

# InterSystems IRIS: Achieving Speed and Usability in an Integrated Vector Database

*David Van De Griek, Boya Song, Philip Miloslavsky, Yuchen Liu,  
Yiwen Huang, Mark Hanson, Jeff Fried, Dmitriy Bochkov*

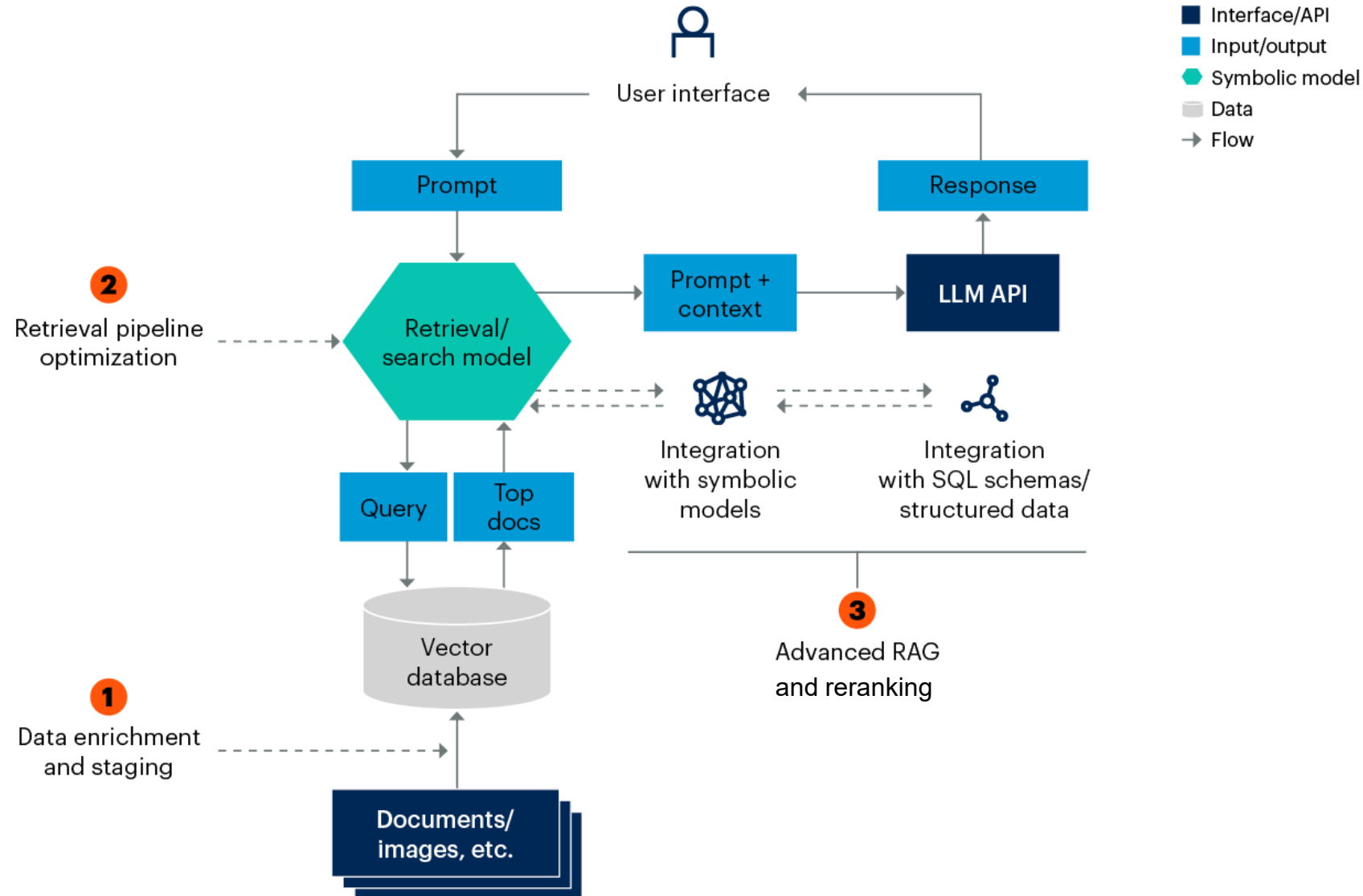
**NEDB 2025**

# Overview

- **New design for Integrated Vector DB**
- **Focused on RAG applications**
- **Based on:**
  - **multimodel core (“common data plane”)**
  - **vector data type and vector operations**
  - **compact storage model**
  - **vector-aware SQL optimizer**
  - **embedded Python**
- **Commercially in use**



# Vector Database as used in RAG applications



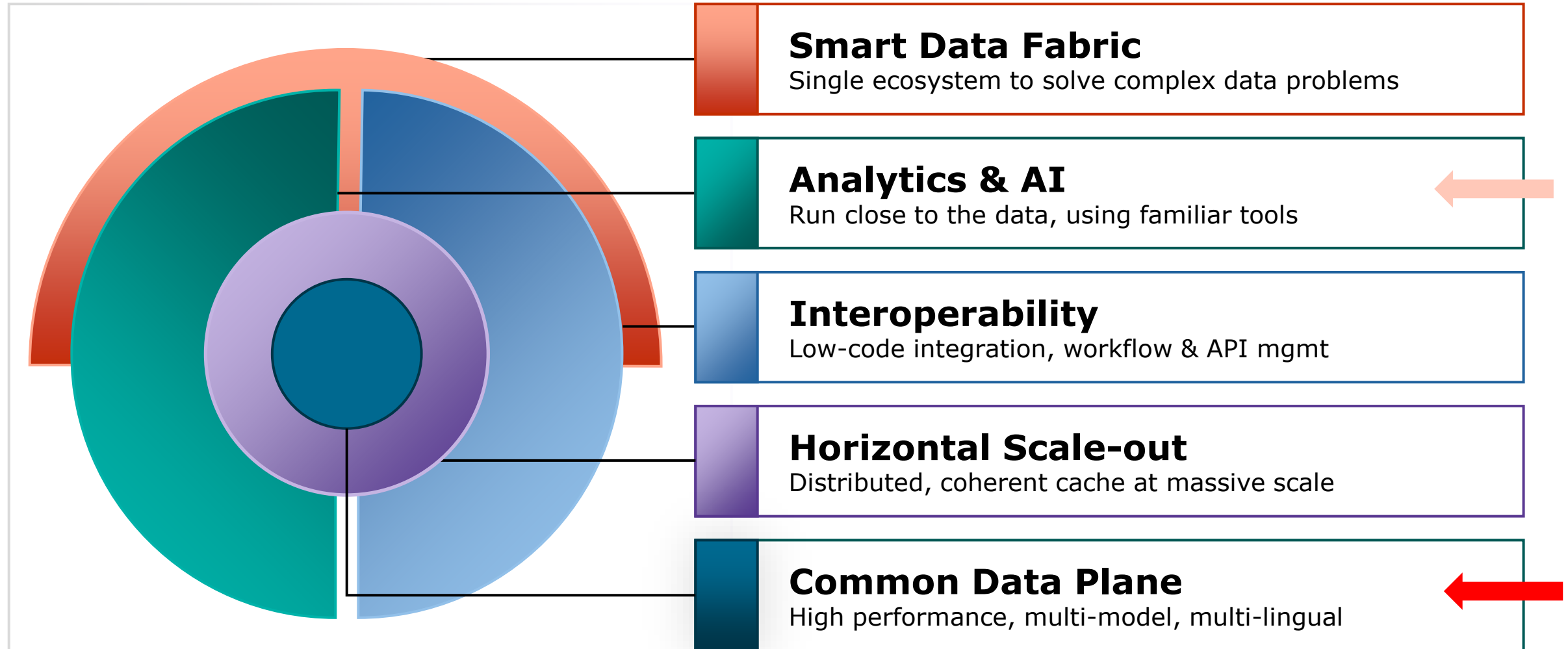
# Specialized and Integrated Vector Databases



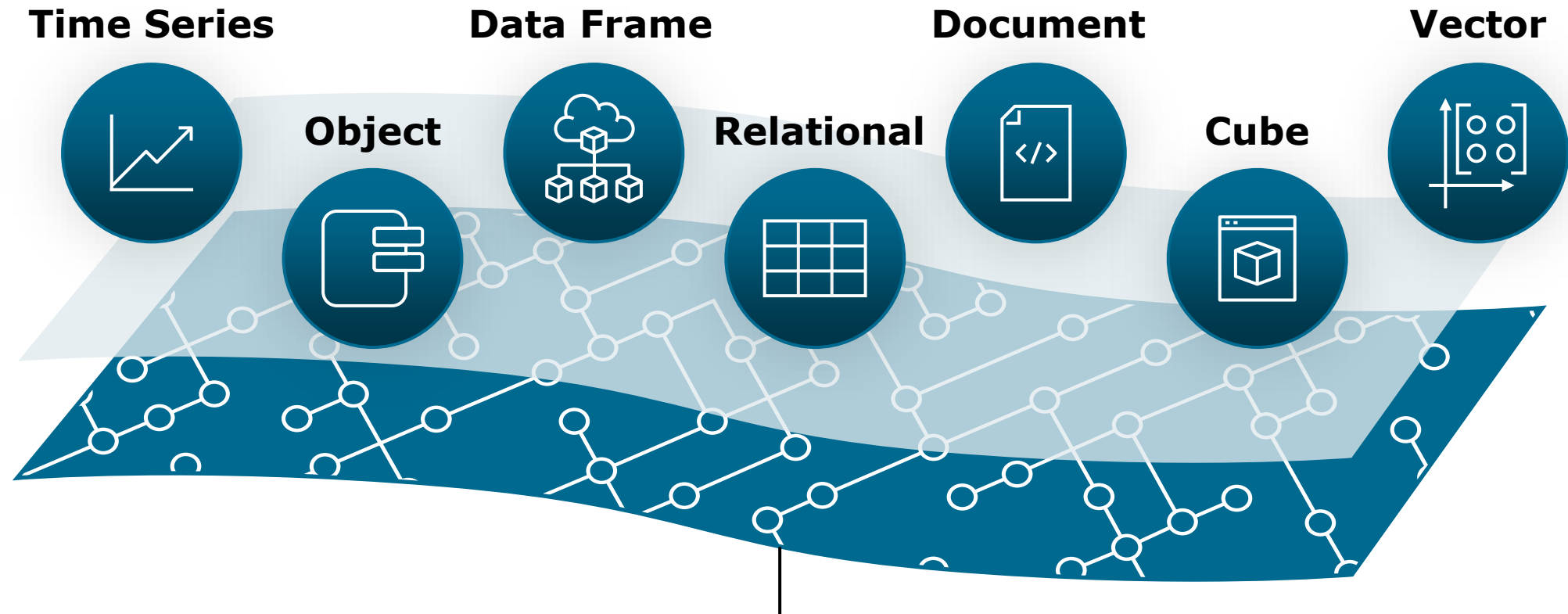
Criteria	Specialized Vector Databases	Integrated Vector Databases
Examples	Pinecone, Milvus, Weviate	PPASE, PostgreSQL+pgvector, ElasticSearch <b>InterSystems IRIS</b>
Ease of Use	Designed specifically for vector data	Integrated into existing databases, leveraging familiar interfaces and tools
Footprint	Typically requires additional infrastructure and resources	Utilizes existing database infrastructure, reducing the need for additional resources
Performance	Optimized for high-dimensional vector searches, often with advanced indexing techniques	Performance can vary based on the underlying database but benefits from integrated indexing



# InterSystems IRIS Architecture Layers



# The Common Data Plane



**$\wedge$ global( <key1>,<key2>,... ) = \$encoding( <val1>,<val2>,... )**

# Fast, Flexible Data Encodings

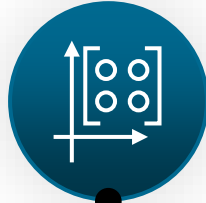


## Lists



```
$list("lcars",1138,88.0,...)
```

## Vectors



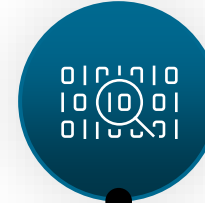
```
$vector(4.8,15.1,6.23,...)
```

## Documents



```
$pva({"id":10816,  
      "fname":"roy",  
      ... })
```

## Bitmaps



```
$bit(1,0,1,0,1,0,...)
```

# Projections



Document

```
{
  "location": "ICU5",
  "collector_name": "Smartlinx5",
  "sensor_name": "BP3",
  "bed_id": 8605,
  "readings": [
    { "start_time": "2024-07-04 10:12:03.642",
      "interval": 3.333,
      "values": [7200, 7300, 7700, 8500, 9100, ...]
    }
  ],
  ...
}
```

PUT

GET

Location	Sensor	Date	Time	Value
ICU5	BP3	2024-07-04	10:12:03.642	7200
ICU5	BP3	2024-07-04	10:12:06.975	7300
ICU5	BP3	2024-07-04	10:12:10.308	7700
ICU5	BP3	2024-07-04	10:12:13.641	8500
ICU5	BP3	2024-07-04	10:12:16.974	9100
...	...	...	...	...



Relational

SELECT

INSERT

`^wave(202407, "ICU5", 123) = $list("Smartlinx5", "BP3", 8605)`

`^wave(202407, "ICU5", 123, 1, "ts") = $list("2024-07-04 10:12:03.642", 3.333)`

`^wave(202407, "ICU5", 123, 1, "v") = $vector(7200, 7300, 7700, 8500, 9100, ...)`



# Design Goal



Unified, versatile data engine that supports vector fields and vector indices (e.g. HNSW index)

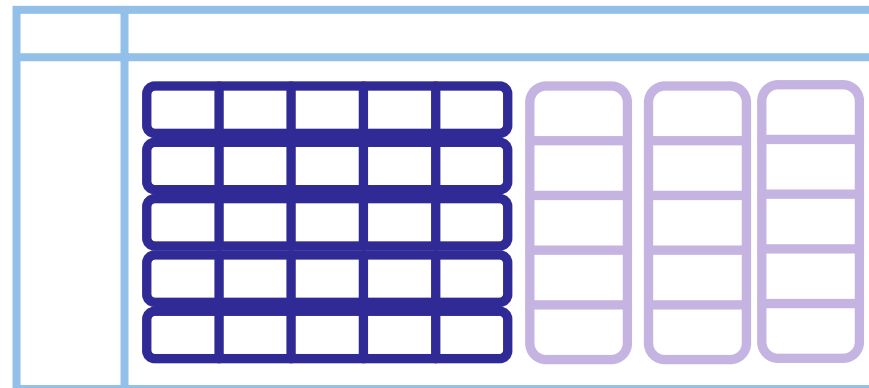
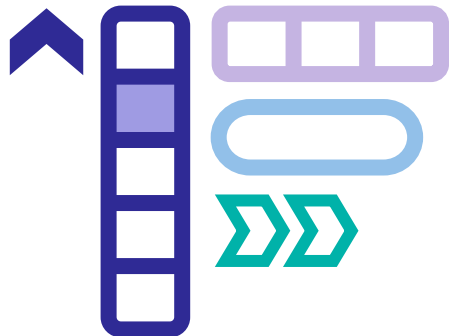
- Unified storage for data and index vector data, columnar data and regular data
  - Minimize data duplication
  - Index/field size not limited by memory size
- Unified SQL engine with great query processing capabilities, including
  - Transactions
  - Full SQL Support
  - Filtered Vector Search
  - Vector Range Search
  - Vector Range Join
  - Vector with Fulltext Search
  - Semantic Join

# Vector Storage Model evolved from columnar use cases



**Storage model** for SQL based on native \$vector data type to deliver key analytical querying facilities needed for next-generation Data Warehouses, Lakes and Lakehouses

- Aligns physical table layout with typical analytical access patterns
- \$vector language feature designed to support **translytical workloads**
- **Order of magnitude speedup** for analytical queries thanks to SIMD use and vectorized execution
- **Schema flexibility** - mixing row & column storage - is a key InterSystems IRIS differentiator
- **Indexing flexibility** – can use a columnar index on row-based storage, etc.



# Storage design



**Vector fields** stored as a collection of chunked vectors, 64K values per chunk.

Metadata (Columnar Index Map and Columnar Data Map) support fast sorting and SIMD operations.

Special %Vector type: %Embedding

Flexible design, including **ability to accommodate VERY long vectors**

- Store long vectors in their own globals for performance and footprint

- Provision for managing space/precision tradeoffs

  - \$vector can be integer, decimal FP; supports sparse encodings and different storage size magnitude for both sparse and dense encodings

- new DataDefinitionLocation property



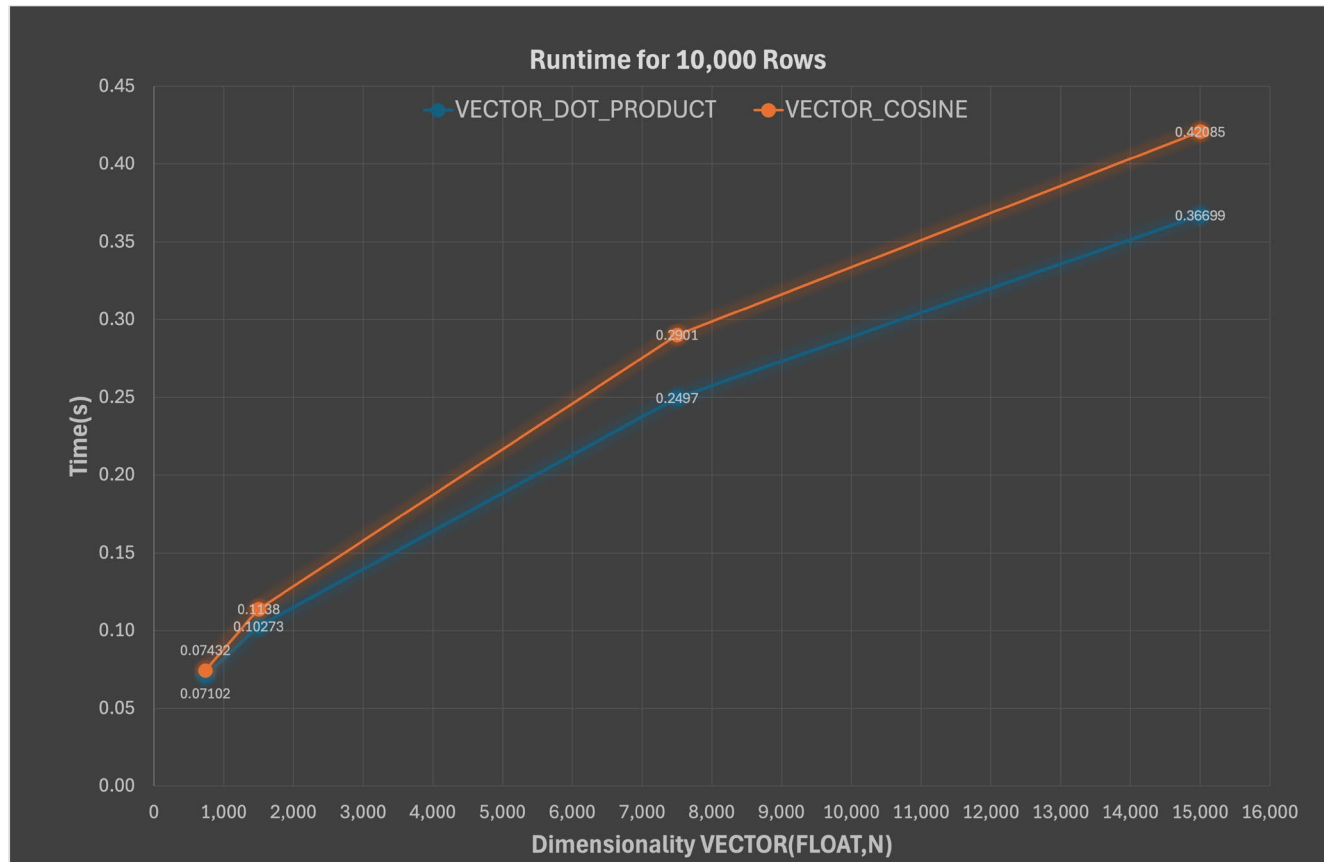
# Fast vector operations

- **Numeric Operations (scalar and vector-wise)**
  - \$VECTOROP("+", "-", "/" | ... | "cosine" | "dot-product", vector, vector | scalar, bitmap ) returns vector
- **String Operations (scalar and vector-wise)**
  - \$VECTOROP("\_", "lower" | "substring" | ... , vector, vector | scalar, bitmap ) returns vector
- **Filter Operations (scalar and vector-wise)**
  - \$VECTOROP("=", ">" | "<" | ... | "defined" | "undefined", vector, vector | scalar, bitmap) returns bitmap
- **Aggregate Operations**
  - \$VECTOROP("count" | "max" | "min" | "sum", vector, bitmap) returns scalar
- **Grouping Operations**
  - \$VECTOROP("group", "count" | "max" | "min" | "sum", vector, bitmap, list) modifies list
- **Miscellaneous Operations**
  - \$VECTOROP("convert", vector )
  - \$VECTOROP("mask", vector, scalar)
  - \$VECTOROP("positions", vector, bitmap)
  - \$VECTOROP("bytesize", vector)
  - ...



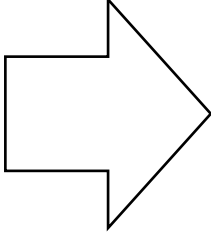
# Vector Dimensionality and Performance

- We can support vector storage and operation on high-dimensional vectors thanks to the fast \$vectorop
- Runtime of vector operation increases sub-linearly



# Query processing implementation: Unified SQL Engine



- Unified Storage Model
    - comparable access cost
      - for any storage type
      - for any index type
  - Universal Query Optimizer
    - pre-optimizer query rewrite
    - awareness of vector algorithms
    - multi-index plans
  - Adaptive Parallel Execution
    - data format agnostic
- 
- Efficient integration of vector/embedding data type
  - Example: Semantic Join
    - join with a similarity condition on word embeddings <sup>[1]</sup>
    - tolerates misspellings and different formats to deliver more join results



# Semantic Join in SQL



```
select
    FilmA.title Film1,
    FilmB.title Film2,
    ReviewA.star_rating * ReviewB.star_rating CombinedRating
from Cinema.Film FilmA
join Cinema.Film FilmB
    on (vector_cosine(FilmA.overview_embedding, FilmB.overview_embedding)>.55)
join Cinema.Review ReviewA on (FilmA.imdb_id=ReviewA.imdb_id)
join Cinema.Review ReviewB on (FilmB.imdb_id=ReviewB.imdb_id)
where FilmA.imdb_id<FilmB.imdb_id
    and FilmA.release_year>1950
    and FilmB.length>60
order by CombinedRating desc, FilmA.imdb_id, FilmB.imdb_id
```

# Semantic Join Result



DBeaver 23.1.2 - <USER> cinema.sql

```
<USER> Script-1 * <USER> cinema.sql X
select
  FilmA.title Film1,
  FilmB.title Film2,
  ReviewA.star_rating * ReviewB.star_rating CombinedRating,
  FilmA.overview,
  FilmB.overview
from Cinema.Film FilmA join
     Cinema.Film FilmB on (vector_cosine(FilmA.overviewembedding, FilmB.overviewembedding) > .55) JOIN
     Cinema.Review ReviewA on (FilmA.imdb_id = ReviewA.imdb_id) JOIN
     Cinema.Review ReviewB on (FilmB.imdb_id = ReviewB.imdb_id)
where FilmA.imdb_id != FilmB.imdb_id
order by CombinedRating desc, FilmA.imdb_id, FilmB.imdb_id
```

Film 1 X

select FilmA.title Film1, FilmB.title Film2, ReviewA.star\_rating CombinedRating, FilmA.overview, FilmB.overview

	ABC Film1	ABC Film2	ABC CombinedRating	ABC overview	ABC overview
1	2001: A Space Odyssey	2010	25	Humanity finds a mysterious object buried beneath the	This is a sequel to 2001 A Space
2	The Godfather	The Godfather: Part II	25	Spanning the years 1945 to 1955, a chronicle of the fict	In the continuing saga of the Corl
3	The Godfather: Part II	The Godfather	25	In the continuing saga of the Corleone crime family, a yc	Spanning the years 1945 to 1955
4	Star Wars	The Empire Strikes Back	25	Princess Leia is captured and held hostage by the evil Ir	The epic saga continues as Luke
5	Star Wars	Return of the Jedi	25	Princess Leia is captured and held hostage by the evil Ir	As Rebel leaders map their strate
6	The Empire Strikes Back	Star Wars	25	The epic saga continues as Luke Skywalker, in hopes of	Princess Leia is captured and hel
7	The Empire Strikes Back	Return of the Jedi	25	The epic saga continues as Luke Skywalker, in hopes of	As Rebel leaders map their strate
8	Return of the Jedi	Star Wars	25	As Rebel leaders map their strategy for an all-out attack	Princess Leia is captured and hel
9	Return of the Jedi	The Empire Strikes Back	25	As Rebel leaders map their strategy for an all-out attack	The epic saga continues as Luke
10	2010	2001: A Space Odyssey	25	This is a sequel to 2001 A Space Odyssey. It is now 201	Humanity finds a mysterious obje
11	The Godfather	The Godfather: Part III	20	Spanning the years 1945 to 1955, a chronicle of the fict	In the midst of trying to legitimiz
12	The Godfather: Part II	The Godfather: Part III	20	In the continuing saga of the Corleone crime family, a yc	In the midst of trying to legitimiz
13	The Godfather: Part III	The Godfather	20	In the midst of trying to legitimize his business dealings	Spanning the years 1945 to 1955
14	The Godfather: Part III	The Godfather: Part II	20	In the midst of trying to legitimize his business dealings	In the continuing saga of the Corl
15	The Hunt for Red Octo	Crimson Tide	20	A new Soviet nuclear missile sub (a Boomer) heading o	On a US nuclear missile sub, a yo
16	Crimson Tide	The Hunt for Red Octo	20	On a US nuclear missile sub, a young first officer stages	A new Soviet nuclear missile sub

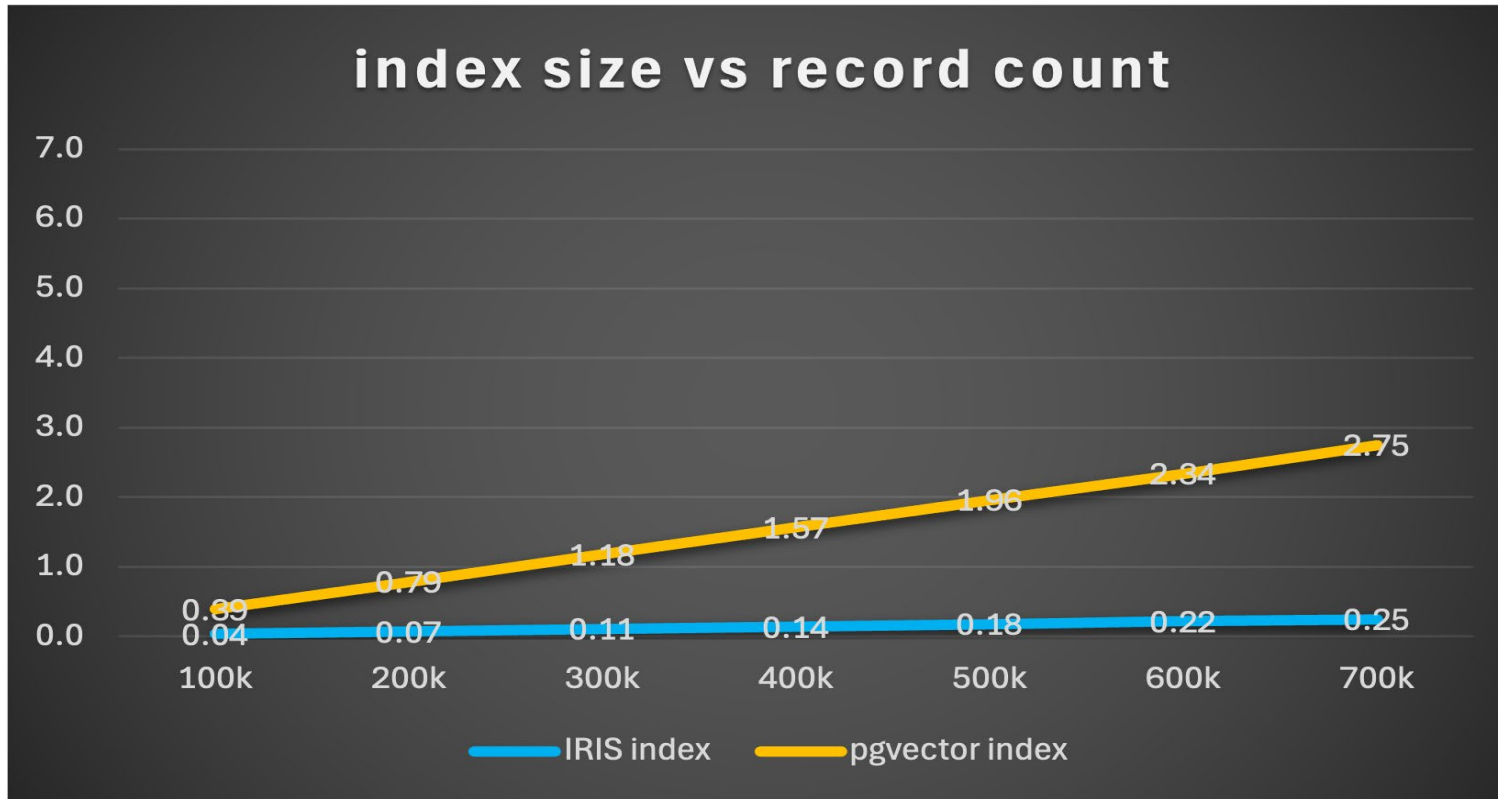
Refresh Save Cancel Export data 200 180 180 row(s) fetched - 86ms, on 2024-12-18 at 15:17:23

EST en\_US Writable Smart Insert 68 : 73 : 3968 Sel: 0 | 0

# Query processing implementation: HNSW index



- Implemented HNSW index according to Malkov 2016
- No need to store the vectors in the index as the SQL engine can access the original vector field



Reference: Yury Malkov, Dmitry A. Yashunin,  
**Efficient and Robust Approximate Nearest  
Neighbor Search Using Hierarchical  
Navigable Small World Graphs**  
IEEE TPAMI 2016 [arXiv link](#)

- Challenge: needs to be compatible with parallel INSERT

# SerenityGPT – built on InterSystems IRIS & Vector Search



How it works

About our product

Benefits

Security

Our team

FAQ

Book a Demo



S E R E N I T Y G P T



## Still searching with keywords? Frustrated with the results? Struggling to search across all your data?

SerenityGPT indexes all the content you want, lets you ask natural language questions –just like asking a teammate– and get accurate, unified, real-time responses.  
We'll even set it up for you!

Book your demo today



N

Already with us

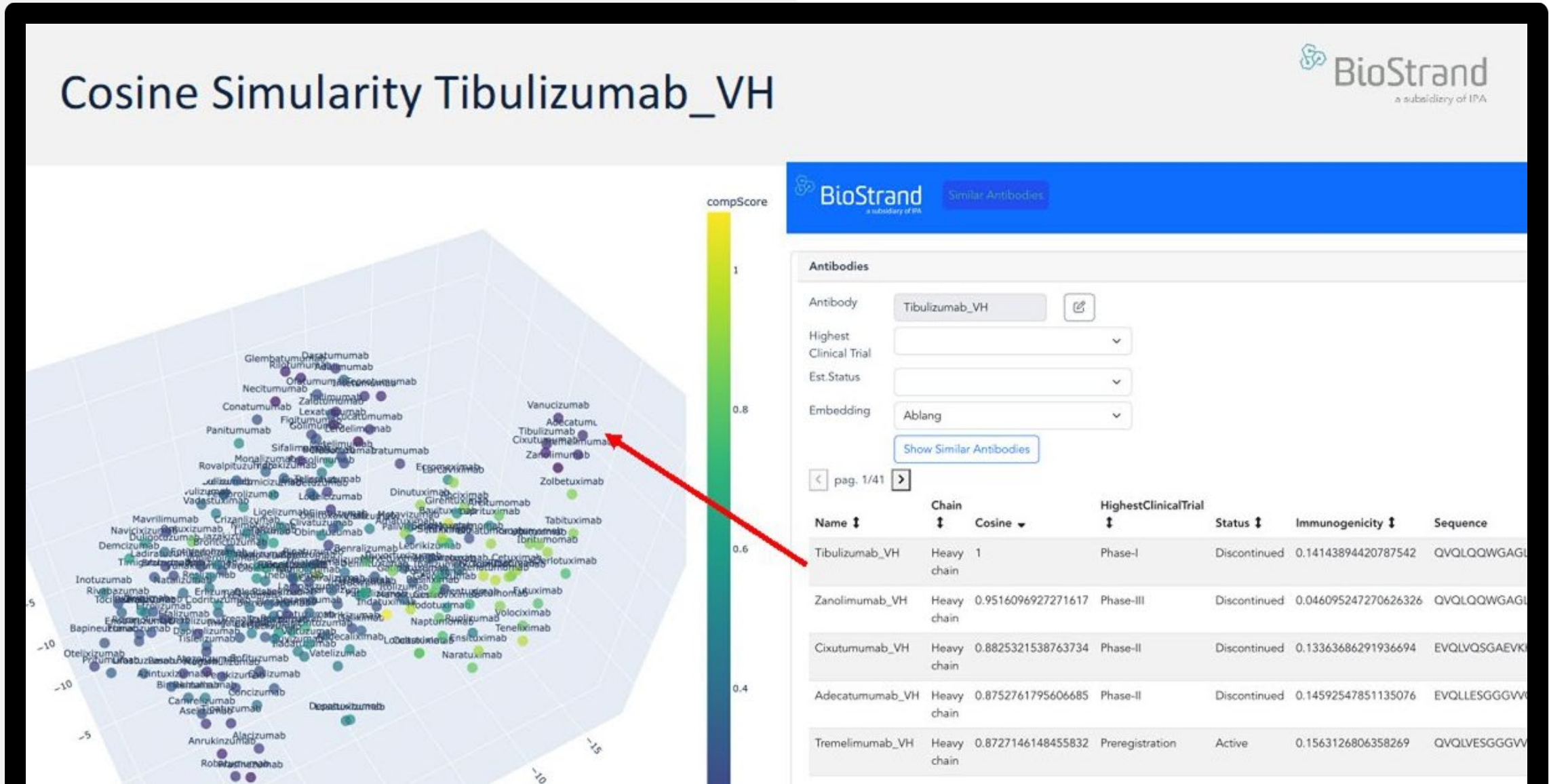
 InterSystems®

 Testronic  
A Catalis Company

PERFORCE



# Biostrand: Complex Data Analysis With IRIS Vector Search





Contact:



[Jeff.Fried@InterSystems.com](mailto:Jeff.Fried@InterSystems.com)



[@jeffried](https://twitter.com/jeffried)

# Thank you