www.kdb.ai

KDB.AI Powered by KX

A Comprehensive Guide to Vector Databases

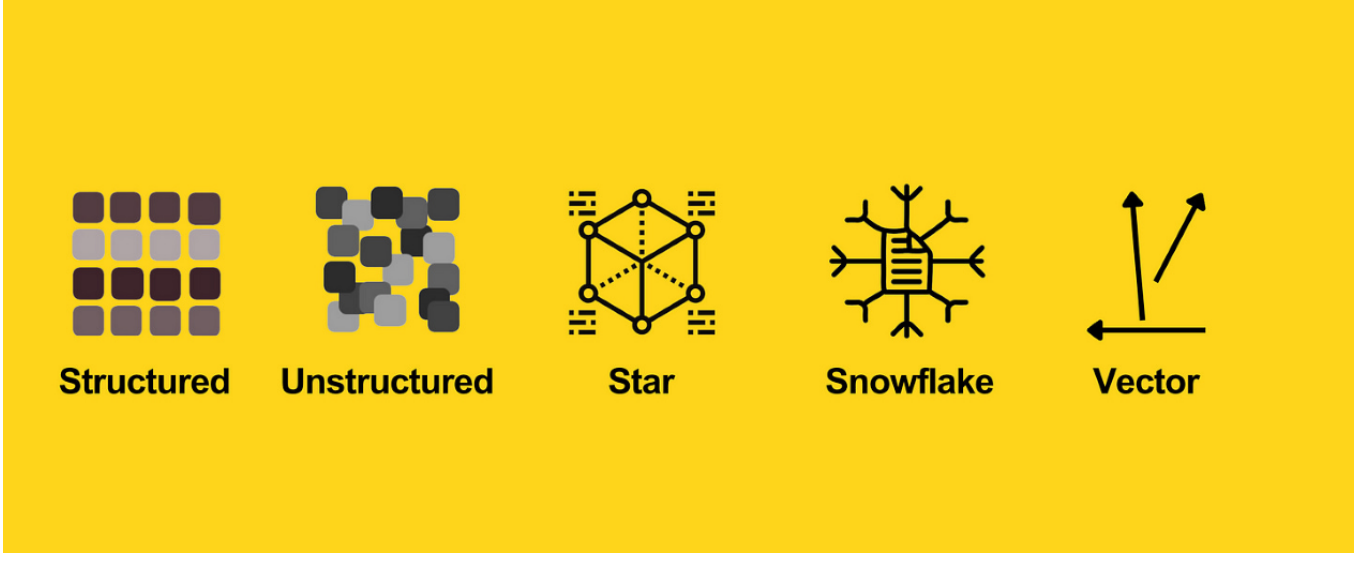
Vector databases are a new wave of data management designed for generative AI, IoT and time-series applications.

Table of **Contents**

Vector Databases Use Cases

Vector databases are a new wave of data management designed for generative AI, IoT and time-series applications. Here's why they matter, what makes them different, how they work, the new use cases they're designed for, and how to get started.

Over sixty years ago, databases became a crucial part of computing along with application logic, user interfaces, and middleware. With each new generation of technology, new database tools rose to prominence to capture, query, update, and manage information. Relational databases were motivated by accounting, the internet drove growth in unstructured and document stores, and online analytics created a need for STAR and Snowflake systems.



The latest innovation curve in database technology is vector databases, driven by data science applications that leverage generative AI for similarity search, anomaly detection, and temporal data. These applications rely on data stored as vectors – objects represented as numeric values in space and time – a task that previous generations of systems were not designed for.

For instance, vectors are an efficient way to represent words in sentences or model relationships between multimedia data such as text, images, audio, and video. Vectors are also useful in expressing the temporal order of readings from IoT sensors, automated operations, and digital applications. While some attempt to fit vectors into conventional stores, this approach is neither efficient nor optimal.

Vector databases are built specifically for the unique structure of vector and temporal data. Thanks to their unique physics, they typically process workloads 100 times faster than traditional stores and at a fraction of the cost.

Finally, since their storage format matches how programmers think, application logic can be developed more quickly and efficiently. As Steve Jobs once proclaimed, the most efficient, elegant, bug-free code is the code you don't write at all. A direct mapping between structure and data helps developers express logic at the speed of thought.

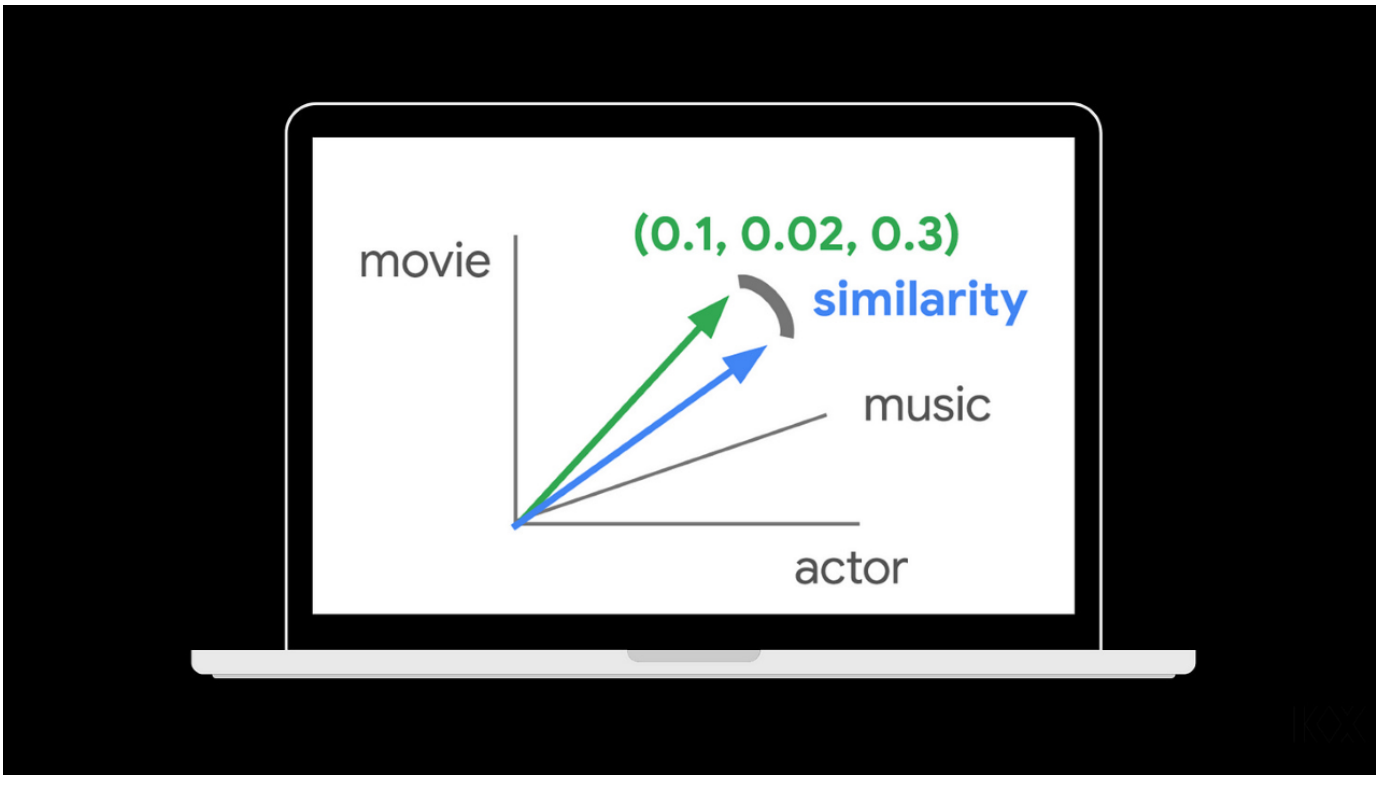
This article provides an introduction to vector databases, including how they work, the business drivers behind them, important use cases, and core attributes that lead to success. So let's discover the wonderful world of vector databases and take the first step towards better data storage.

What is a Vector Database?

A vector database is a specialized DBMS that stores vector embeddings utilizing innovative techniques for storage, indexing, and query processing. They offer data management capabilities, such as CRUD and language bindings to widely used data science languages such as Python, SQL, Java, and Tensorflow. Additionally, they deliver advanced features such as high-speed ingestion, sharding, and replication.

Vector databases are designed to handle critical query and algorithmic styles seen in similarity search, anomaly search, observability, fraud detection, and IoT sensor analytics. Such emerging styles are the outcome of digital transformation and the rise of generative AI.

Google uses similarity search to find and suggest personalized content, be it music or movies, based on a user's interests and those of others. The importance of similarity search extends beyond Google to all applications that recommend products, detect fraud by identifying patterns of access, spot similarities among images, and even detect data quality issues.



The market for vector databases is in its early stages. Established enterprise leaders in time series such as KX have entered the market, while startups like Milvus, Pinecone, Weaviate, Vald, Deephaven, and Qdrant are receiving early-stage venture capital. In this paper, we'll explore KDB.AI as the prototypical vector database implementation.

The Foundational Vector Database Use Case: Similarity Search

Imagine your application aims to suggest movies to users. You have a colossal dataset of half a million flicks and millions of users. How would you tackle this challenge?

Traditional data stores require each movie and user be manually tagged with metadata to identify common characteristics: Mark enjoyed animated movies, andFinding Nemo belongs to the animated genre, hence Mark might appreciateFinding Nemo.

Vectors are a superior solution to this problem. Rather than relying on manual tagging, vector embeddings encode a depiction of items, allowing for easy and rapid comparison. Google shows how to use vectors in their foundational machine learningcourse.

Let's say Chad recently watched and liked a bunch of movies, like Shrek, The Incredibles, Harry PotterMemento and others. Our mission is to recommend other movies that Chad will love based on this "profile" of what Chad likes:


Shrek


Incredibles

The Triplets of Belleville


Harry Potter


Star Wars


Bleu

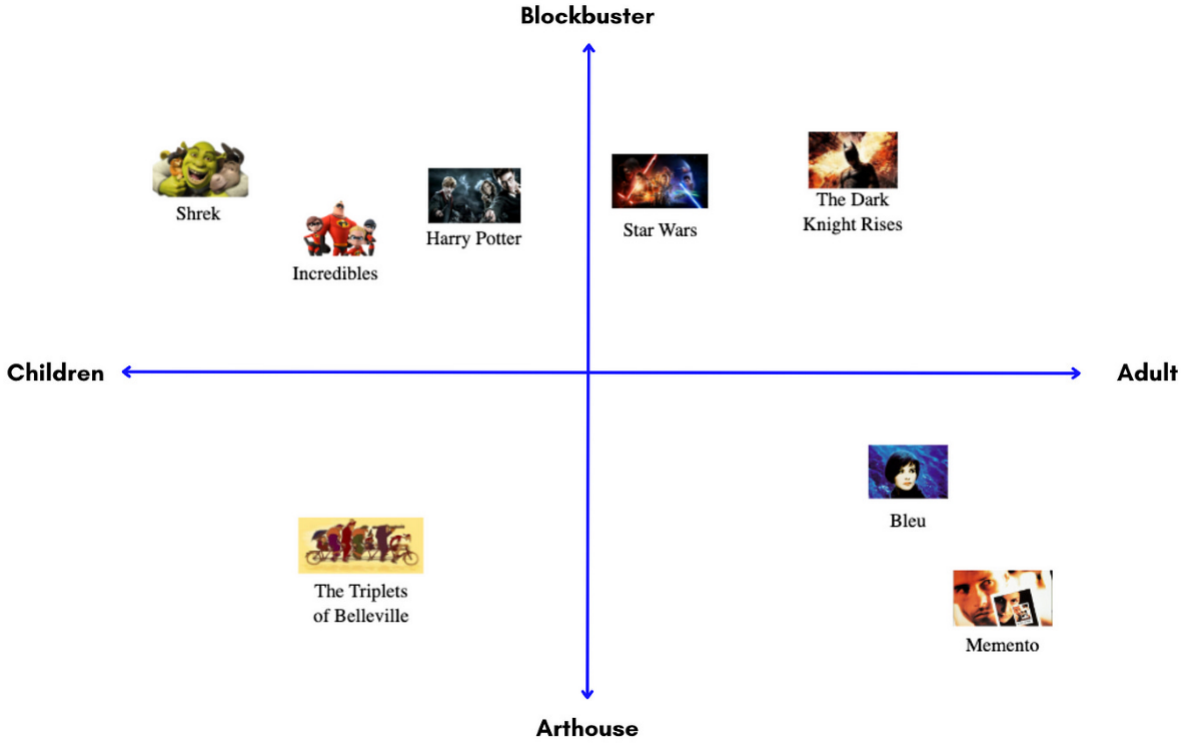

The Dark Knight Rises


Memento

Although some of his picks fit into the same genre, such asThe Incredibles andShrek, others such as Mementoand The Dark Knight Risesdon't. The variety in the movies' genre renders deciphering his rationale for choosing them a bit tricky.

The task at hand is right up machine learning's alley. It excels at tackling challenges like this, using its advanced algorithms to provide solutions.

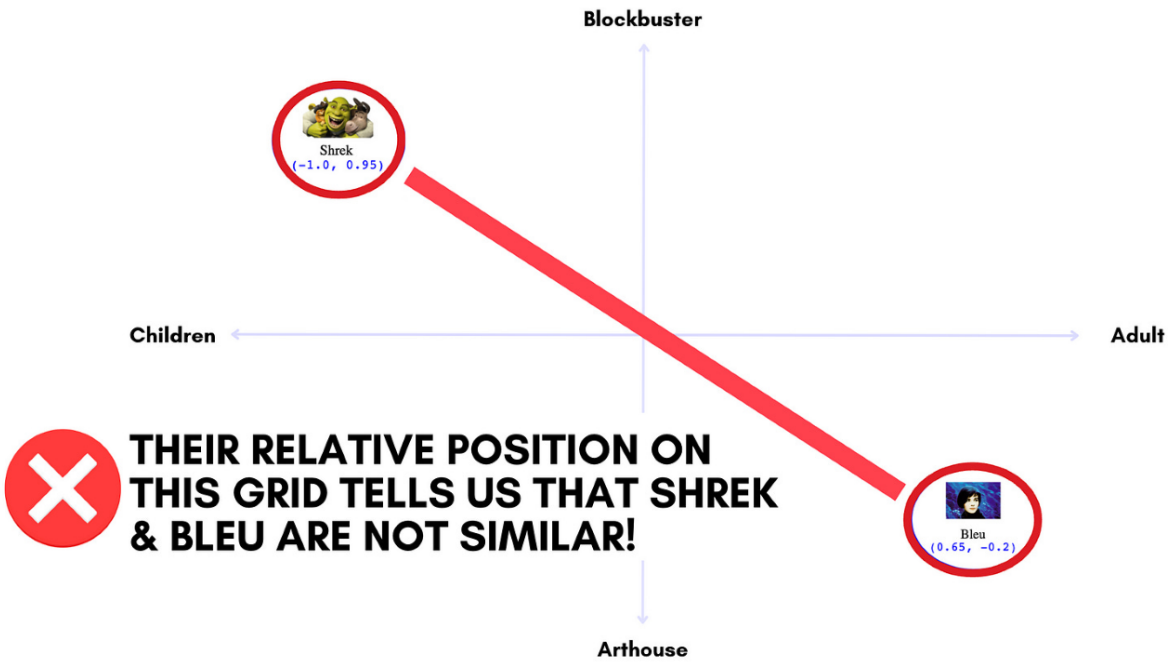
Let’s begin with a mental image of our movie classification grid. A simple method to categorize movies is by their type – blockbuster or arthouse, adult or children-oriented – plotted on a spectrum (below). This method reveals thatShrekand Mementoare distinctly disparate.



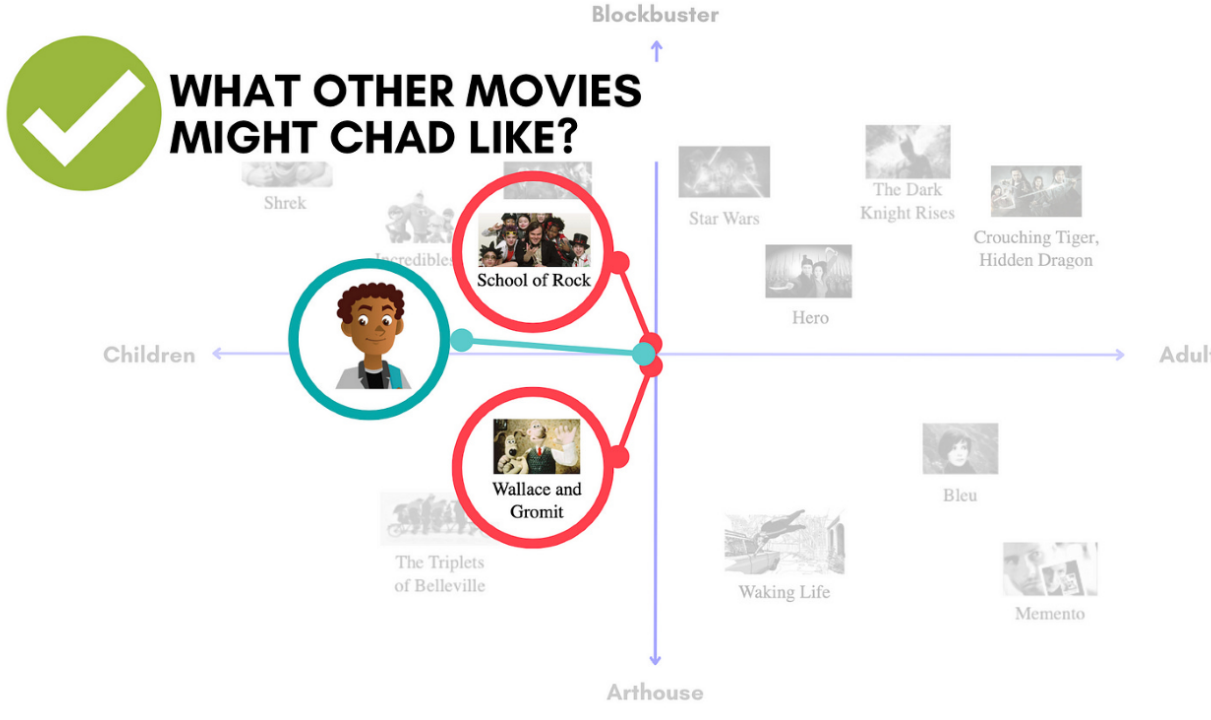
While this is easy to see on this simple picture, it’s harder to make a determination like this based on 500,000 movies and 1,000,000 users.

Vector databases are well-suited for this purpose. They simplify encoding of many dimensions, such as style, story narrative, tone, topic, and demographics. This facilitates swift search and recommendation of new movies to Chad based on the distance between entities in this space.

For example, by placing our 500,000 movies on this grid, assigning a coordinate to each movie, we can quickly compute that these two movies, Bleu andShrek, are dissimilar.



To ensure Chad receives tailored recommendations, we analyze the dimensions of his interests using movies he likes such asThe Incredibles,Star Wars, andThe Triplets of Belleville. By searching for movies that are similar, likeSchool of Rockand Wallace and Gromit, we can confidently suggest these titles.



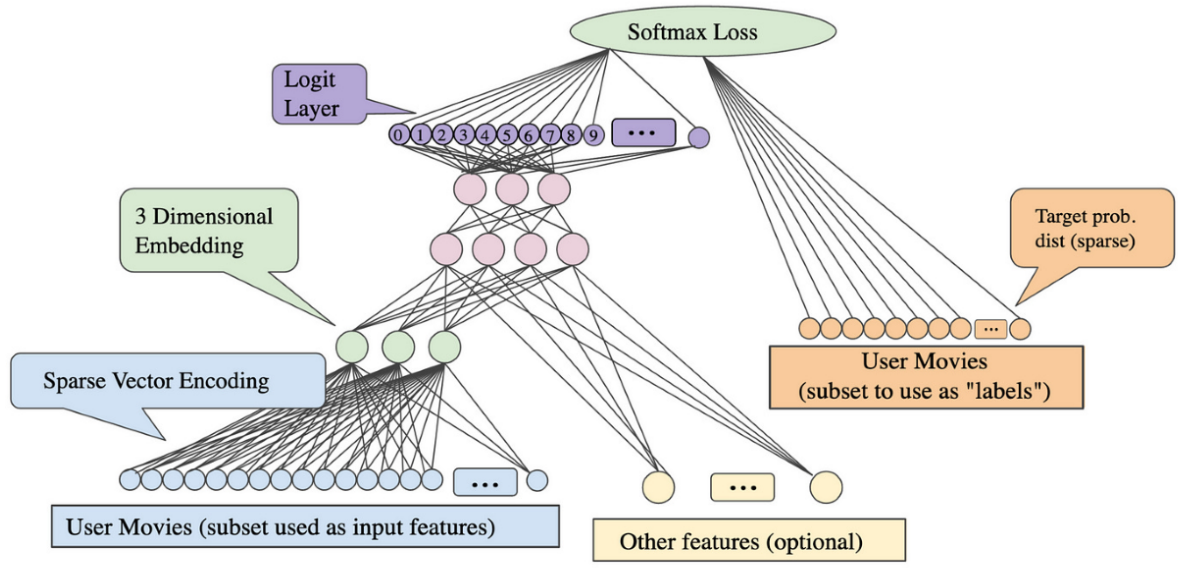
Modern companies serve countless customers and lots of products. The ability to quickly identify similarities is paramount for cutting-edge digital applications such as customer experience, fraud detection, and product recommendation engines.

Using Neural Networks to Automate Vector Embeddings

Classifying movies and their audiences onto a 1,000-grid system would be an impossible task using manpower alone. Fortunately, AI and machine learning excel at this. Neural networks enable us to take highly educated guesses for placement of millions of items on the grids that we need to analyze.

Vector Embeddingis the process of representing items as numerical vectors that can be effectively processed by machine learning algorithms. This technique maps items in a high-dimensional vector space to position similar items in close proximity and dissimilar ones farther apart. This approach significantly improves the efficiency of machine learning algorithms.

As an example, to gauge a movie's suitability for children, we can utilize the vector space mentioned earlier. With machine learning algorithms, explained in a Google education videoand showcased below, we can automate the assignment of numerical values to these aspects using neural networks.

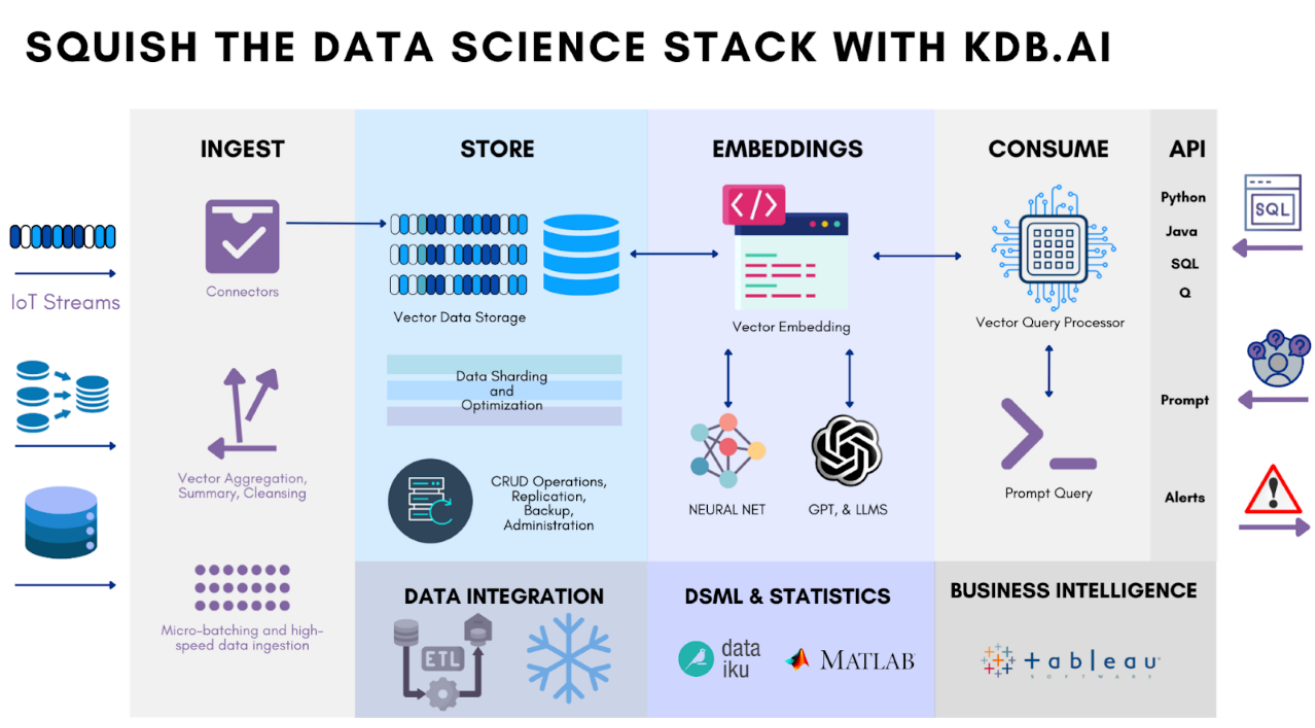


As an example, to gauge a movie's suitability for children, we can utilize the vector space mentioned earlier. With machine learning algorithms, explained in a Google education videoand showcased below, we can automate the assignment of numerical values to these aspects using neural networks.

Squish the Stack with KDB.AI

KDB.AI, the vector database from time series pioneer KX, launched in 2023. It's explicitly designed for cloud-native vector data management, vector embeddings, and fast-paced engineering tasks, as seen in GPT-style query exploration.

The KX stack is a comprehensive solution that dramatically streamlines traditional data science workflows. By bundling advanced connectivity, vector encodings, built-in algorithms, and powerful data organizational tools – all with support for Python, Java, SQL – the KX stack "squishes the stack," reducing dependencies, accelerating processing, and simplifying data exploration.



The data pipeline usually relies on a combination of tools and technologies to process and store data in traditional formats. However, KDB.AI streamlines the process by amalgamating a diverse range of capabilities into a singular engine. This mechanism effectively revolutionizes conventional techniques of structured and unstructured data processing, by facilitating data processing into vectors.

It has five main phases:

01

INGEST

To deploy KDB.AI, the initial step is to ingest data from external databases, ETL, or streaming data sources through native connectors in the platform. This data is then aggregated, summarized, and cleansed for preparation and organization by the data store. This facilitates your deployment process, enabling efficient integration and management of external data and providing a sound structure for the storage of your insights.

02

STORE

KDB.AI is engineered to encode and compactly store vector data structures, optimizing real-time streaming input ingestion, prompt-based queries, and the execution of business intelligence tools. Fast queries that execute with “thought-like” speed are the result, making KDB.AI the optimal solution for businesses that require real-time analytics.

03

EMBEDDINGS

KDB’s built-in algorithms encode vector embeddings into the database, drastically enhancing storage efficiency for accelerated query support and real-time operations. The KDB store provision of optimized CRUD operations for vector data is further augmented by GPT, neural net-style algorithms, and other designed optimization capabilities.

04

CONSUME

The Consume phase supports Python, Java, SQL, and Q queries, while accommodating new prompt interfaces. These interfaces enable interactive questions on vector embeddings, handling complex queries adeptly from a simple command prompt.

05

INTEGRATION

Effective integration tools are critical when it comes to database optimisation. Complementary tools like Informatica, Dataiku, Matlab, Power BI, and Tableau are necessary to perform tasks such as data management, modeling, analytics and visualization with fluidity. At KDB.AI, the prospect of using such tools for data integration and source estimation is of utmost importance, as it ensures the optimum functioning of the database.

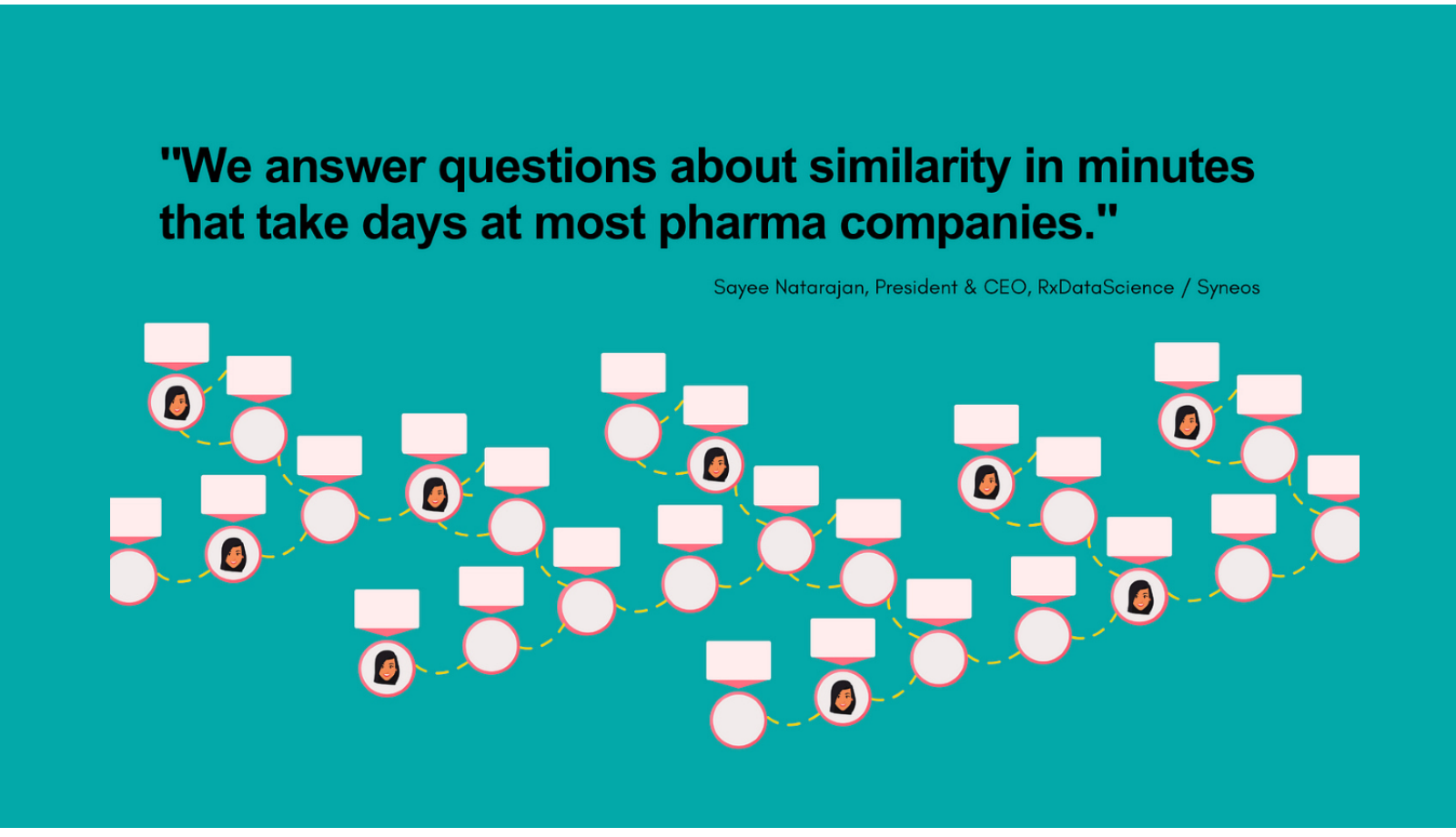
The Business Value of Vector Databases

Vectors play a key role in accelerating application performance and efficiency by modeling data by time, order, and similarity, much like the real-world. By doing so, they significantly reduce the translation overhead of queries. For instance, vector databases outperform non-vector representations by a factor of 100 in speed and 90% in efficiency – a testament to how well they match use to storage. This performance improvement is most noticeable when using similarity and anomaly detection applications powered by AI, especially in applications such as,

- Recommendation Engines
- Natural Language Understanding
- Fraud detection
- Fault Detection
- IoT-based Automation
- System Observability
- Cybersecurity
- Algorithmic Trading
- Surveillance
- Security

RxDataScience/Syneos, a company that specializes in clinical trials, uses vectors to evaluate the similarity between potential participants. By doing so, the company can ensure that it selects participants that represent the patient population, thus making clinical trials more effective.

Before using vectors, this was a significant problem in the field of clinical trials. With conventional tools, it was nearly impossible to identify subtle similarities and differences between participants. Vectors have enabled RxDataScience/Syneos to overcome this hurdle and achieve highly efficient and effective clinical trials.



Marketing claims are often scrutinized for their authenticity, and this sentiment is no different in the database industry. When it comes to performance and efficiency, it's a cutthroat battle as companies fight to reign supreme. Determining what works best is no easy feat, as industry benchmarks are scattered and varied in their findings.

Ultimately, what matters most for your data is the results it produces and how it fits your needs. Achieving this fit requires a data management system that is tailored to the data. Vector databases accomplish this through matching the in-memory representation of data with its storage, allowing for more efficient application logic. In other words, to achieve an elegant fit, proper data management is the key to unlocking the full potential of your data.

Adopting this approach can enable timely market entry, superior service, and an edge over competitors. For more insights on the Syneos use case, query patterns, and applications, visitkx.com.

Vector Databases Use Cases

Vector databases are becoming increasingly popular for their fast and efficient tools for implementing “similarity search,” but what are some of the practical enterprise use cases for this type of query?

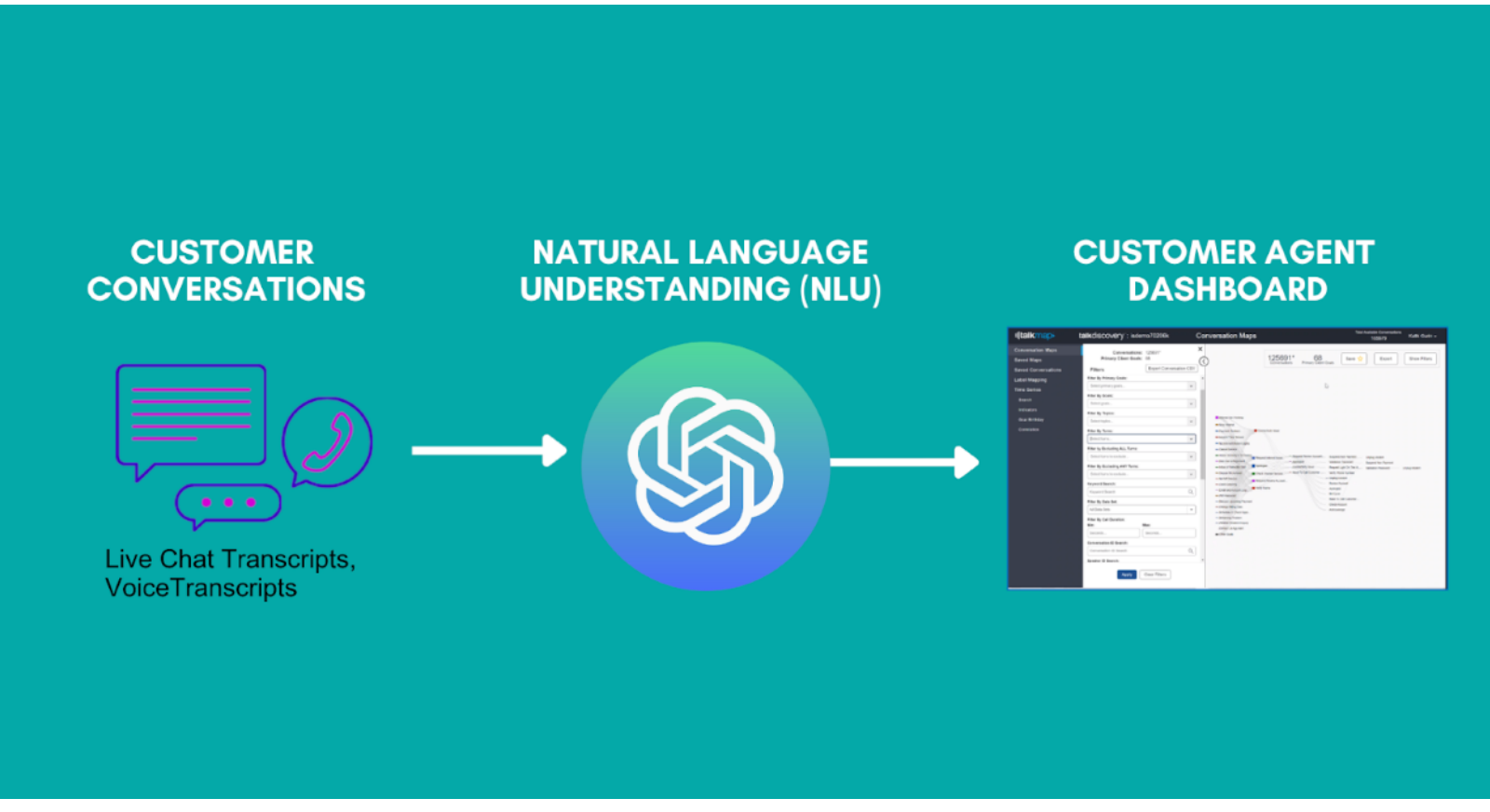
One such use case is in the retail industry, where vector databases can be used to perform product recommendations and personalized shopping experiences. Another use case is in finance, where it can be used to analyze financial data and detect patterns to make effective investment decisions. Additionally, these databases can also be used in the healthcare industry to analyze and compare genome sequences for more personalized medicine.
Vector-style data management and computing are extremely beneficial to a range of emerging applications. Here are some examples.

Recommendation Engines

Similarity forms the cornerstone of all recommendations. While Google’s similarity serves as the archetypal example, any recommendation for products, services or content can benefit from the use of vectors. For online retailers, similarity searches prove valuable in cross-selling and promoting items based on their resemblance to past purchases. The encoding of unstructured data such as images, audio, and video is facilitated by vectorization, obviating the need for manual metadata tagging and resulting in more efficient and automated data analysis.

Natural Language Understanding, or NLU

Vector databases are proving to be incredibly useful in analyzing and processing vast amounts of text data. By converting data to vector format, computers can better understand and respond to natural language inputs, such as in chatbots or virtual assistants.
One exciting example of this is in Customer Experience (CX), where the application of Generative AI is proving to be a game-changer for businesses. Innovative firms like Talkmap are leveraging natural language understanding in real-time, using local, corpus-trained collections of conversations in vector format for customer agent use. This is just one example of how vector databases are changing the game and making it easier for businesses to harness the power of AI. **Here's how GPT and LLMs, combined with vector data stores, can help understand language and turn customers into fans.**



Media Understanding and Similarity

Vector databases play a vital role in comparing images by eliminating distortions and noise from them. Encoding the images, and then comparing them to one another, makes it possible to understand them better. The areas of application of image understanding are vast and varied, ranging from medical imaging, oil exploration, security, surveillance, and public transportation automation.

For instance, similarity search with images helps monitor real-time traffic more efficiently. The application below shows video data streaming. Under the covers, AI extracts each frame using video-to-image technology and each car is evaluated in terms of its similarity to others.
Once encoded in real-time this application can perform traffic analysis, be used to keep conditions safe and secure, and alert the public or law enforcement of potential problems. By analyzing and correlating details regarding the cars, location, and other traffic data, we can make smarter choices regarding public transportation and traffic safety.

Observability & Anomaly Detection

The opposite of similarity search is anomaly detection, which involves identifying objects that are distant or different from an expected result. In other words, it's all about spotting those outliers in a set of data. Anomaly detection and observability play a crucial role in various applications, particularly in those that assess threat, fraudulent activities and temporal trends in time.

Applications such as those in finance, healthcare and security heavily rely on accurate anomaly detection to flag fraudulent transactions, detect irregular patterns in vital signs, or identify and mitigate potential security threats lurking in large datasets. As a result, improving anomaly detection methods has become a critical mission and is constantly being enhanced with advanced AI techniques, like,

- System management
- Fraud detection
- Risk management
- Spam filtering
- Identity theft
- Intelligence
- Weather forecasting
- Sentiment analysis

These use cases assess and match patterns of access, with vector database technology facilitating faster and more efficient processing.

Semantic Search

Vector databases are designed to store and index vector embeddings derived from natural language processing (NLP) models. These embeddings assist in comprehending the meaning and context of text strings, sentences, and entire documents, culminating in more precise, suitable, and accurate search outcomes.

Through natural language queries, users can retrieve what they need more efficiently without requiring insight into the data classification specifics. This creates a better search experience with more intuitive and productive results.

Data Quality, Deduplication and Record Matching

Vector similarity search has proved useful for record matching and deduplication, especially in the realm of data quality, integration and analytics. By identifying semantic differences, errors, and similarity; similarity algorithms assist in finding duplicate data, enhancing data catalogs and pointing analysts towards new data sources.
One of the many uses of this technology is data processing, where it aids in making enterprise data catalogs more useable and relevance by entirely removing cloned content. Additionally, by suggesting new sources of data to analysts, it enhances the accuracy and quality of future analytics.

Required Capabilities of Vector Databases

Vector databases are specialized in storing, compressing, indexing, and retrieving vectors more efficiently than databases that are designed for relational, unstructured, graph, or STAR schema data stores. The optimization of a vector database depends on the scope of its implementation since each is customized for datasets that cater to various use-cases.

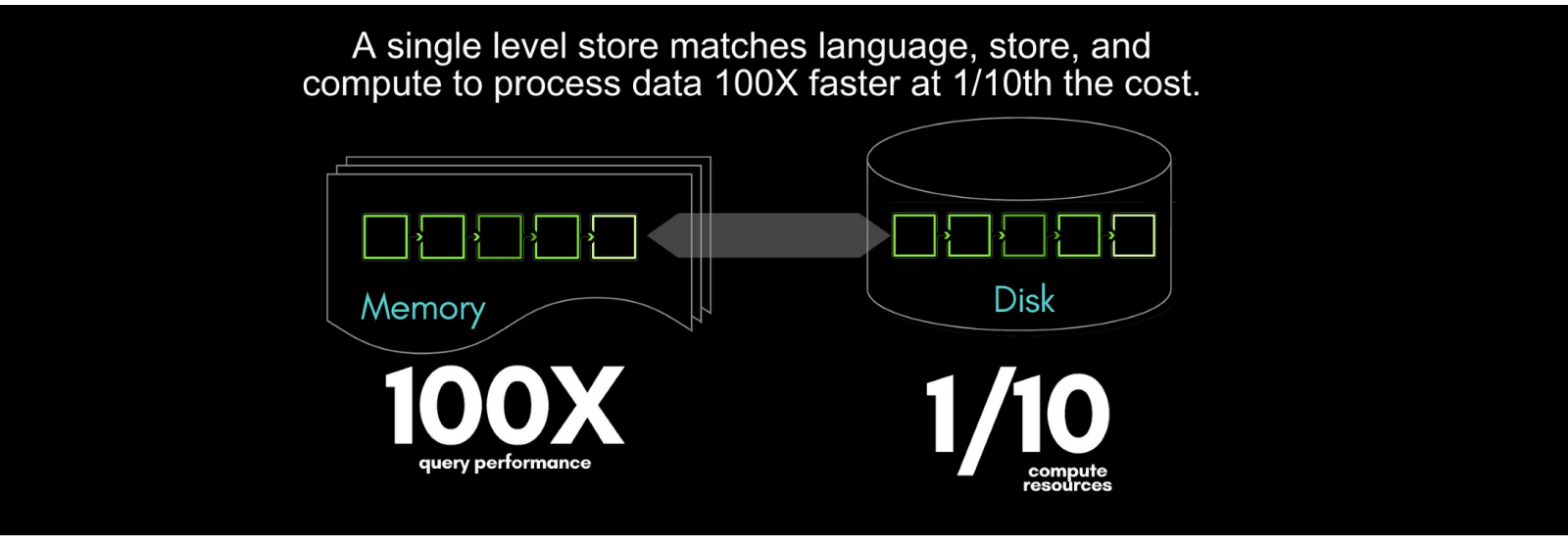
Here are the seven crucial elements that need to be included in vector databases:

#1: SINGLE LEVEL STORE

To be truly game-changing, technology must be leaps and bounds ahead of its predecessors in both speed and efficiency, by at least fifty times. This is where Vector databases come in, synchronizing data storage with the in-memory representation that's used to process it. This not only delivers blazing-fast performance but also opens up new possibilities for data-driven innovation.

"To be truly disruptive, a database must be 50 times faster than its predecessor."
~ Michael Stonebraker, 2014ACM Turing Award

Vector databases adopt the principle of a single level store, which implies that the on-disk representation of data is mirrored as closely as possible in vector format. This approach enables efficient querying and fast processing of large datasets while minimizing the need for data movement.



By design, these "physics" allow a vector database to function faster and more efficiently than non-vector stores. In addition, the data store retains metadata and metrics that gauge the relevance and similarity among vector embeddings. Certain metrics, like Euclidean distance, cosine similarity, and dot products outperform others.

#2: VECTOR INDEXING

Vector databases use structures like nearest neighbor indexes to assess how similar objects are to each other. Traditional nearest neighbor search has a problem for large indexes as they require a comparison between the search query and the entire indexed vector, which takes time.

To address this problem, vector databases implementApproximate Nearest Neighbor (ANN) algorithms, which offer very precise balances while also providing very fast performance. Popular methods for building ANN indexes include techniques such as HNSW, IVF, and PQ. Each technique improves performance in different ways, such as reducing memory resources or improving accuracy.

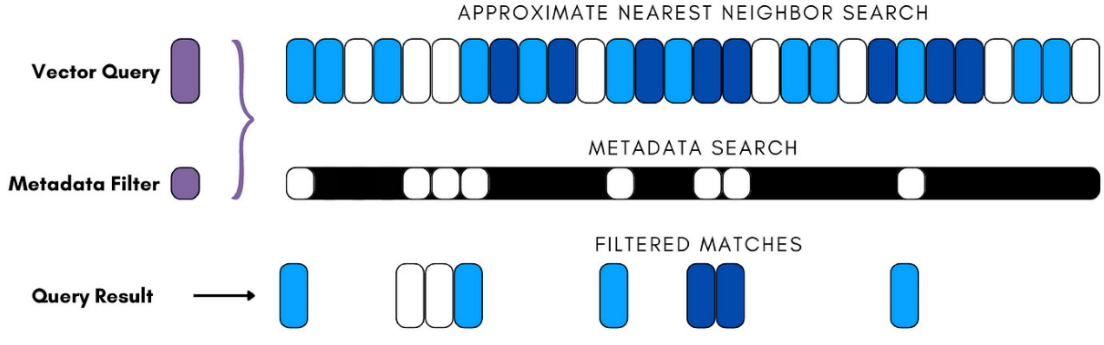
It's common for vector database implementations to mix these algorithms to achieve an optimal mix of performance and accuracy. Without vector indexing algorithms, managing vector data often becomes difficult to scale. Stand-alone frameworks like Faiss require experienced teams of engineers with a good grasp of database theory and implementation. Therefore, most companies prefer to buy a packed database instead of building their own.

#3: FILTERING

Filtering is a powerful search feature that leverages metadata to deliver a streamlined subset of results that precisely match your criteria. This not only boosts result relevance, but also reduces unnecessary query processing, making for a faster, more efficient search experience.

Furthermore, metadata filters like those used in the approximate nearest neighbor algorithm can even be applied to gain highly accurate recommendations. For instance, a vector database query may first seek out movies similar to those that Chad likes, then apply filters such as genre, director, or year to fine-tune the results.

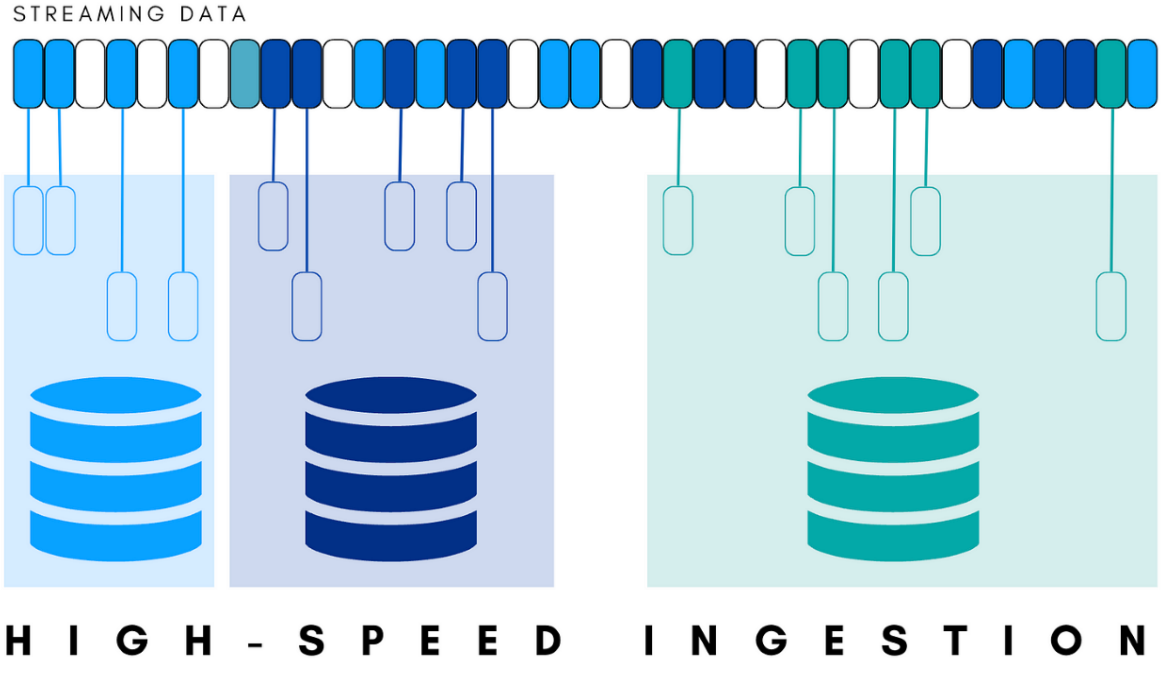
In short, whether you're optimizing search results or refining recommendations, using metadata filters is an essential strategy for both efficacy and efficiency.



#4: HIGH-SPEED INGESTION OF STREAMING DATA

IoT-connected devices transmit data that's best organized by vector, making it crucial to collect, filter, collate, and store as time-series data. The catch? Connected devices emit copious amounts of data: thousands, millions, billions, or trillions of updates daily!

To make sense of this information, it's essential to filter, aggregate, and store on the fly; applications rely on viewing streaming data in real-time with the highest fidelity, ensuring that analyses lead to usable insights.



Vector databases are an ideal format to store streaming data as vectors inherently hold temporal order. This makes them perfect for answering typical questions that enterprises might ask, like:

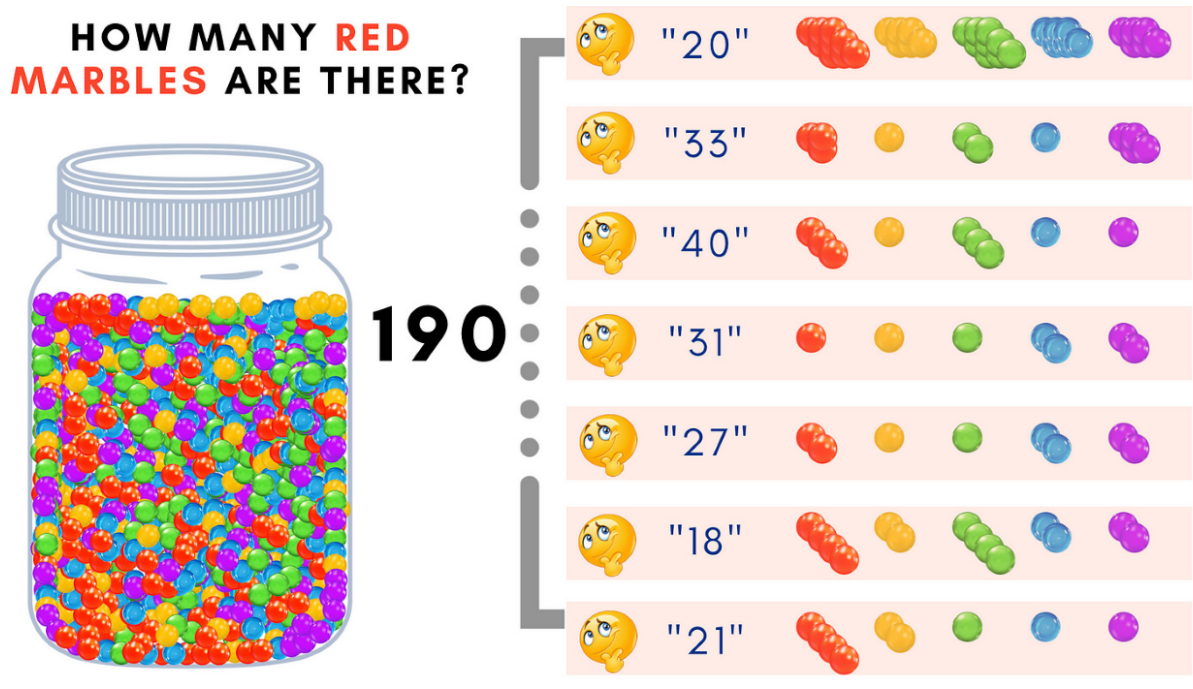
- At what point did the temperature reading reach hazardous levels and continue to exceed them for more than five minutes?
- Display the most recent five attempts to access the potentially compromised account.
- Retrieve the 5 most recent instances of sudden volatility changes for this stock.

To achieve high-speed ingest, streaming updates need to be micro-batched in-memory and stored efficiently on disk. This requires balancing tradeoffs between volume, latency, and algorithmic filtering.
Some vector databases provide control over these tradeoffs, enabling developers and administrators to tailor storage to their application's specific requirements.

#5: SHARDING AND GPU SUPPORT

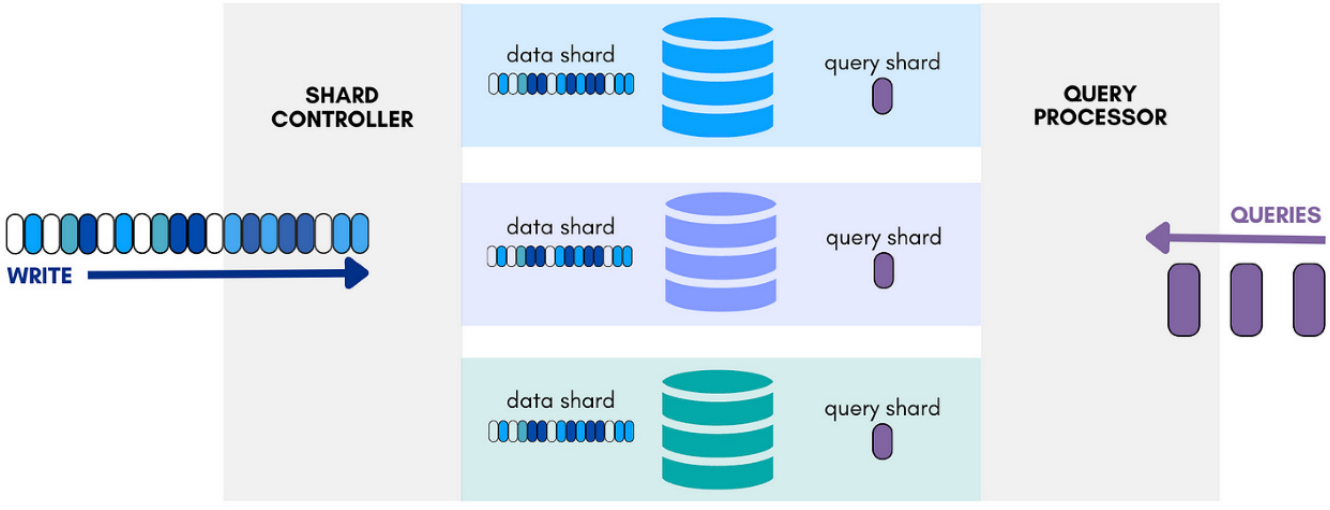
Imagine you've got a jar with 1,000 marbles from a friend, and they offer a million dollars to the first among 100 people to count the number of red marbles in the jar. What do you do?

The best strategy to win the prize is to pool your resources and work together. In this analogy, you're the vector database, the marbles (arranged by color) are vectors, and the jar represents your database. The act of dividing tasks is called "sharding."



As the scale of database problems grows, shard query processing comes into play. When the number of queries or the size of the dataset surpasses a single machine's capabilities, data gets sharded. In vector databases, vectors are also divided into shards. Every shard is combed through to identify the best match, which is then seamlessly combined with the other shards' results.

Vector databases frequently utilize GPUs or cloud-native tools, such as Kubernetes, to partition individual work units and assign dedicated resources to each one. These shards then operate concurrently to scan vectors and integrate their findings into a consolidated end result.



Sharding can help large queries run quickly across vast amounts of data. For example, if you have 20 million vectors to search through, you can use 20 shards to get results in the time it takes to search 1 million vectors with only one shard.

By using 40 shards (with 500,000 vectors per shard), you can achieve even faster results.

Simply put, using fewer vectors per shard reduces query latency and enables you to search billions of vectors in a reasonable timeframe. While there is more to it, sharding makes large-scale search tasks more efficient and manageable.

#6: Replication

Suppose you just won a million dollars from your friend for counting red marbles. Now, she's challenging you to answer five more questions correctly about other colors, or lose it all on double or nothing. Would you take the bet?

The likelihood is higher if all ten of your friends are still in the picture.
Imagine recruiting 40 new team members, dividing them into groups of 10, and providing each of them a jar for context. You're likely anticipating the sweet aroma of success – mirrored ability to answer questions. With the influx of inquiries, you'd now have redundant yet replicated capability at your fingertips.

Vector databases work in a similar manner, creating redundant "workers" to handle parallel requests. In the event of machine failures, replicas enhance availability by allowing redundant resources to take over in multiple availability zones guaranteed by cloud platforms, ensuring prompt resource allocation and high availability.

#7: Language Access, APIs and Vector Primitives

Vector queries can be posed via various programming languages – from widely used languages such as SQL and Python to more sophisticated ones like Tensorflow – or through intuitive, visual drag and drop interfaces found in business intelligence tools like Power BI and Tableau.

Database query processors transform user requests into data sets for display or programming consumption. APIs simplify vector database management for developers. Some vector databases provide native data frame language bindings for Python, Java, Go, REST APIs, and SQL support to allow any application, program or business intelligence tool access to vector data.

SUMMARY

Vector databases are the latest innovation in database technology, used for similarity search, anomaly detection, and temporal data. They store vector data, which expresses relationships between multimedia data and represents data as numerical values. Traditional stores are not designed to handle vector data. Vector databases process workloads 100 times faster and at a lower cost than conventional stores.

This article introduced vector databases, including their functionality, business drivers, use cases, and core attributes for success. To learn more, get started with KDB.AI today and take the first step towards better data storage!

KDB.AI Powered by KX

www.kdb.ai