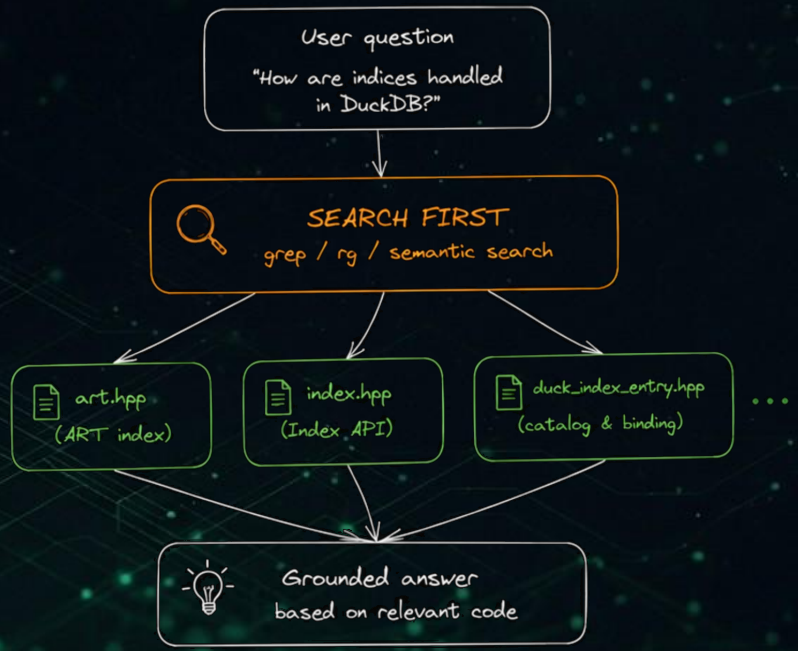
Grep your lakehouse

Search-first retrieval for DuckDB-powered agents

Sylvain Utard · x.com/**sylv_1_** · github.com/**redox**

Altertable

Coding agents keep doing it




```
● Ready. What would you like to work on?

✻ Brewed for 6s

  tell me how duckdb is handling indices

  Bash(find /Users/redox/dev/altertable-ai/duckdb/src -type f -name "*.hpp" | xargs grep -l "Index\|ART" | grep -i "index\|art" | head -20)
    Running…
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/function/partition_stats.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/storage/index.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/storage/index_storage_info.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/storage/storage_index.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/common/index_vector.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/common/column_index_map.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/common/index_map.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/common/table_index.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/common/column_index.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/common/radix_partitioning.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/common/projection_index.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/common/hive_partitioning.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/execution/partition_info.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/catalog/catalog_entry/index_catalog_entry.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/catalog/catalog_entry/duck_index_entry.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/optimizer/join_order/relation_index.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/parser/parsed_data/create_index_info.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/function/window/window_index_tree.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/storage/table/table_index_list.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/planner/operator/logical_create_index.hpp

  Bash(find /Users/redox/dev/altertable-ai/duckdb/src/include -type f | xargs grep -l "class.*Index" 2>/dev/null | head -10)
    Running…
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/transaction/commit_state.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/planner/binder.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/planner/logical_tokens.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/catalog/catalog.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/storage/data_table.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/storage/index.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/storage/storage_index.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/storage/checkpoint_manager.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/common/index_vector.hpp
    /Users/redox/dev/altertable-ai/duckdb/src/include/duckdb/common/enum_util.hpp
```

tell me more about how indices are handled in duckdb

Thought for 2s

Exploring the DuckDB codebase to explain how indices are implemented and used.

Explored 14 files, 13 searches

Searched How are indices created and managed i...

Grepped CREATE INDEX|IndexType|index_type

Searched files **/index/**

Thought briefly

Grepped RegisterIndexType|IndexTypes|ART

Searched How does the optimizer use indexes fo...

Read art.hpp L1-120

Read index_constraint_type.hpp L1-36

Every agent has the same first instinct:
THEY NEED TO SEARCH FIRST

Altertable

That same pattern is coming to data.

Yet…

- *Data Analyst agents are only really good at Text-to-SQL: “How to query X?”*
- *Knowing what table or column to use is not always straightforward!*

Sure a semantic layer is helping, especially in the BI space where the whole schema can be documented.

➡ The **missing primitive** is **schema-agnostic retrieval**

I'm Sylvain, and I spent 13y building search engines

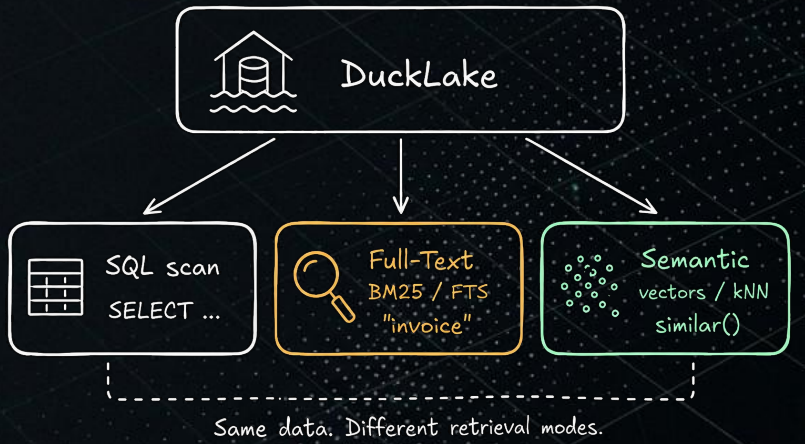
- *2008-2013: C++ Engineer working for the french Google alternative*
- *2013-2021: 1st employee & VP Engineering **@Algolia**, search as a service startup*
- *2021-2025: B2C experience **@Sorare**, data at scale*
- *Now: Co-founder & CEO **@Altertable**, data runtime for the AI era*
  - *Lakehouse with federation (we hate pipelines)*
  - *Knowledge graph & always-on AI agents*
  - *SQL, CLI, MCP, Flight SQL, APIs, Iceberg, S3 ...*

TLDR; we extended DuckLake with search capabilities: `ducklake_search`

Altertable

Bringing Full-Text & Semantic search to DuckLake

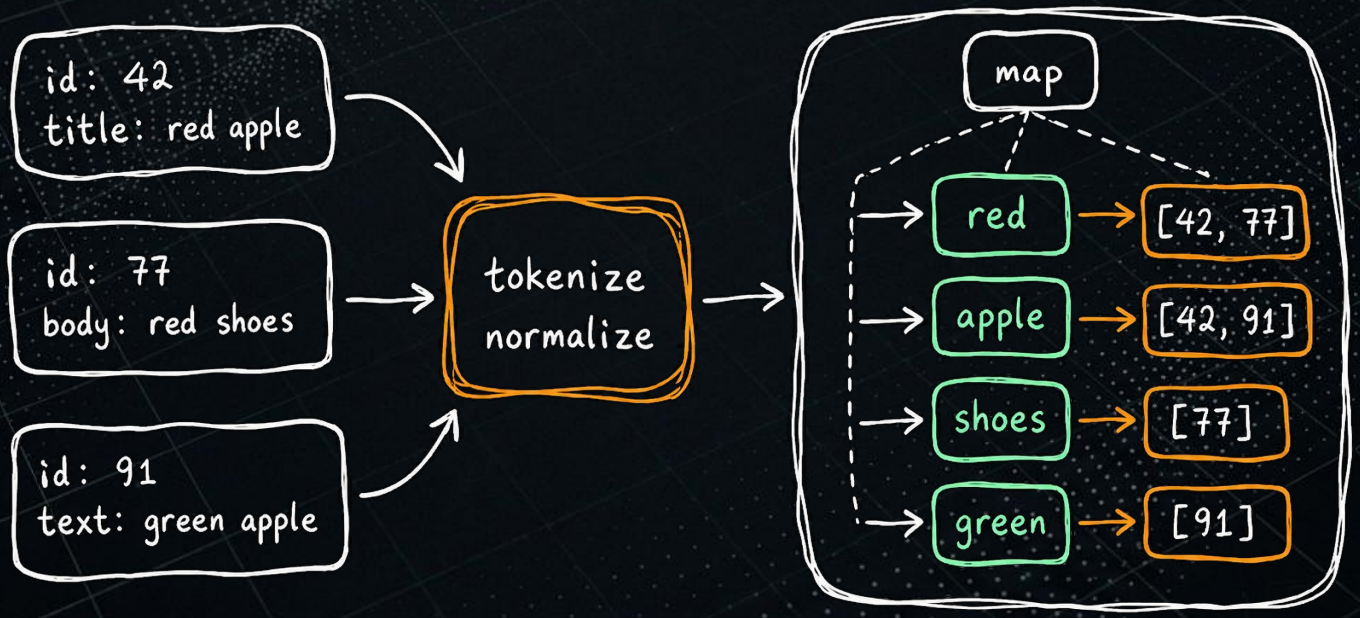
- The data already lives in the lakehouse
- Nobody likes building pipelines to keep a DB in sync with a search engine



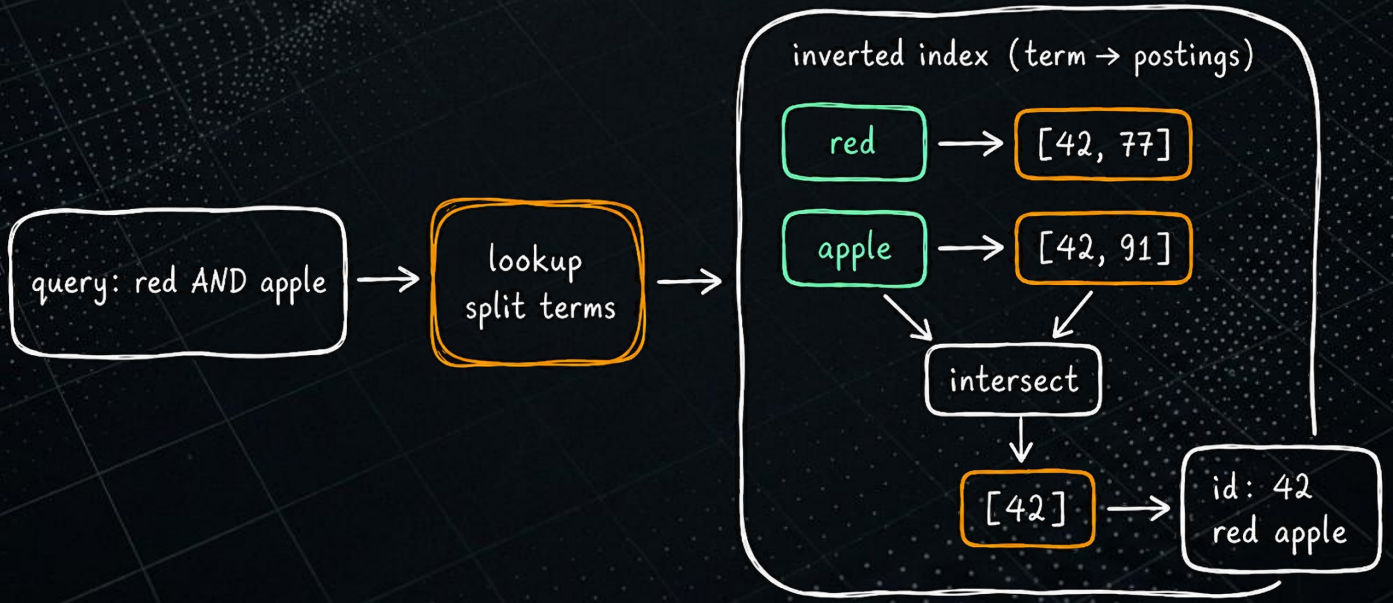
➡ We should add new retrieval modes on top of the same storage layer

Altertable

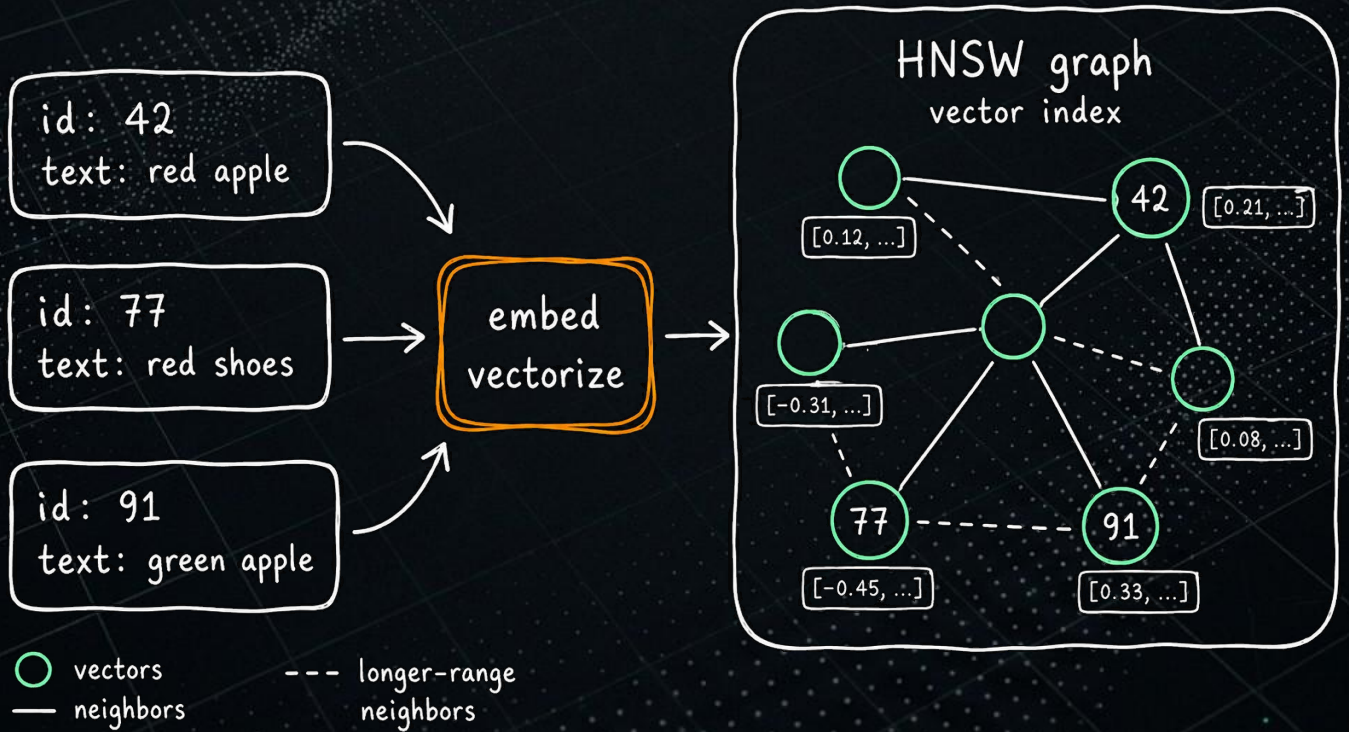
Reminder: what does full-text search mean?



Reminder: how does full-text search work?

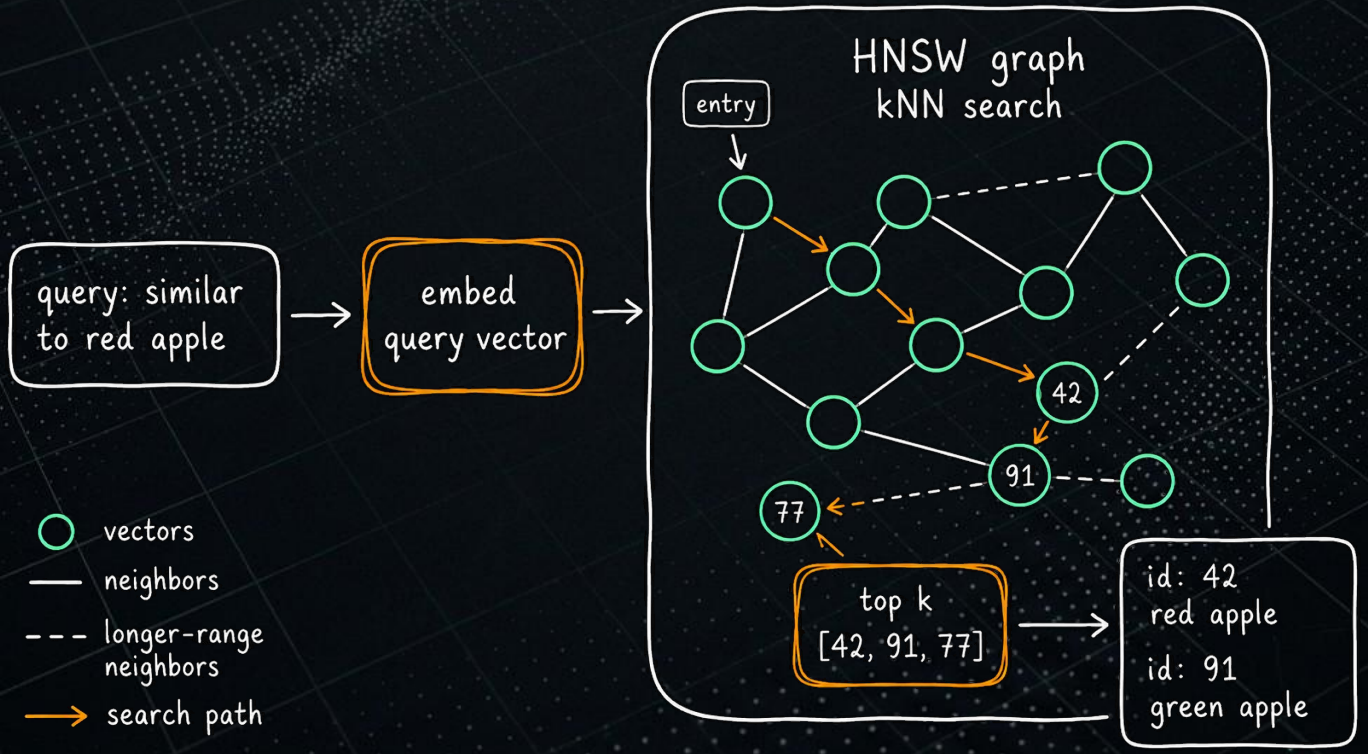


Reminder: what does semantic search indexing mean?



Altertable

Reminder: how does semantic search work?



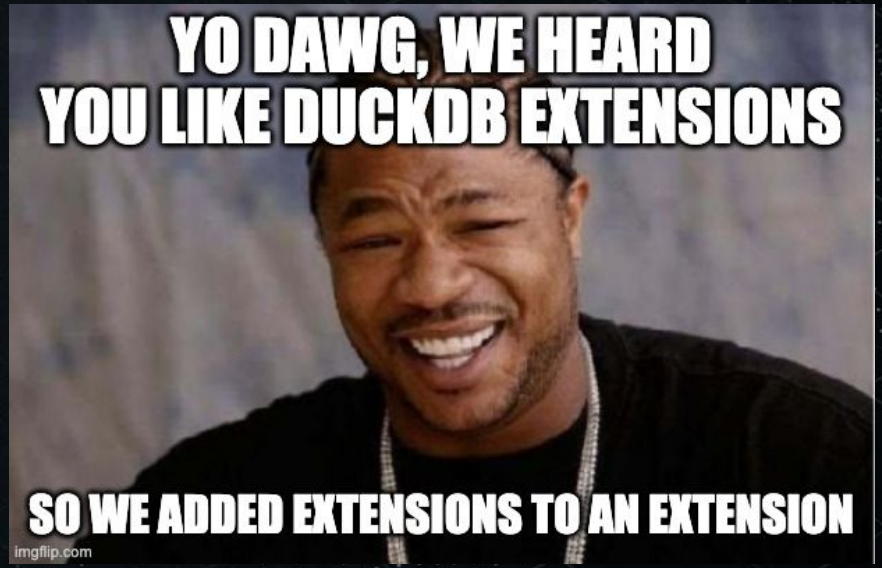
Altertable

Extending DuckLake

Extending DuckLake

- Inspired by DuckDB's extensions, we extended the DuckLake extension with extensions

- Built `ducklake_search` on top of
  - DuckLake
  - Tantivy
  - hnsw-rs
  - turboquant
  - model2vec-rs



Extending DuckLake's SQL surface

"It's all just SQL"

Altertable

Extending DuckLake's SQL surface

Creating the indices

```
CREATE INDEX fts_index_name ON table_name USING FTS(col1, col2) WITH (option1, option2)
```

```
CREATE INDEX semantic_index_name ON table_name USING SEM(col1, col2) WITH (option1, option2)
```

Extending DuckLake's SQL surface

And do something like

```
SELECT
    *
FROM
    tickets
WHERE
    created_at >= now() - INTERVAL 30 days
    AND search(message, 'scalable lakheouse') -- full-text (fuzzy) search
LIMIT 10
```

Extending DuckLake's SQL surface

Actually, rather this

```
SELECT
    *
FROM
    tickets
WHERE
    created_at >= now() - INTERVAL 30 days
    AND message @@ 'scalable lakheouse' -- full-text (fuzzy) search
LIMIT 10
```

Extending DuckLake's SQL surface

In reality, you want to manipulate the relevance score:

```
SELECT
   *
FROM
   tickets
WHERE
   created_at >= now() - INTERVAL 30 days
   AND message @@ 'scalable lakheouse' -- full-text (fuzzy) search
ORDER BY
   score DESC -- virtual column reflecting match relevance
LIMIT 10
```

Extending DuckLake's SQL surface

Or even searching in all columns

```
SELECT
   *
FROM
   tickets
WHERE
   created_at >= now() - INTERVAL 30 days
   AND * @@ 'scalable lakheouse' -- full-text (fuzzy) search all columns
ORDER BY
   score DESC -- virtual column reflecting match relevance
LIMIT 10
```

Extending DuckLake's SQL surface

Or do semantic search

```
SELECT
   *
FROM
   tickets
WHERE
   created_at >= now() - INTERVAL 30 days
   AND * <=> 'GDPR EU Residency' -- semantic search
ORDER BY
   score DESC -- virtual column reflecting match relevance
LIMIT 10
```

Extending DuckLake's SQL surface

Well, as you can imagine; this becomes quite powerful:

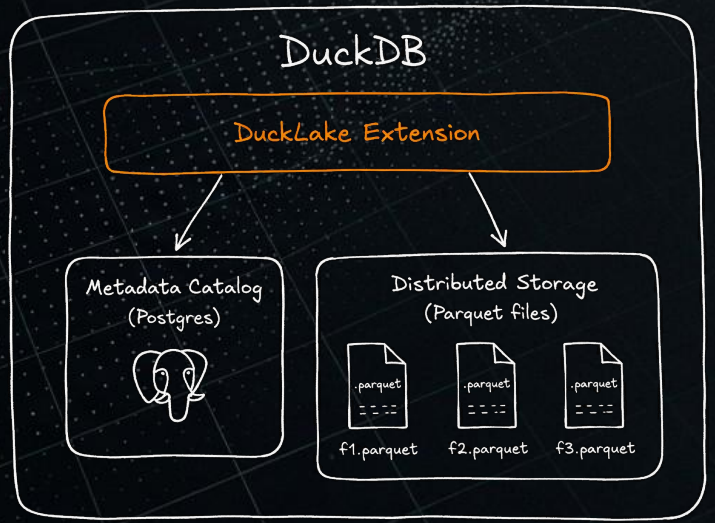
```
WITH candidates AS (
      SELECT ticket_id, workspace_id, priority, created_at
      FROM tickets
      WHERE * <=> 'GDPR EU Residency' -- semantic search
)
SELECT workspace_id, priority, COUNT(*) AS ticket_count
FROM candidates
GROUP BY 1, 2
ORDER BY ticket_count DESC;
```

Implementing `ducklake_search`

“The elegance of the abstractions isn't affected by the shittiness of the implementations.” ©

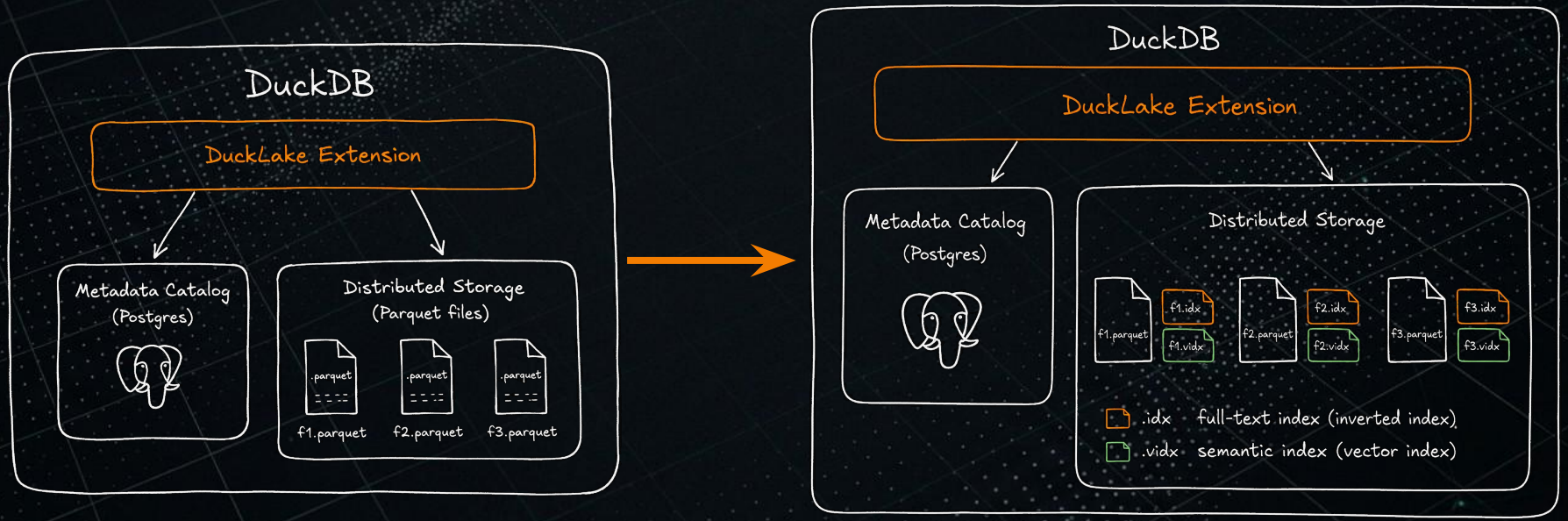
Altertable

Building search indices as parquet sidecars



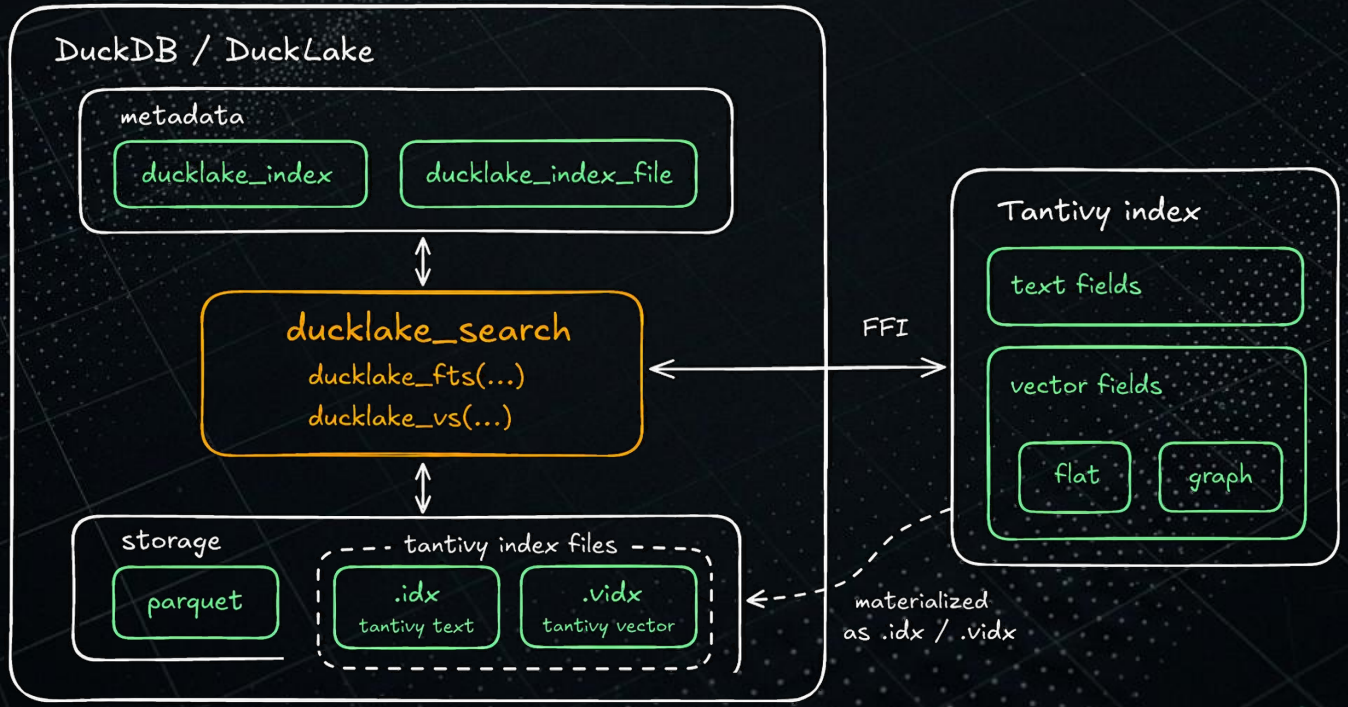
Altertable

Building search indices as parquet sidecars



Altertable

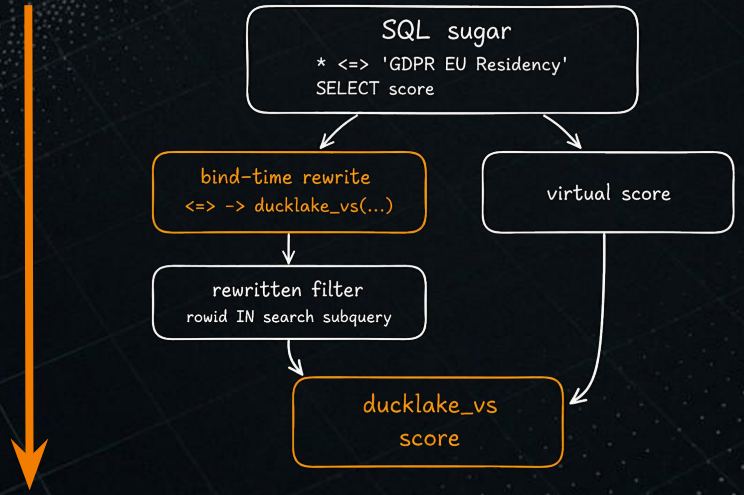
Implementing `ducklake_search`



Altertable

Rewriting the query to work smoothly with DuckLake

```
SELECT * FROM tickets WHERE * <=> 'GDPR EU Residency'
```




```
SELECT * FROM tickets WHERE rowid IN (
    SELECT rowid FROM ducklake_vs('*', 'GDPR EU Residency')
) ORDER BY score DESC -- simplified version
```

Altertable

TLDR;

- DuckLake's bet: **it's all just SQL**. So does `ducklake_search`.
- DuckLake extensions would be an excellent super-secret next big thing
- We love OSS and would love to open-source it
- A *lot* of simplifications has been made for the talk
- I hate pipelines and you should too


altertable.ai
“Follow us, even if you like pipelines” © Altertable