YDB

YDB: How To Implement Streaming RAG In A Distributed Database

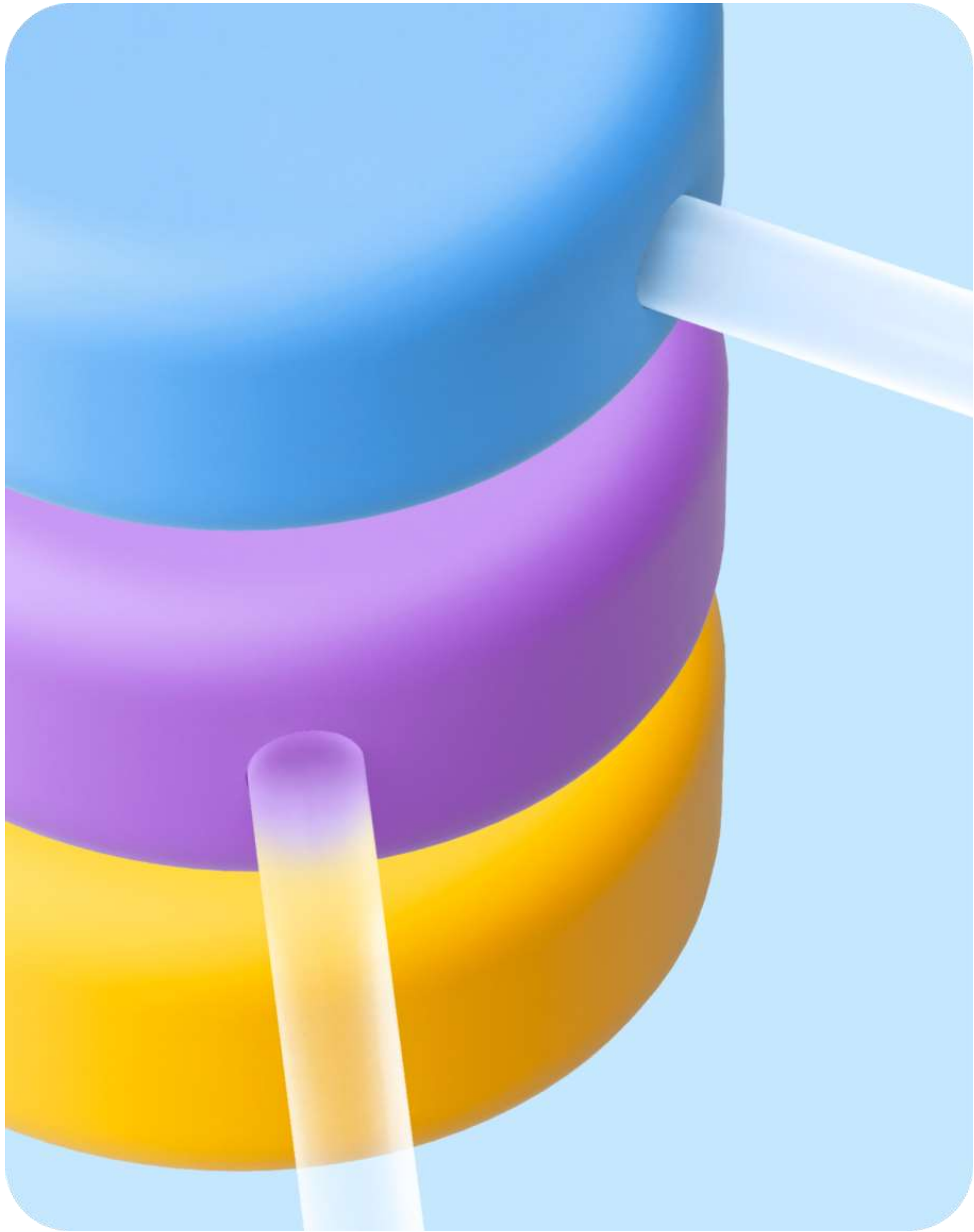
Alexander Zevaykin,
PhD, YDB Team Leader

Elena Kalinina,
YDB Technical Manager

ydb.tech

Streaming RAG

Retrieval Augmented Generation



Enterprises Are Not Developing Their Own Generative AI Models

Too complex and too costly

Millions of dollars

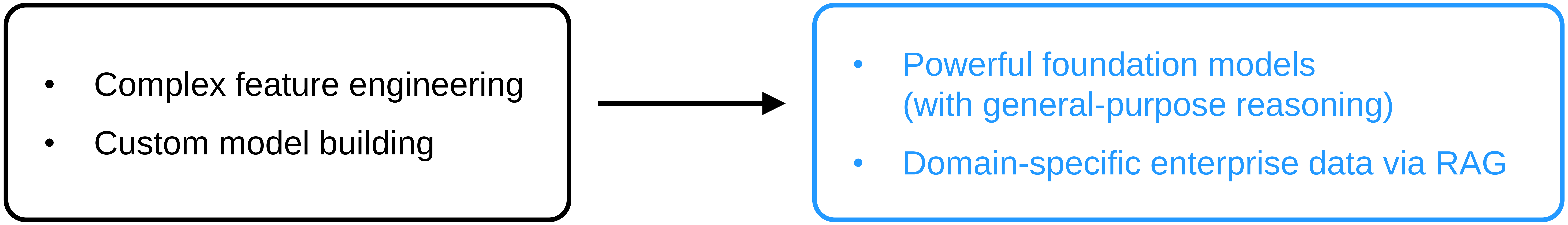
A lot of data acquisition

Compute infrastructure

Scarce GPU

Expertise

Data privacy



- Complex feature engineering
- Custom model building

- Powerful foundation models (with general-purpose reasoning)
- Domain-specific enterprise data via RAG

Retrieval Augmented Generation

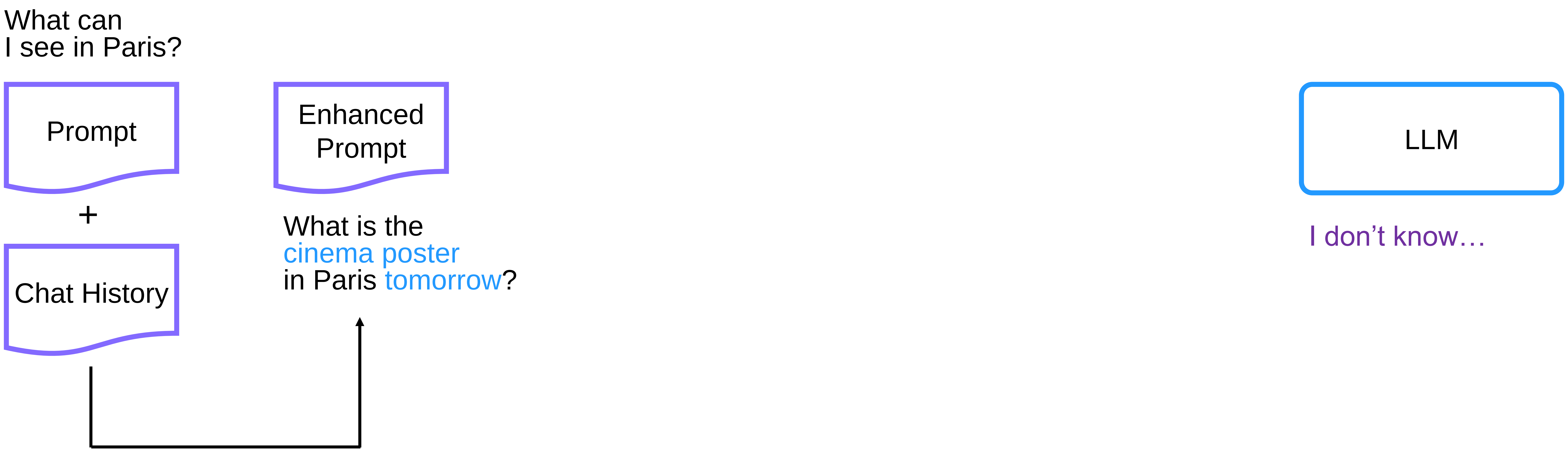
What can
I see in Paris?

Prompt

LLM

I don’t know…

Retrieval Augmented Generation

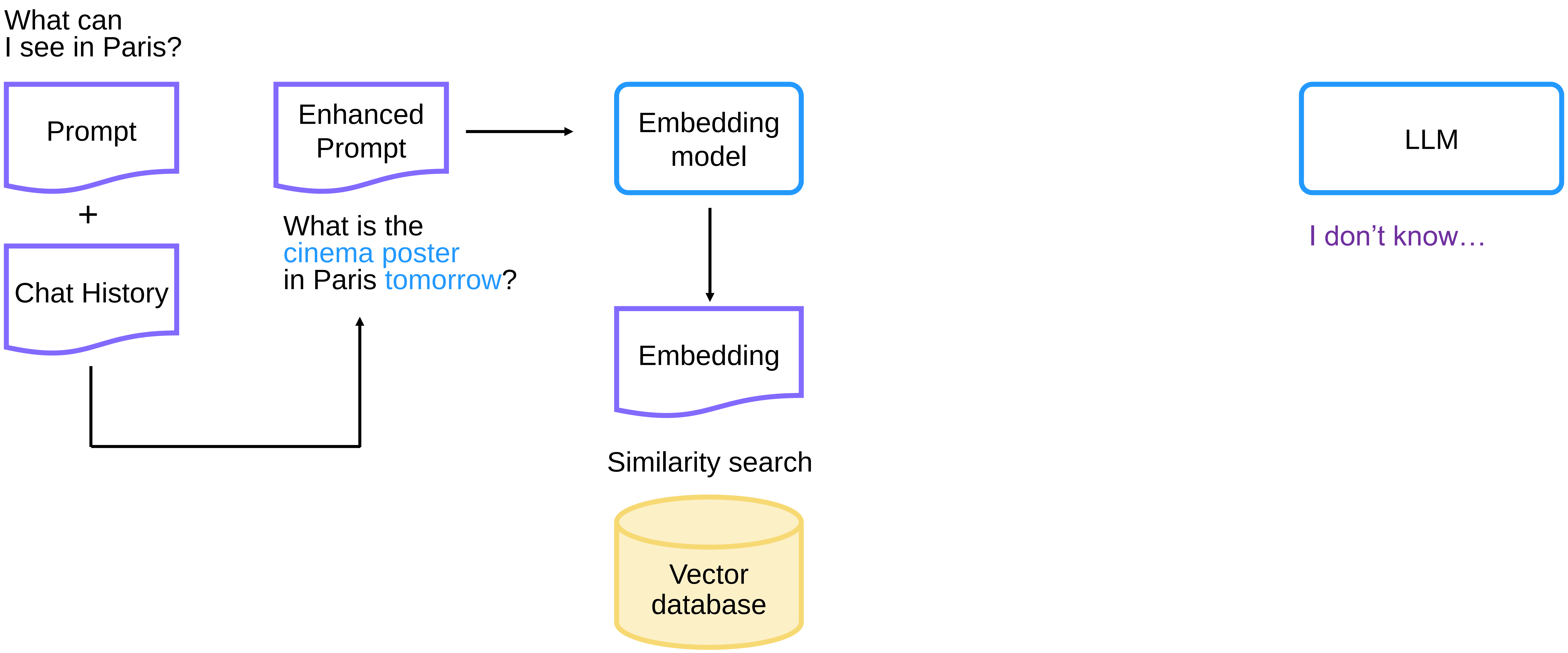


What can
I see in Paris?

What is the
cinema poster
in Paris tomorrow?

I don’t know…

Retrieval Augmented Generation



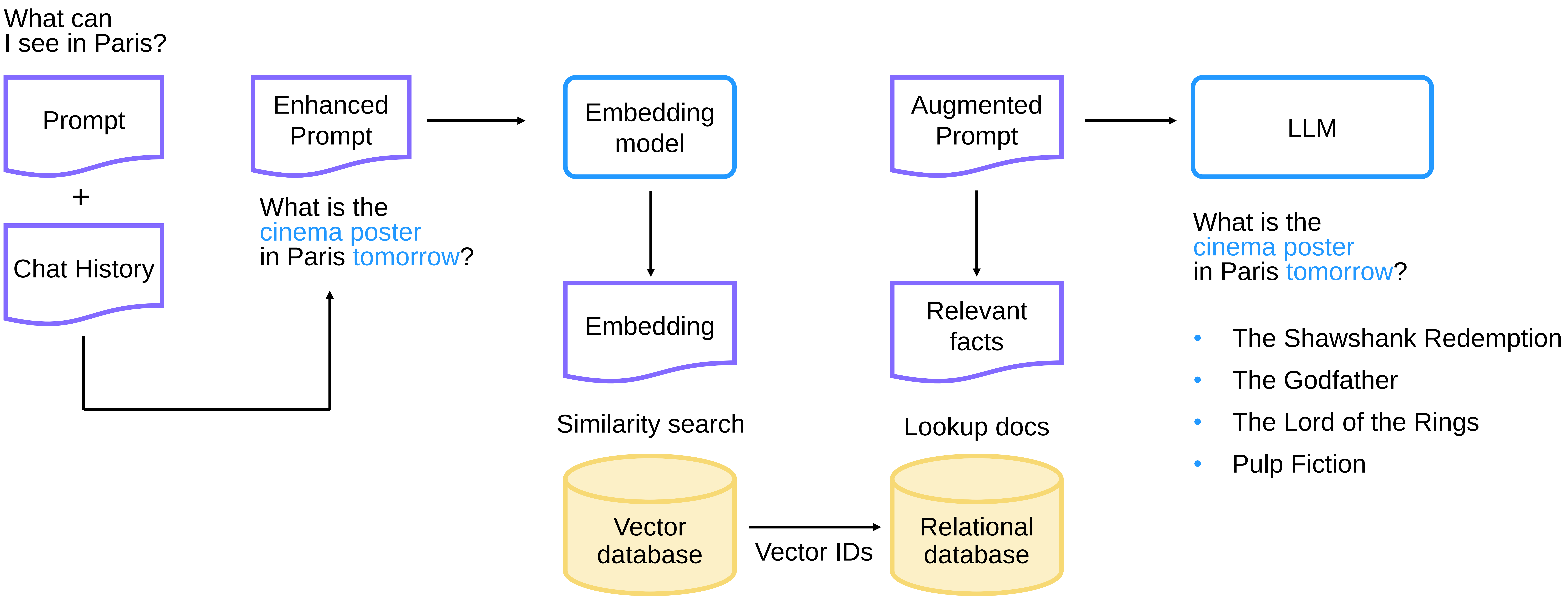
What can
I see in Paris?

What is the
cinema poster
in Paris tomorrow?

Similarity search

I don’t know…

Retrieval Augmented Generation



Trend #2: Moving Toward Streaming RAG

AI without up-to-date data is frustrating and its value is limited

Generative AI in the enterprise is more about streaming, not batch

2018 GPT-1

2019 GPT-2

2020 GPT-3

2022 GPT-3.5

2023 GPT-4

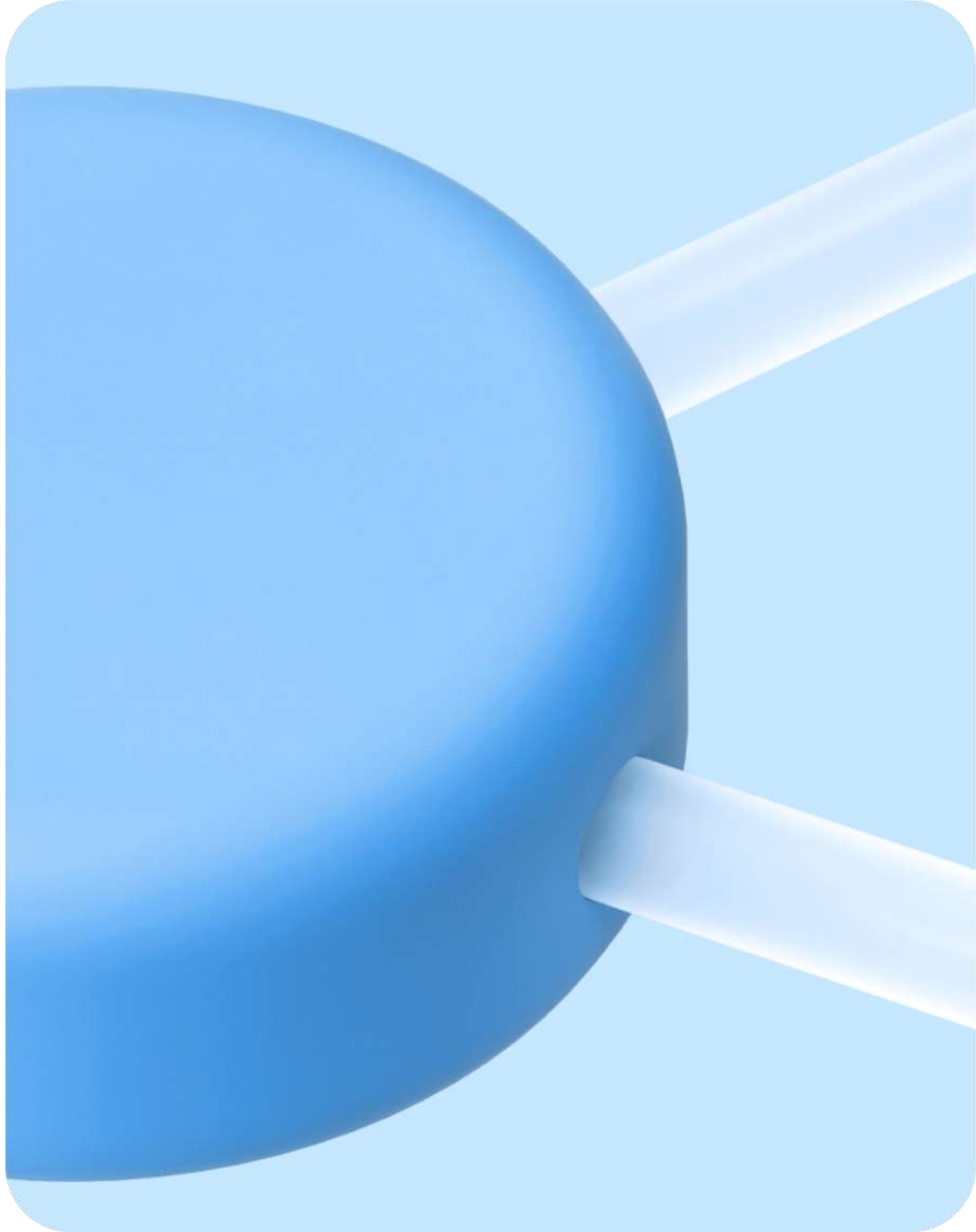
Use Cases Of Streaming RAG

Real-Time Financial Advisory Platform

Dynamic Healthcare Monitoring and Assistance

Live News Analysis

YDB



What Is YDB?

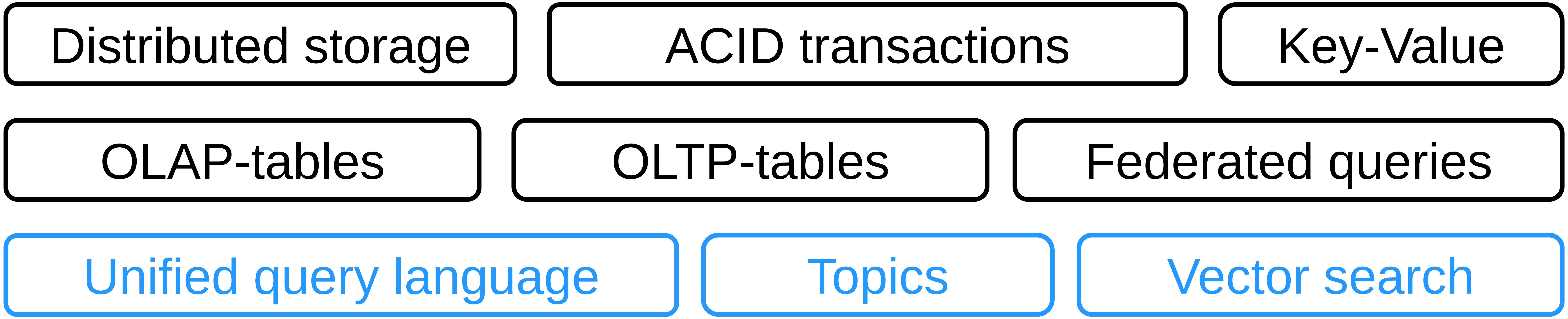
Distributed SQL RDMS for operational, analytical and streaming workloads

github.com/ydb-platform/ydb

ydb.tech

- Horizontal scaling
- ACID transactions in multiple AZ
- Operability and automatic recovery in case of failures
- Scaling by millions of transactions per second and **petabytes** of data
- Production installations of **tens of thousands** of servers
- Open-Source under Apache 2.0 license

YDB As A Platform



Distributed storage

ACID transactions

Key-Value

OLAP-tables

OLTP-tables

Federated queries

Unified query language

Topics

Vector search

YDB Intrinsic Advantages

Scale

Sharding

Replication

Spiky workload

Cross Datacenter

Production readiness

Fault-tolerance

Rolling update

Persistence

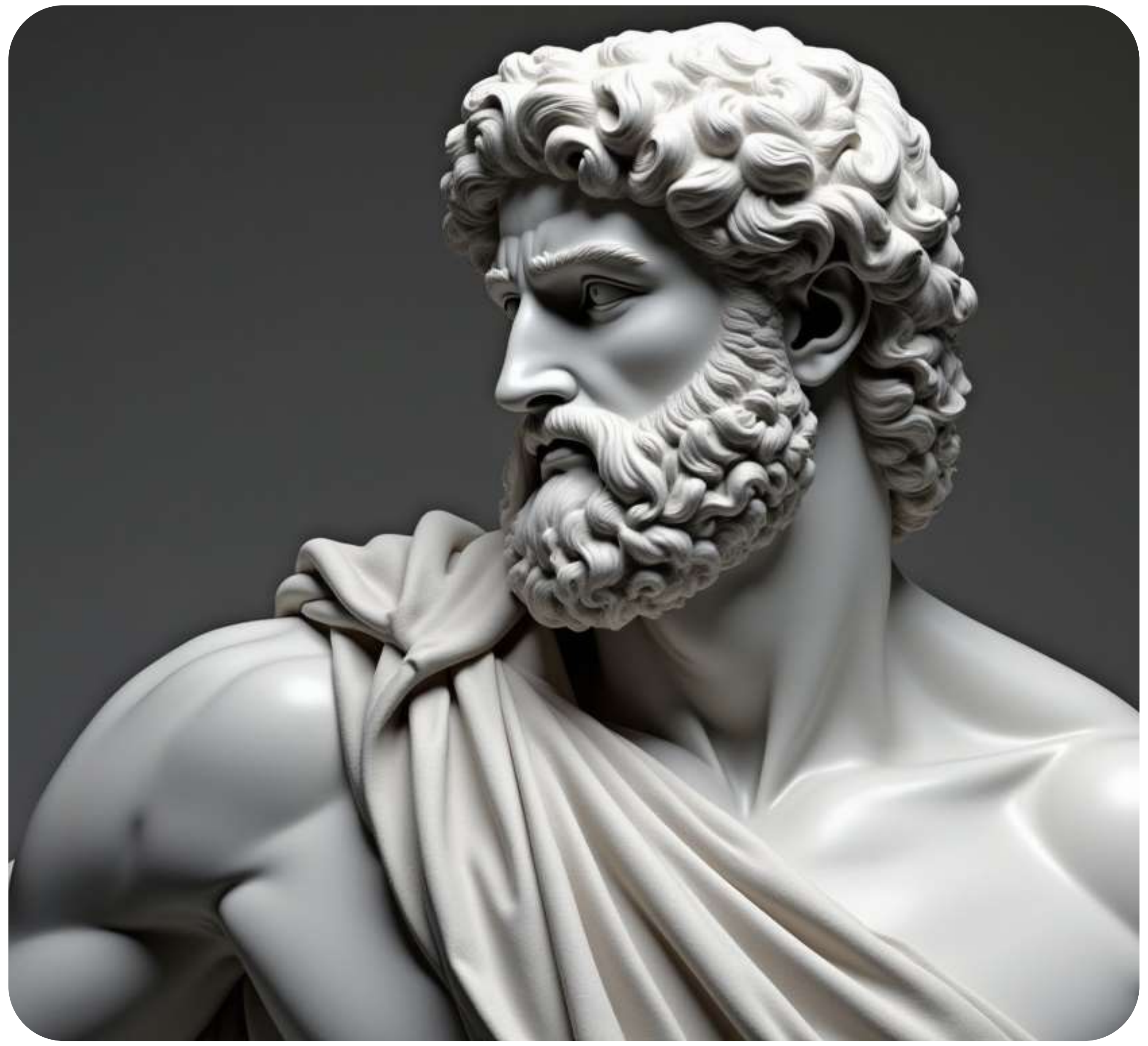
Consistency

Alerting

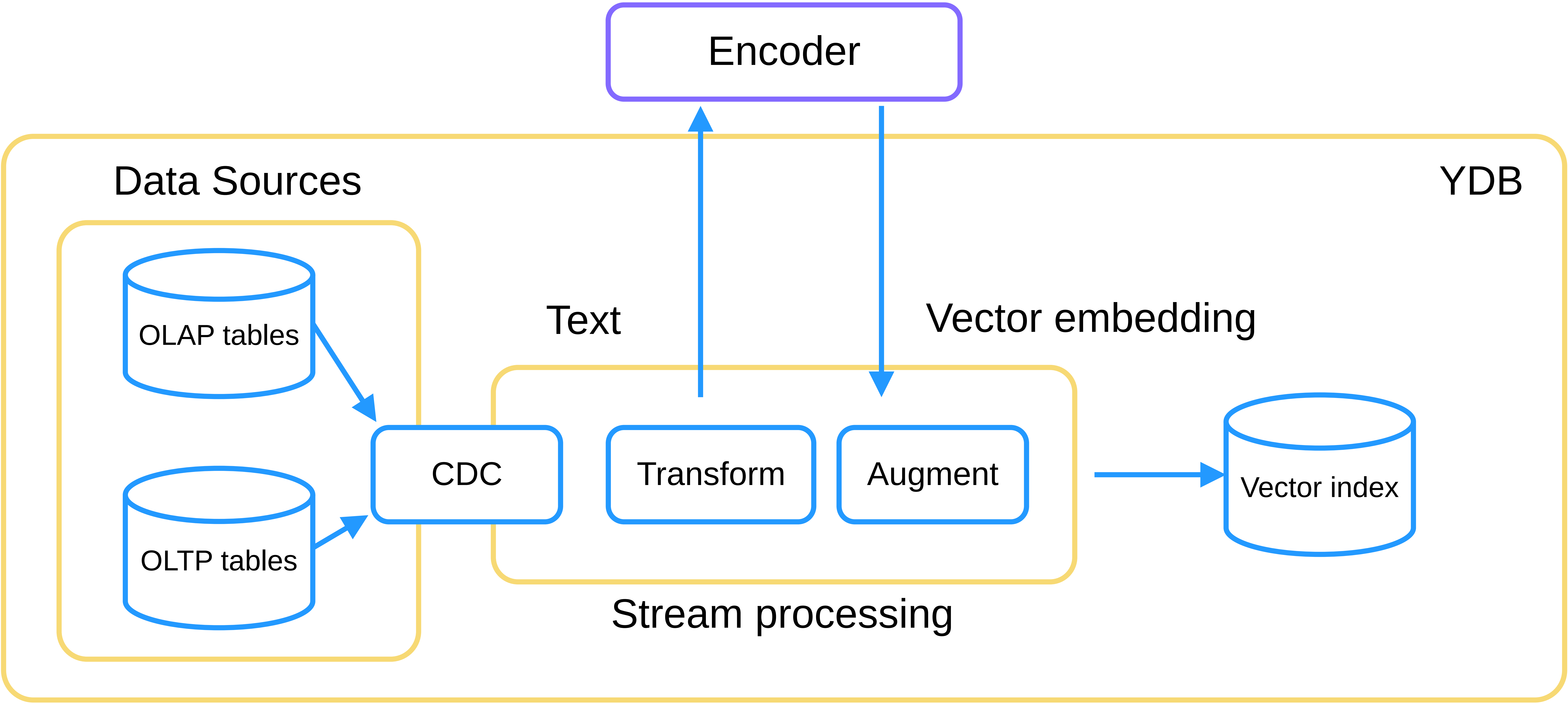
Support

Monitoring

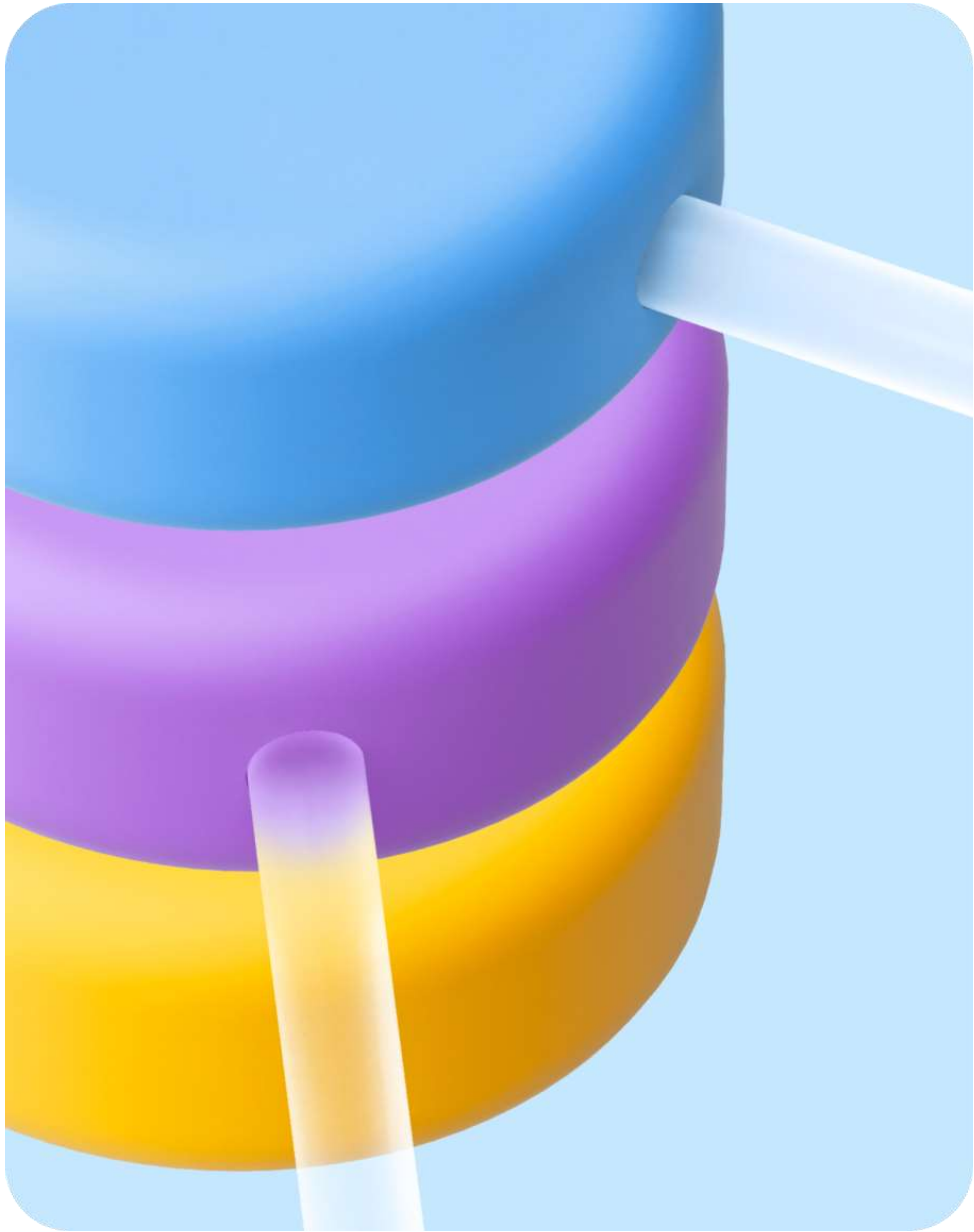
Herculean tasks



YDB: Real-time Streaming RAG



Streaming Processing



Elena Kalinina,

Technical Project Manager, YDB

YDB Topics — What’s This?

YDB Topics is an implementation of persistent queues within YDB

Main features

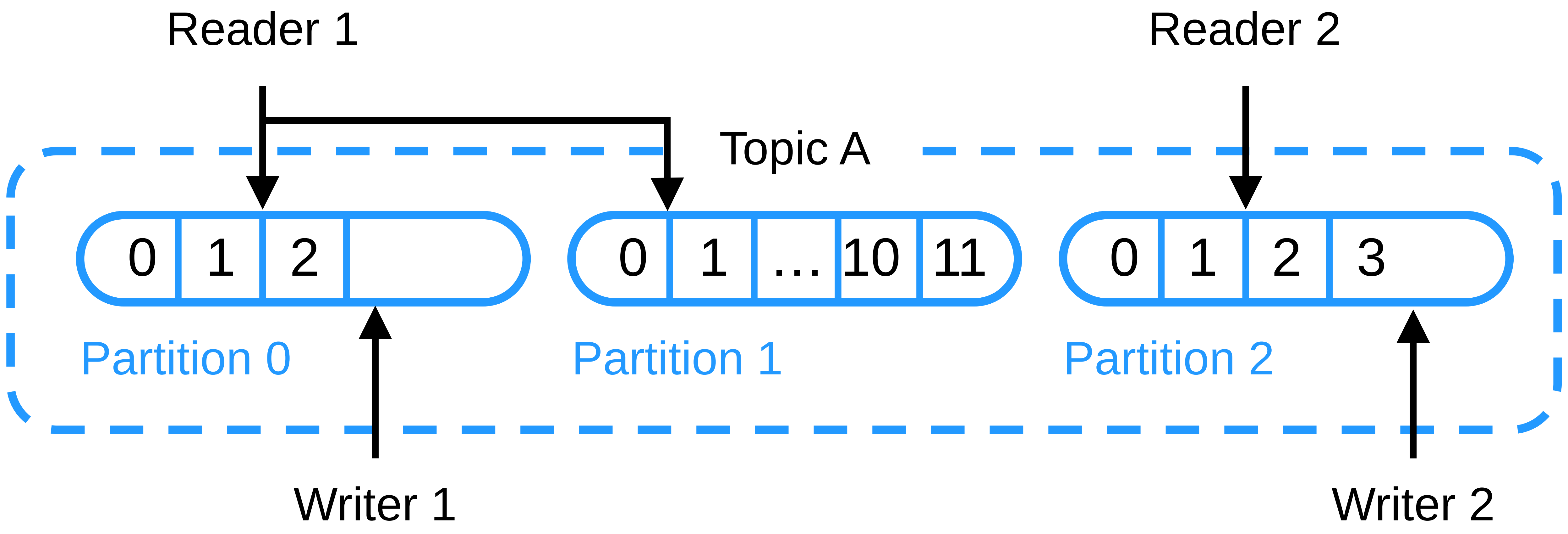
- Reliability
- High throughput

Based on YDB platform

- Change Data Capture (CDC)
- Transactions with topics and tables

API

- YDB Topic API
  C++ SDK, Java SDK, Python SDK, Go SDK
  All YDB Topics features are supported
- Apache Kafka API



Change Data Capture

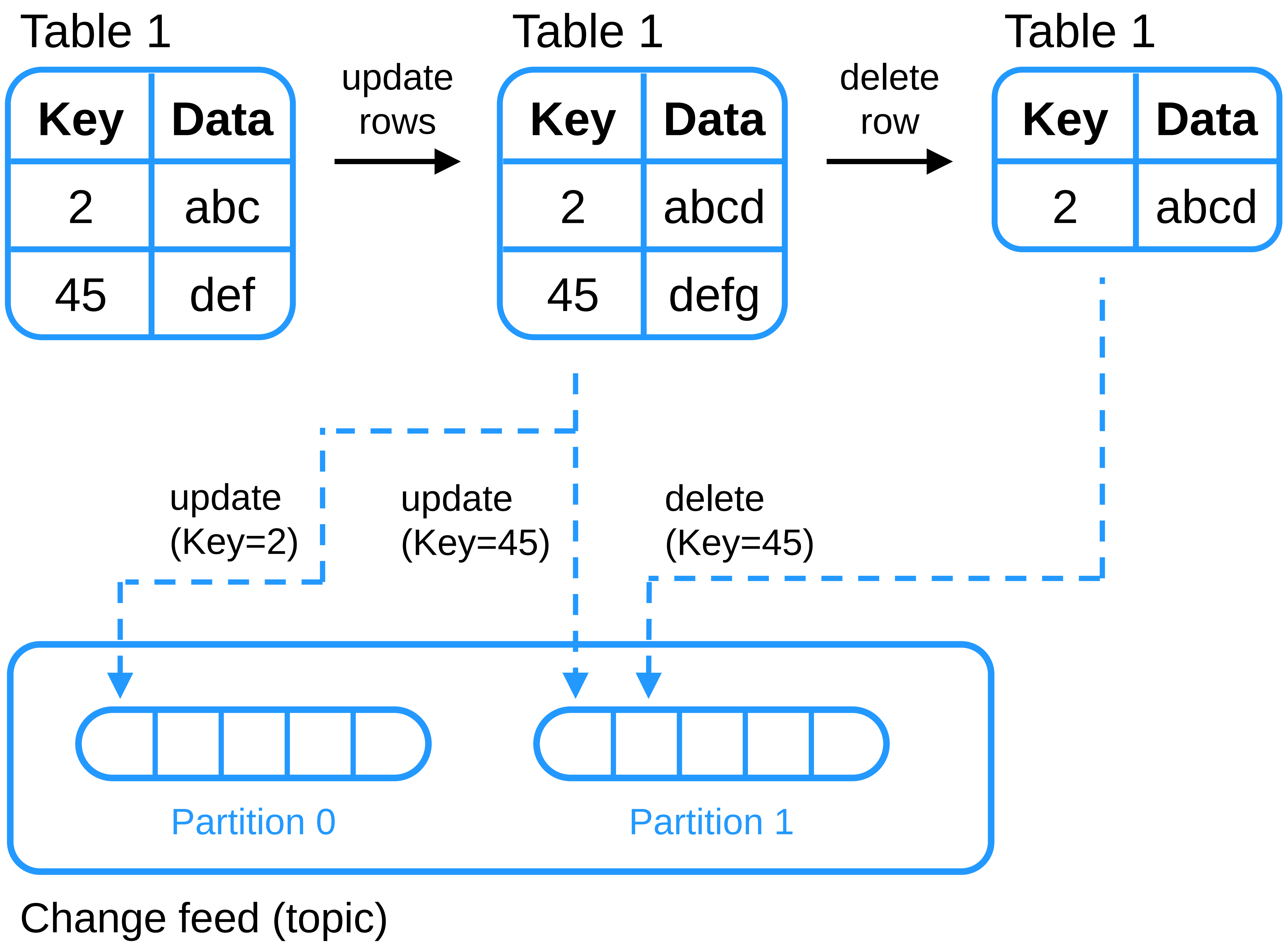


Table 1

Key	Data
2	abc
45	def

Table 1

Key	Data
2	abcd
45	defg

Table 1

Key	Data
2	abcd

- Changefeeds

  for capture any table changes

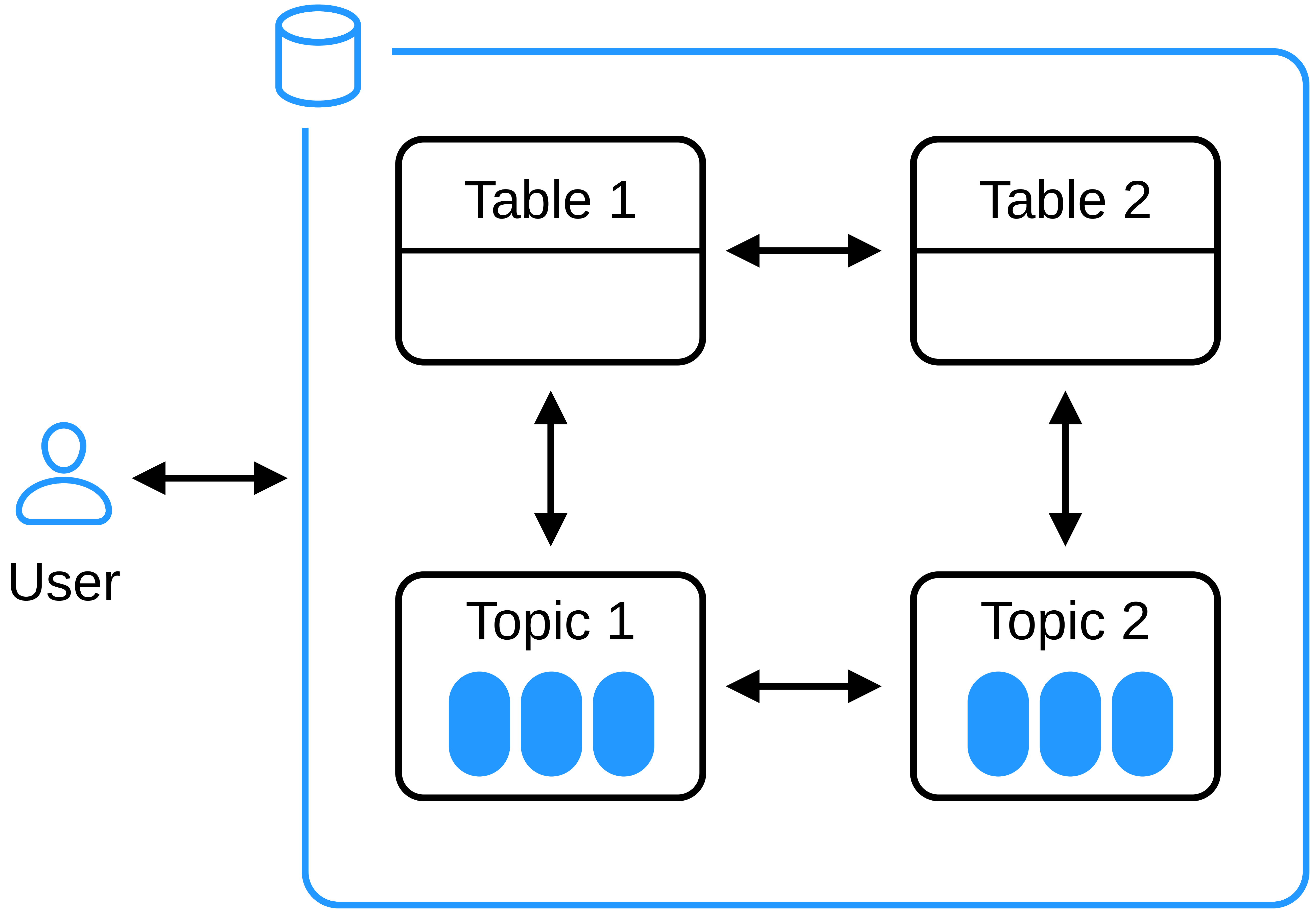
- Exactly once delivery
- Change records are sharded
- Order of changes

Transfer Data From Topic To Table

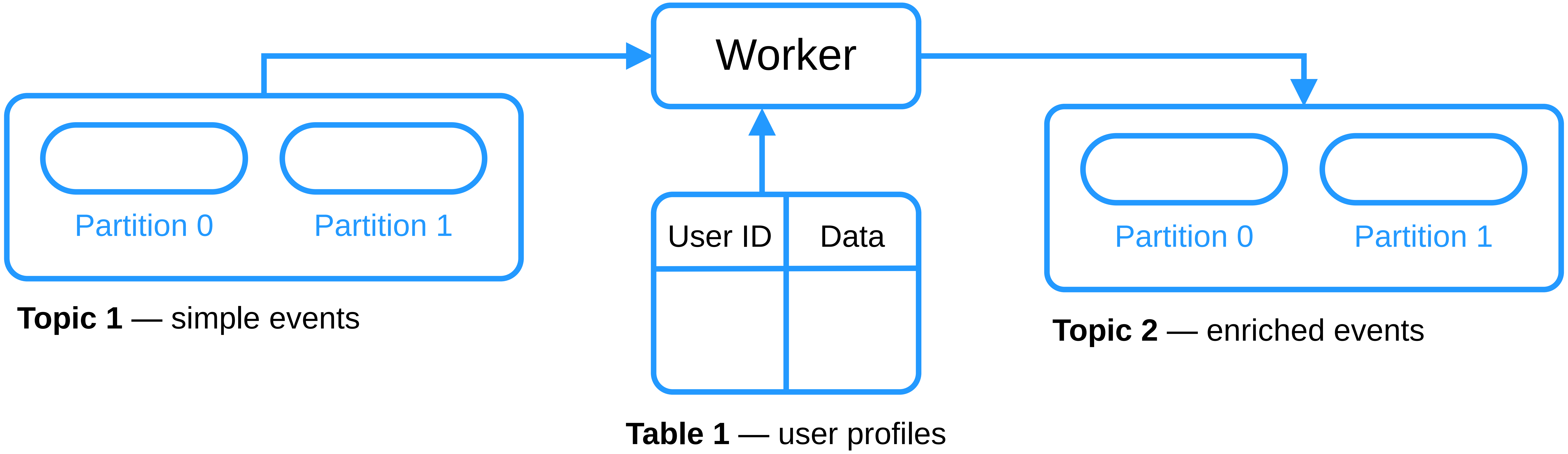
```
CREATE TABLE TargetTable (<Some Columns>);

CREATE TOPIC SourceTopic;

$transform = {
      < Some Complex Transformation Logic >
};

CREATE TRANSFER ExampleTransfer
FROM SourceTopic TO TargetTable USING $transform;
```

Transactions With Tables And Topics

- ACID transactions involving tables and topics
- Within one database



Distributed Transaction Example: Enrich Events


Worker
Partition 0
Partition 1
Topic 1 — simple events
User ID
Data
Table 1 — user profiles
Partition 0
Partition 1
Topic 2 — enriched events

```
BEGIN TRANSACTION Tx1;
A = READ 1 EVENT FROM Topic1;
B = READ Data FROM Table1 WHERE UserID = GetUserID(A);
C = EnrichEvent (A,B);
WRITE INTO Topic2: EVENT C;
COMMIT Tx1;
```

Transactions With Tables And Topics

We add ACID guarantees to topic-table operations

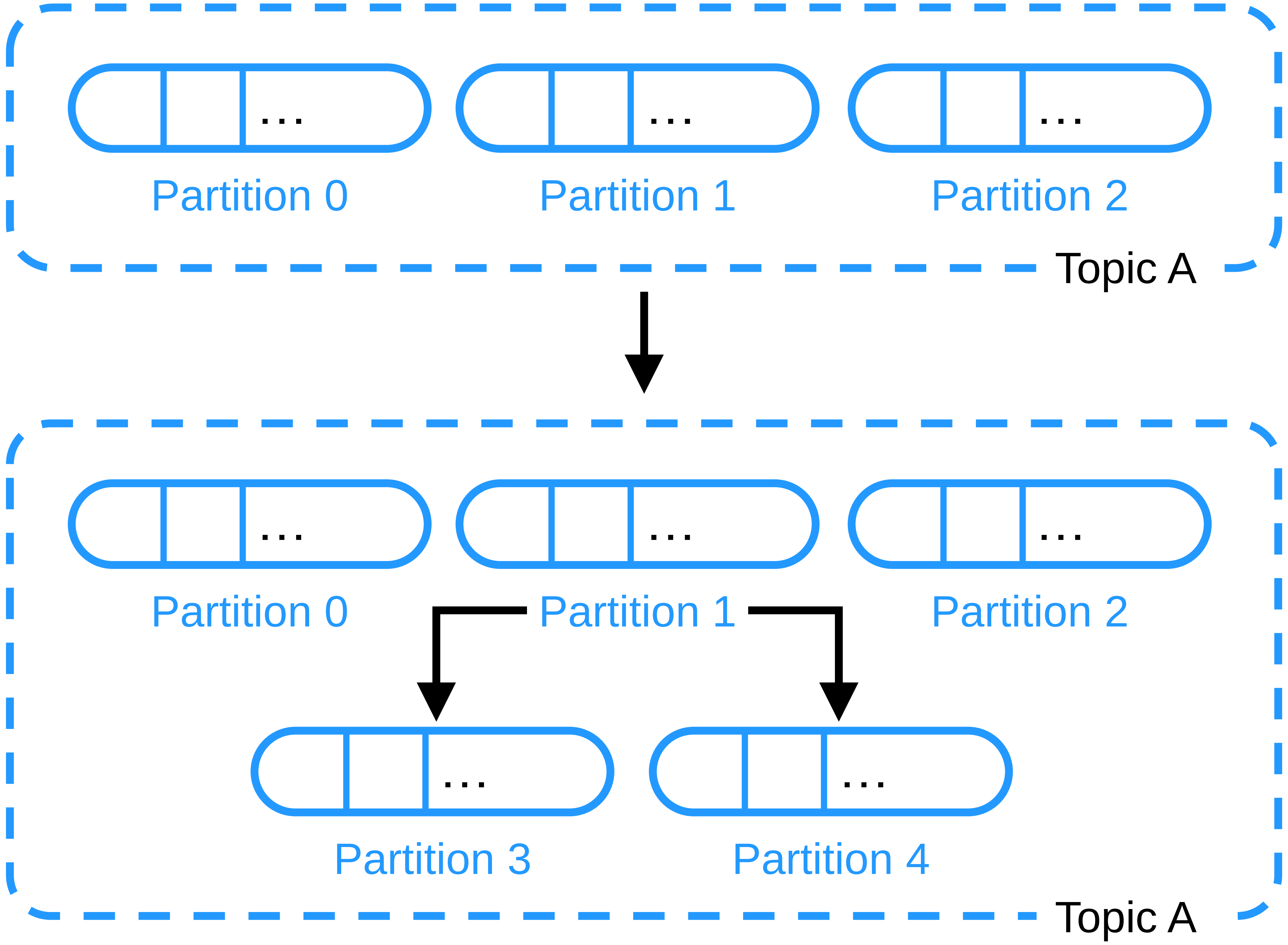
It simplifies user code

CPU usage and system throughput are the same

Minimal impact on latency

Topics Autopartitioning

- Topic is divided into the partitions for scalability
- Partitions count can be increased automatically
- Guarantees:
  - Exactly once for writing
  - Reading order

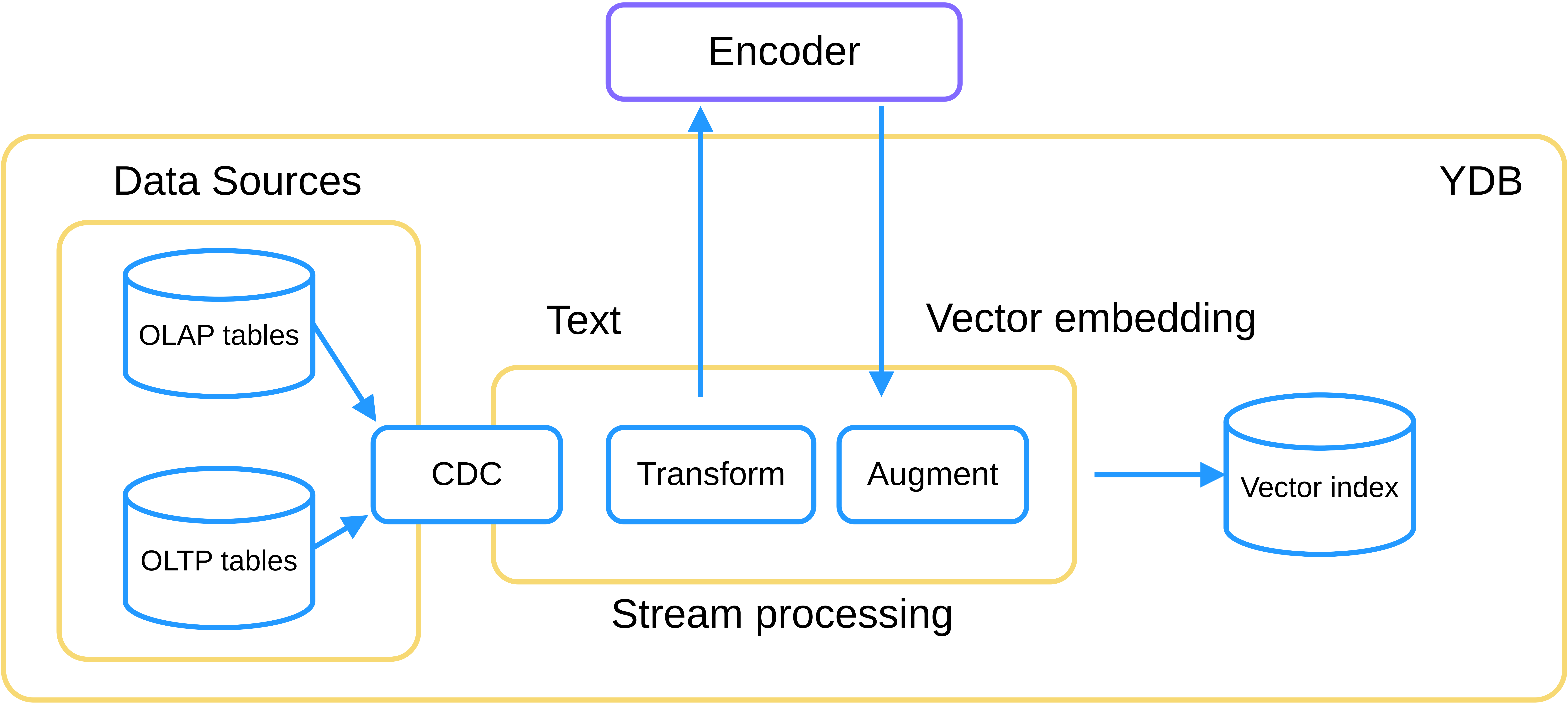


Streaming For RAG in YDB

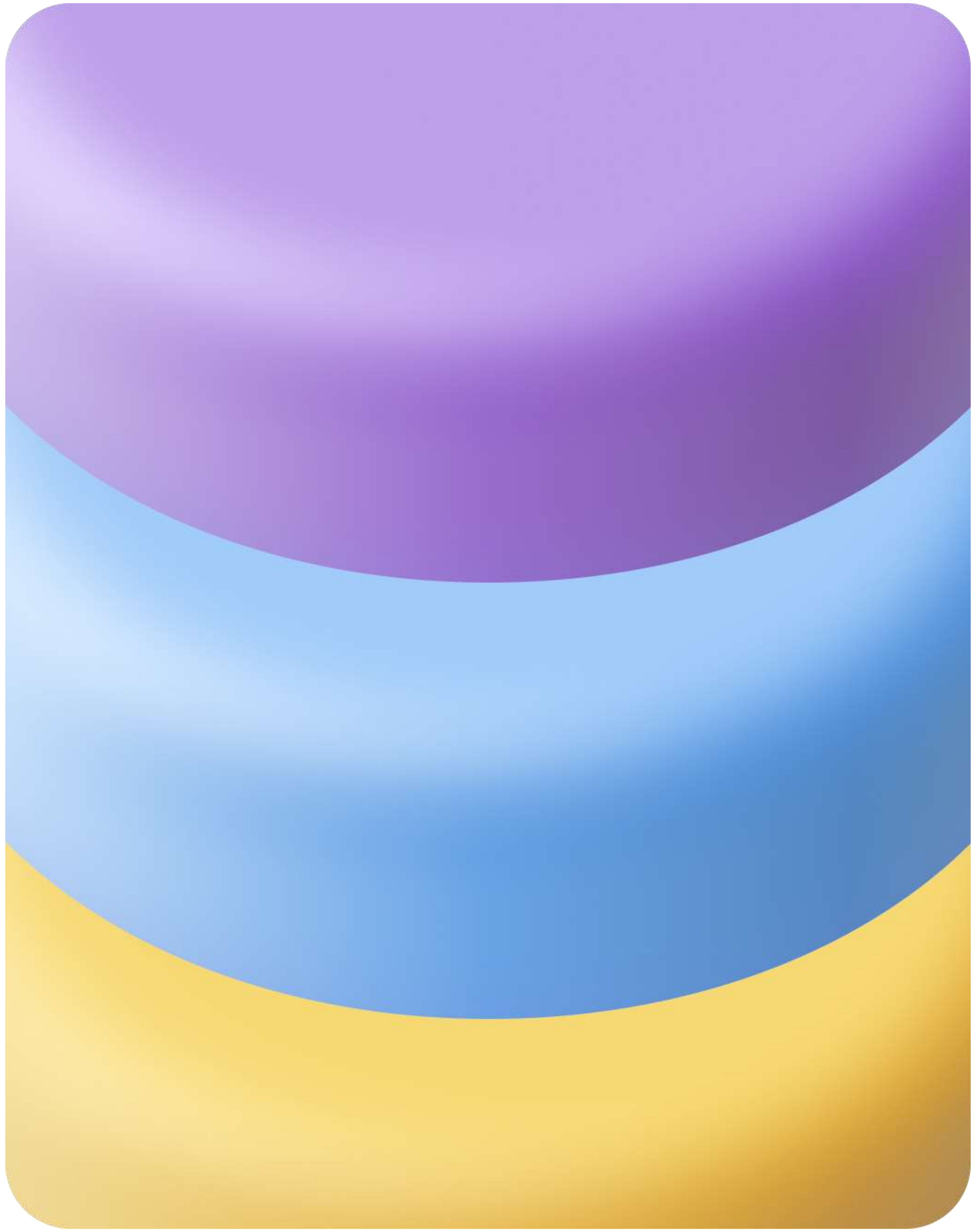
- Deliver table changes with changefeeds
- Transfer data from topic to table
- Any topic-table data transformations within classic ACID transactions
- Topics autopartitioning

- High throughput
- Reliable
- Kafka API compatible

YDB: Real-time Streaming RAG



Vector Index



Vector Index Requirements

- The index is global
- The index is synchronous and consistent

- Table size = billions
- Search latency = tens of ms

- Creation time = O (table size)
- Occupied space = O (table size)

= scale in a linear way

Why Don't Existing Algorithms Suit Us?

Distributed system

Automatic scaling

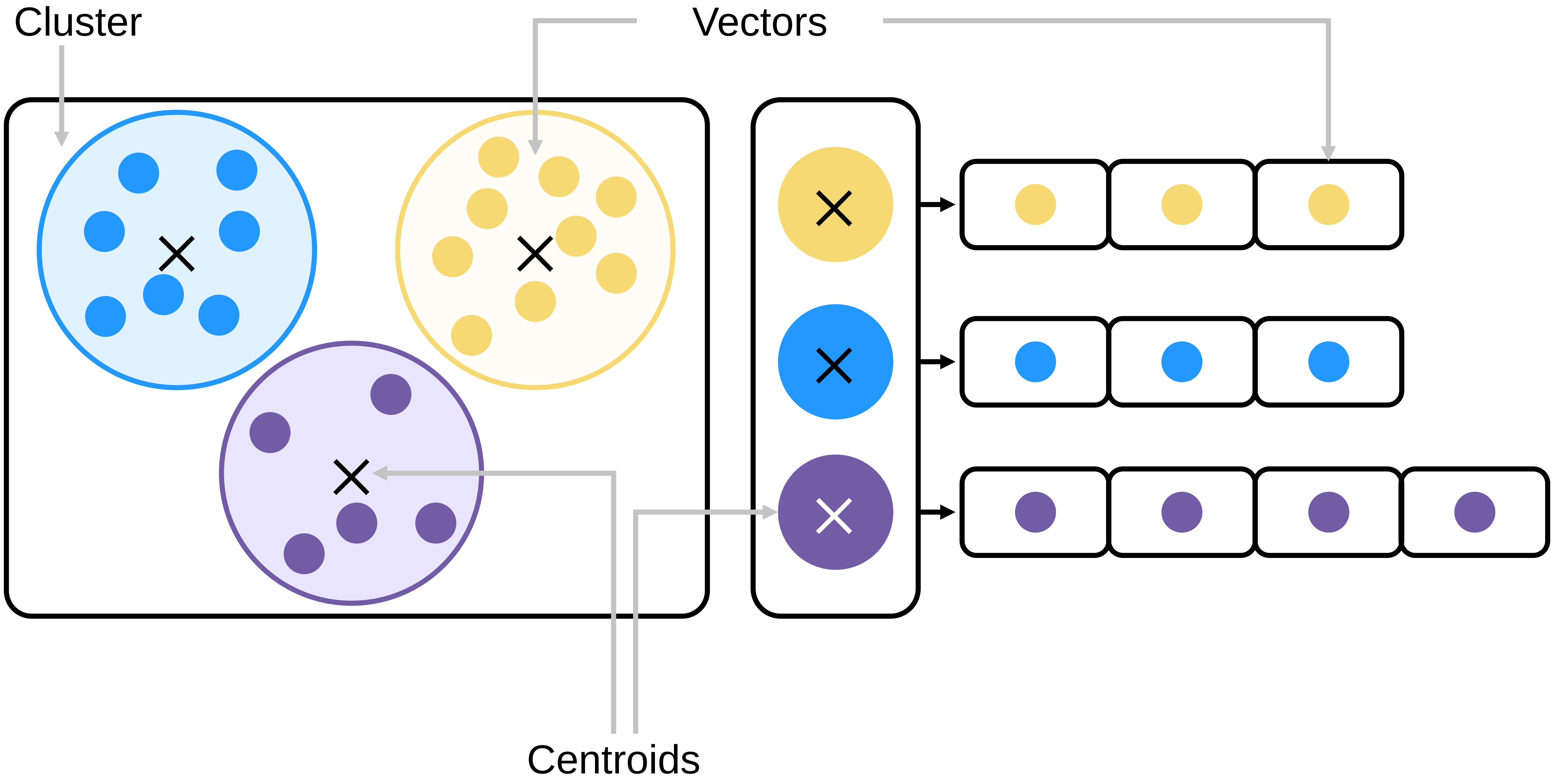
Consistent transactional insertion and searching

SQL Commands

```
CREATE TABLE table (
id Uint32,
embedding String,
INDEX idx_vector
GLOBAL USING vector_kmeans_tree
ON (embedding)
WITH (
    similarity=inner_product,
    vector_type=float,
    vector_dimension=1024),
PRIMARY KEY (id)
)
```

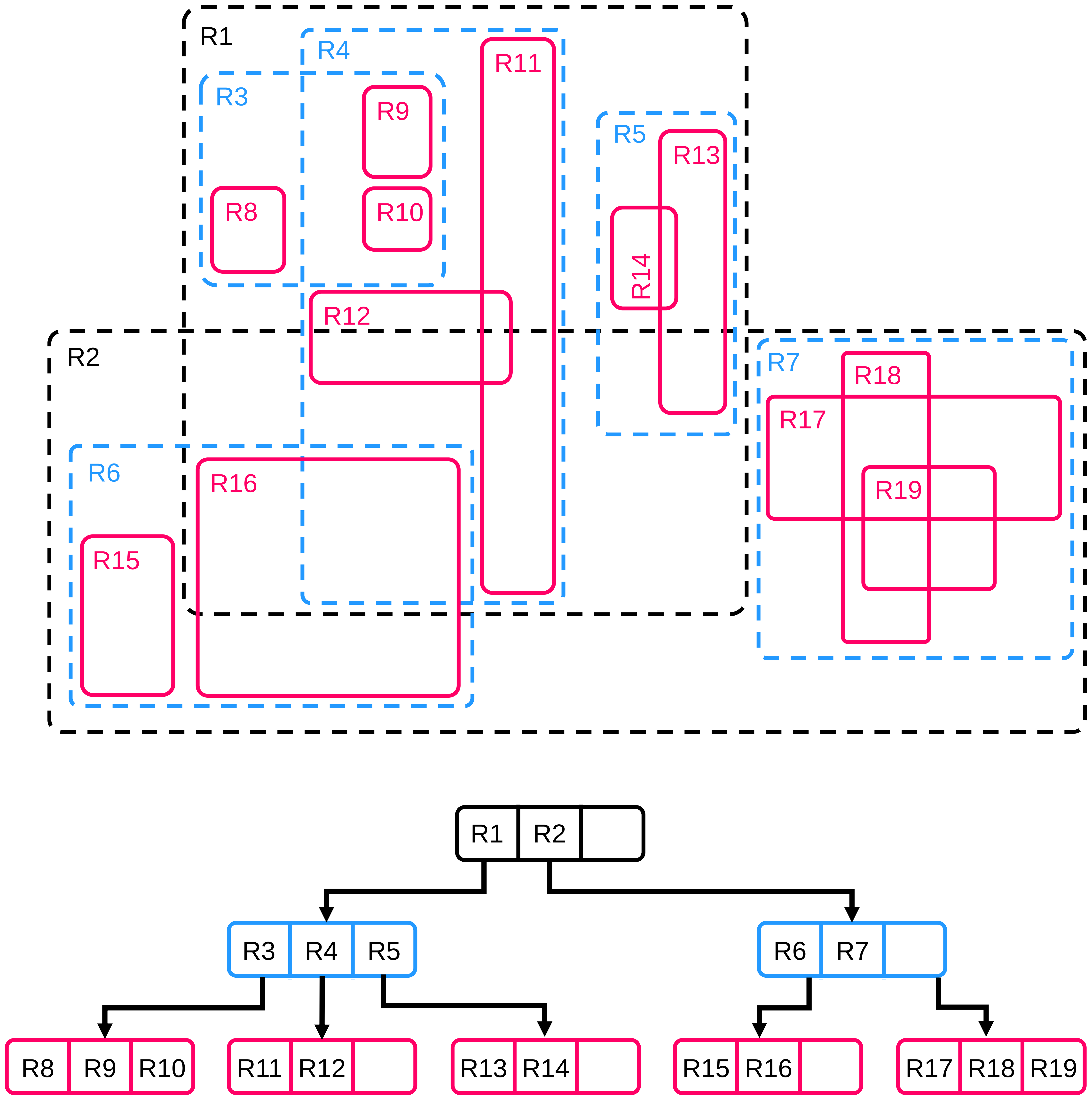
```
SELECT * FROM table
VIEW idx_vector
ORDER BY Knn::CosineDistance(
    embedding,
    $target)
LIMIT $k
```

Vector Index As An Inverted List

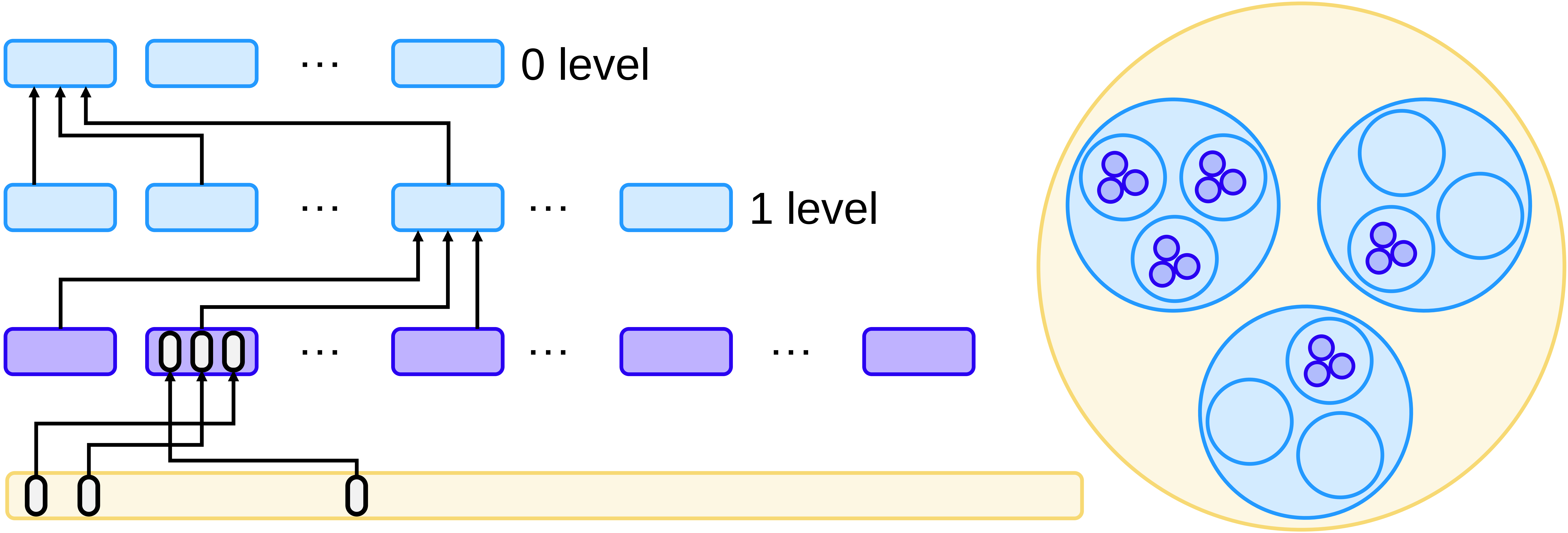


Search Space Pruning Algorithms

R-tree



Hierarchy Of YDB Vector Index Clusters



level table
(parent, id), centroid

posting table
(parent, PK), covered

indexed table
(PK), embedding, covered

Filterable Vector Index

```
SELECT * FROM table VIEW idx_vector
WHERE user_id = $target_user_id
ORDER BY Knn::CosineDistance(
  embedding,$target_embedding)
LIMIT $k;
```

Can’t filter

Before vector index

- can’t use a single index
- need for full scan

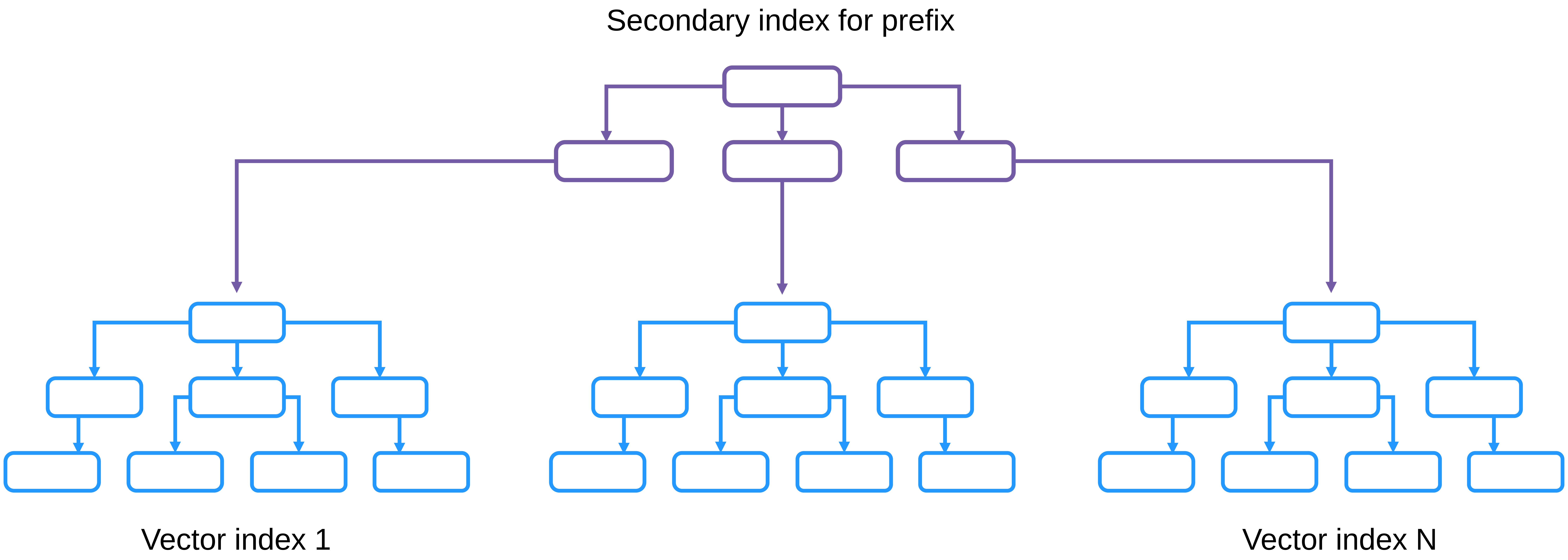
After vector index

- need for a repeat request

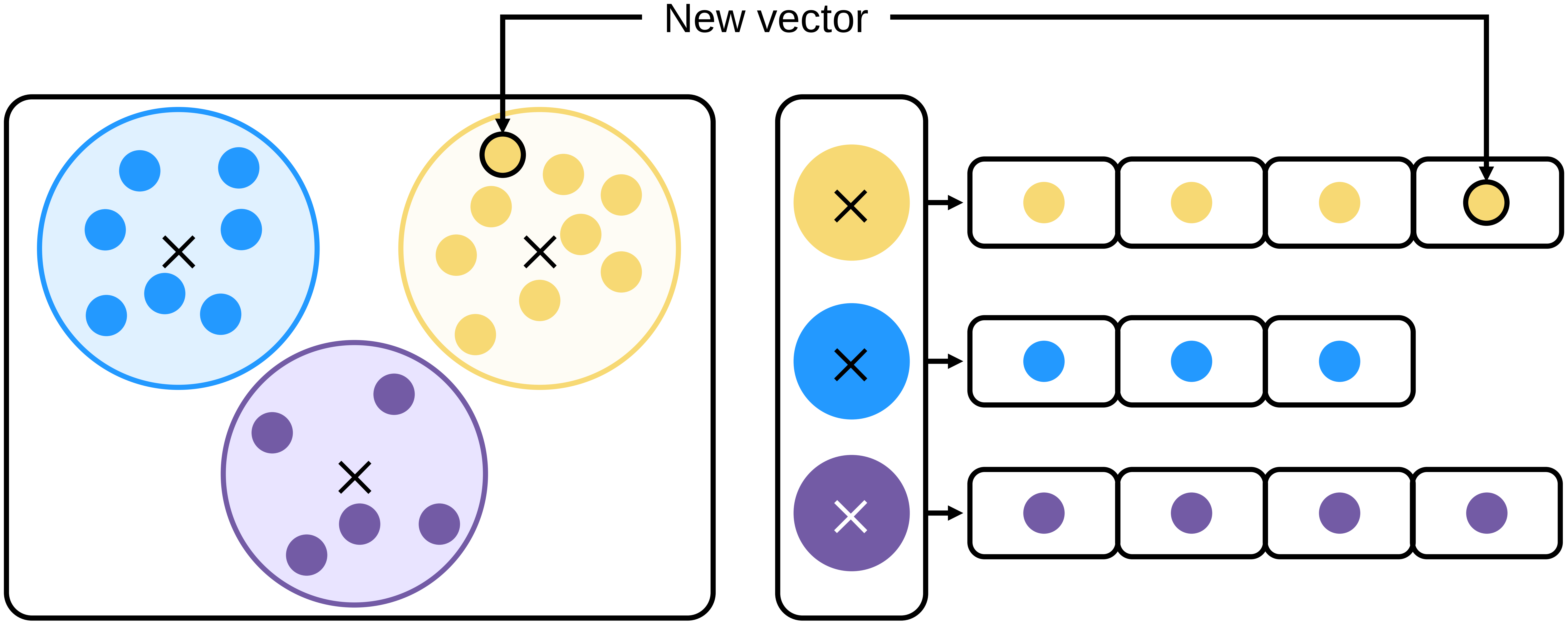
Need to filter

Inside index

Filtrable Vector Index



Insertion To Vector Index



- Just one centroid of one cluster would be changed
- The change needs to be pushed up through the levels

Let’s Stay In Touch

- How to try YDB?
- Why does it scale so well?
- Why is it so robust?
- What client utilities/ languages are supported?

Conclusion

ydb.tech

- YDB is a distributed database
- YDB as a platform offers sophisticated streaming and vector index
- There is a trend for RAG, especially streaming
- We are combining Big Data and AI