graph hybrids
- Data lakehouses
- Vector embedding support

Relational Databases

Relational Database Overview

- Data follows relational model and is stored in tabular form
- Stored on disk as rows
- SQL used for queries
- Examples- PostgreSQL, MySQL

Relational Database Pros and Cons

- Solid performance across broad range of applications
- Strong ecosystem and battle tested
- Data consistency

- Challenging to scale horizontally
- Can struggle with high write volume workloads
- Defined schema

Relational Database Use Cases

- Suitable for most general applications
- Any application where ACID is required

Key-Value Databases

Key-Value Database Overview

- Simplest type of NoSQL database- effectively a hash table
- Keys point to unstructured blob of data
- Examples- DynamoDB, Redis, Riak

Key-Value Database Pros and Cons

- Great read and write performance
- Flexible schema
- Scalable

- Weak consistency
- Minimal querying capabilities

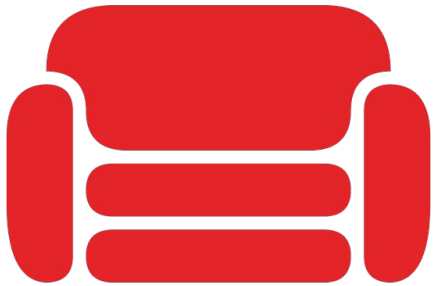
Key-Value Database Use Cases

- Personalized recommendations
- Session management
- Real-time features

Document Databases

Document Database Overview

- Extension of key-value database with added support for metadata
- Semi-structured data format
- Documents hold all relevant data rather than relying on joins
- Examples - MongoDB, CouchDB



Document Database Pros and Cons

- Flexible Schema
- Performance
- Data locality
- Documents can map better to application needs
- Scalable

- Consistency
- Relationships between documents

Document Database Use Cases

- Suitable for most general applications
- Applications where rapid iteration is beneficial

2024 trends - Semantic layer

- More user friendly way to represent data
- With rise of AI/LLMs, semantic layer is even more important
- Will enable non-technical subject matter experts to generate insights

Time Series Databases

Time Series Database Overview

- Designed for time series data
- Must be able to handle queries at both ends of the database spectrum
- Optimized for high write throughput and querying based on time ranges
- Built in features for data lifecycle management
- Examples- InfluxDB, TimescaleDB, QuestDB

Time Series Database Pros and Cons

- Very fast data ingest and query performance
- Developer productivity- data retention, aggregations, query languages

- Update and deletion performance

Time Series Database Use Cases

- Monitoring
- IoT applications
- Financial data
- Analytics and event tracking

Columnar Databases

Columnar Database Overview

- Data is stored in column format rather than row
- Column format has benefits for compression and processing speed
- Used heavily for analytics workloads
- Examples- Clickhouse, Vertica, Redshift

Columnar Database Pros and Cons

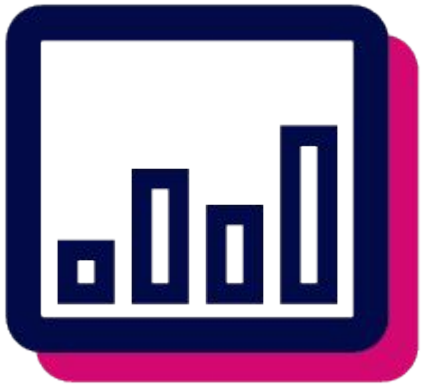
- More efficient for analytic type queries
- Better data compression
- SIMD processing

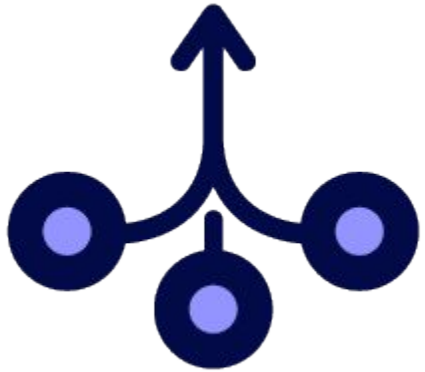
- Writing data can be less efficient
- Slower if accessing multiple column values

Performance

Columnar Database Use Cases

- Analytics
- Data warehousing
- Observability





Case Study - Uber logging data

3x

better compression

10x

better query performance

50%

reduction in hardware costs

Graph Databases

Graph Database Overview

- Used for storing and analyzing relationships between connected data sets
- Specialized query languages
- Native vs non-native graph storage
- Examples- Neo4J, DGraph

Graph Database Pros and Cons

- Fast queries for graph analysis
- Developer productivity - query language, built-in algorithms
- Flexible schema
- Easy to establish new relationships between data points

- Not ideal if application data isn’t highly connected

Query comparison

```sql
SELECT director.name, count(*)
FROM person keanu
  JOIN acted_in ON keanu.id = acted_in.person_id
  JOIN directed ON acted_in.movie_id = directed.movie_id
  JOIN person AS director ON directed.person_id = director.id
WHERE keanu.name = 'Keanu Reeves'
GROUP BY director.name
ORDER BY count(*) DESC
```

```
MATCH (keanu:Person {name: 'Keanu Reeves'})-[:ACTED_IN]->(movie:Movie),
      (director:Person)-[:DIRECTED]->(movie)
RETURN director.name, count(*)
ORDER BY count(*) DESC
```


Graph Database Use Cases

- Fraud detection
- Social networks
- Recommendation features

2024 trends - Interoperability

- Apache Arrow ecosystem
  - FDAP stack
- Data lake/lakehouse ecosystem
  - Iceberg, Delta Lake formats
- Query engines

In-Memory Databases

In-Memory Database Overview

- Databases where data is stored entirely in RAM
- Can use fully optimized data structures without tradeoffs for interacting with disk
- Examples- Redis, Memcached

redis

In-Memory Database Pros and Cons

- High performance
- Low latency
- Versatile data types

- RAM is more expensive than disk
- Large data sets will need to be scaled horizontally
- Secondary database needed for persistence in most cases

In-Memory Database Use Cases

- Caching
- Real-time applications
- Pub/Sub

Search Databases

Search Database Overview

- Designed for efficiently storing and querying text-based documents
- Can handle structured or unstructured data
- Examples- Elasticsearch, Solr, MeiliSearch

Search Database Pros and Cons

- Developer Productivity - Built in algorithms, query languages
- Performance
- Scalability

- With high write volumes you may need to sacrifice indexing or replication

Search Database Use Cases

- Full-text search
- Log analysis
- Autocompletion
- Analytics

Vector Databases

Vector Database Overview

- Designed for storing and searching vector embeddings of unstructured data
- Allows for searching of images, videos, text, audio, etc.
- Examples- Milvus, Pinecone, Weaviate, Qdrant

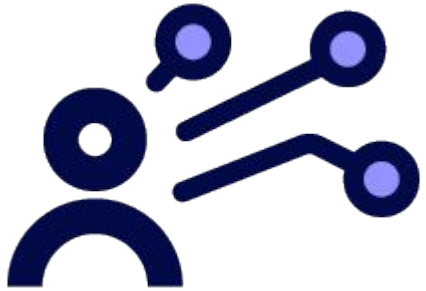
Vector Database Pros and Cons

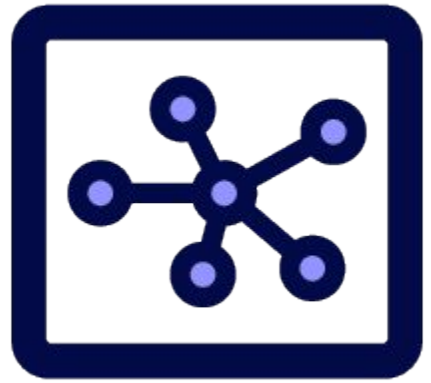
- Efficient vector search
- Scalability
- Hybrid storage

- Very specialized database, overhead might not be worth it

Vector Database Use Cases

- Duplicate removal
- Retrieval Augmented Generation(RAG)
- Anomaly detection
- Ranking and recommendation engines
- Similarity search for unstructured data
- Semantic search





NewSQL Databases

NewSQL Database Overview

- Attempt to get the best of both worlds by taking things from both relational and non-relational databases
- Capable of high consistency while maintaining high availability
- Examples- CockroachDB, Spanner, TiDB

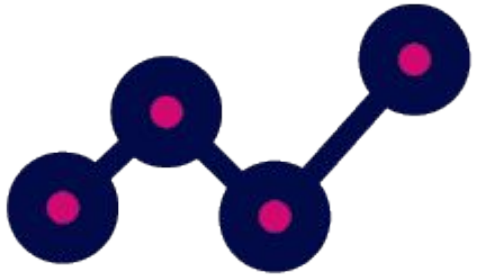
NewSQL Database Pros and Cons

- SQL support
- Horizontal scalability
- Typically “cloud native” architecture friendly

- Complexity, even if hidden
- Potential latency
- Some SQL limitations
- Very new technology

NewSQL Database Use Cases

- Work for most general applications like a normal relational database
- Some can also support analytics workloads



2024 trends - ML/AI enhanced databases

- ML Indexing/Optimization
  - 2-3x faster reads
  - Index several orders of magnitude smaller
  - Indexing recommendations and automation

Get Help + Resources!

Forums: community.influxdata.com

Slack: influxcommunity.slack.com

Github: github.com/InfluxCommunity

Docs: docs.influxdata.com

Blogs: influxdata.com/blog

InfluxDB University: influxdata.com/university