Knowledge Graph

On

OpenSearch

Lior Perry

Open Source

- Apache 2 License
- Community Driven
- Open architecture
- Build for search & scale

Open Search

Yang Db

Relationship Matters

Many of the real world use cases resembling graph more than anything else

Lets ask questions that closely relate to their business entities with an appropriate language

Focusing only on the entities is often missing the major part of the data insight

Graph databases Are addressing these issues

* Support a labeled property entities / relationships

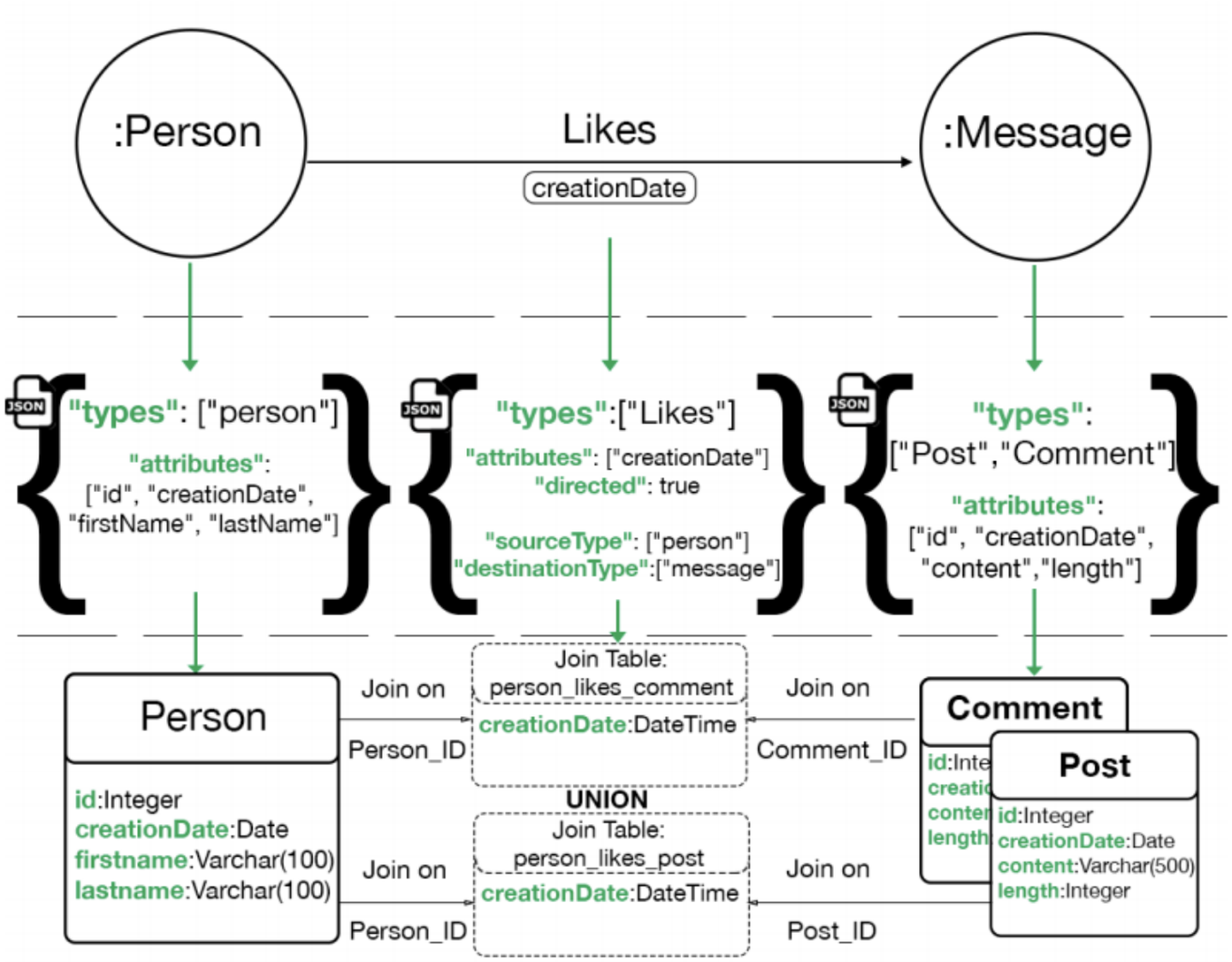
* Allow declarative semantics for asking questions

* Hide physical store complexities from clients

Abstraction of the physical aspects

In this Cypher query we only focus on the logical entities and relationships

The underlying elements store may be implemented in different ways depending on different requirements - latency, space, throughput



Thing are in motion…

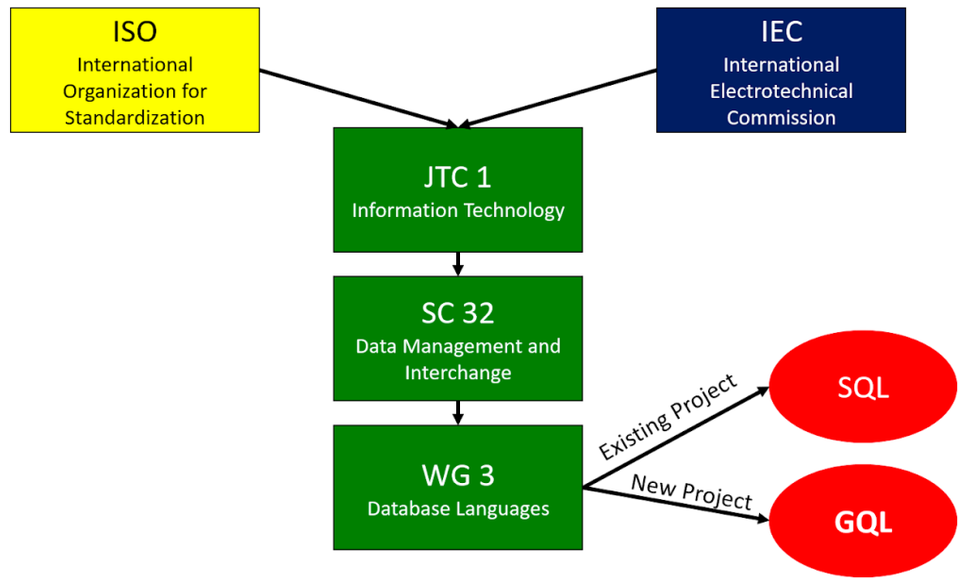
The community has driven the industry and now the graph extension for SQL is underway

- Major players

- Strong Industry support

- Advanced Stage

Oracle, Neo4j, TigerGraph, IBM, SAP/Sybase, JCC Consulting




```
  SELECT owner.name AS account_holder, SUM(t.amount) AS total_transacted_with_Nikita
    FROM MATCH (p:Person) <-[:owner]- (account1:Account)
       , MATCH (account1) -[t:transaction]- (account2) /* match both incoming and outgoing transactions */
       , MATCH (account2:Account) -[:owner]-> (owner:Person|Company)
   WHERE p.name = 'Nikita'
GROUP BY owner
```

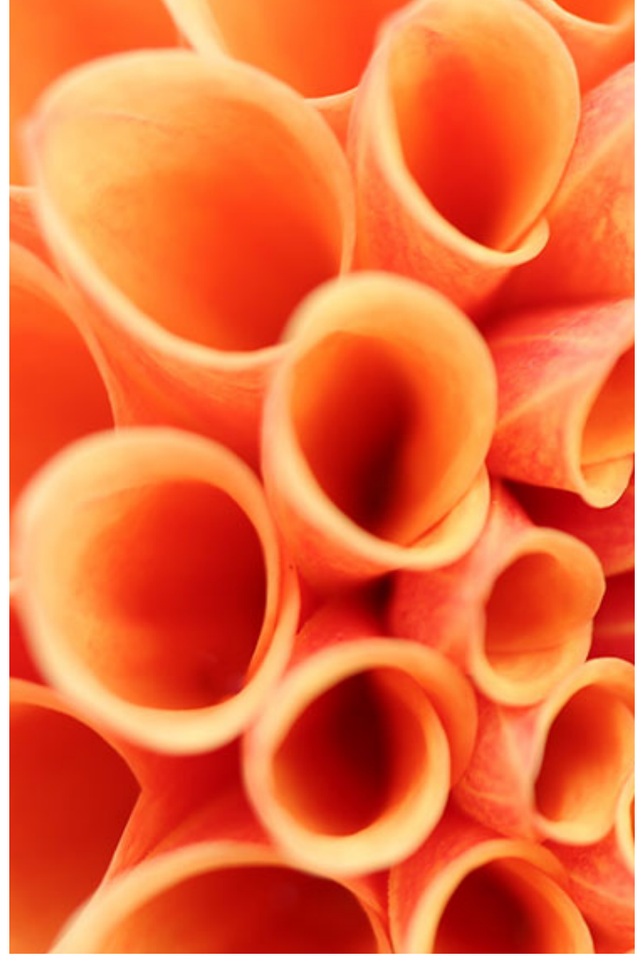
Domain Evolution & Multi-Tenant

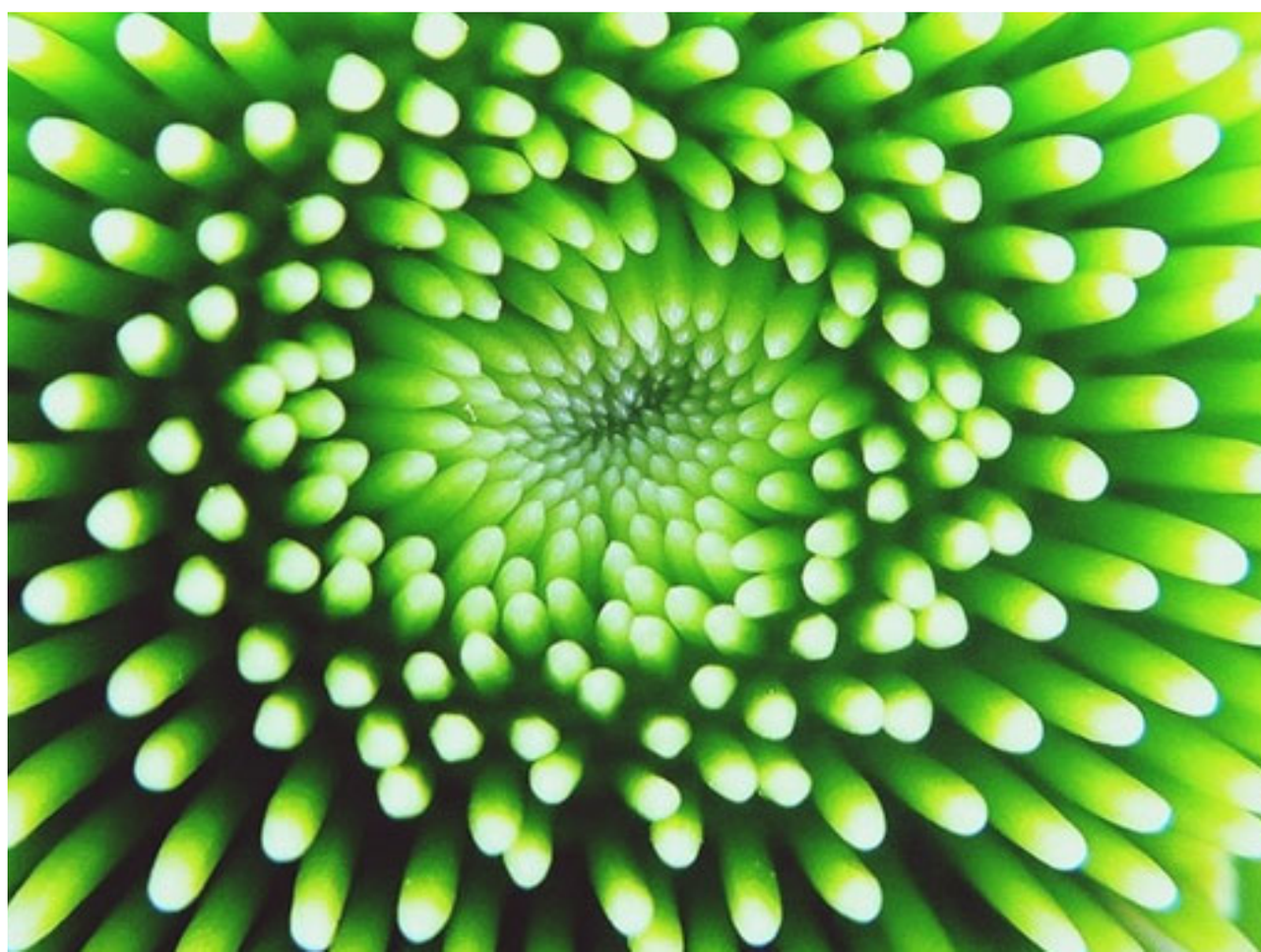
In many cases we would like to change or decouple (evolve) some business entities on our dataset, such changes may cause us to **re-index** our store according to our new domain model.

In the logical ontology we are making a strong effort separating these logical-physical entities so that many of the logical changes would not necessary force us to reindex the data.

Such capability can save us time & money and free us from the fear of changing data entities in our domain.

Allowing different projections and domain entities on the same physical layer using a different catalog for every tenant is a great advantage





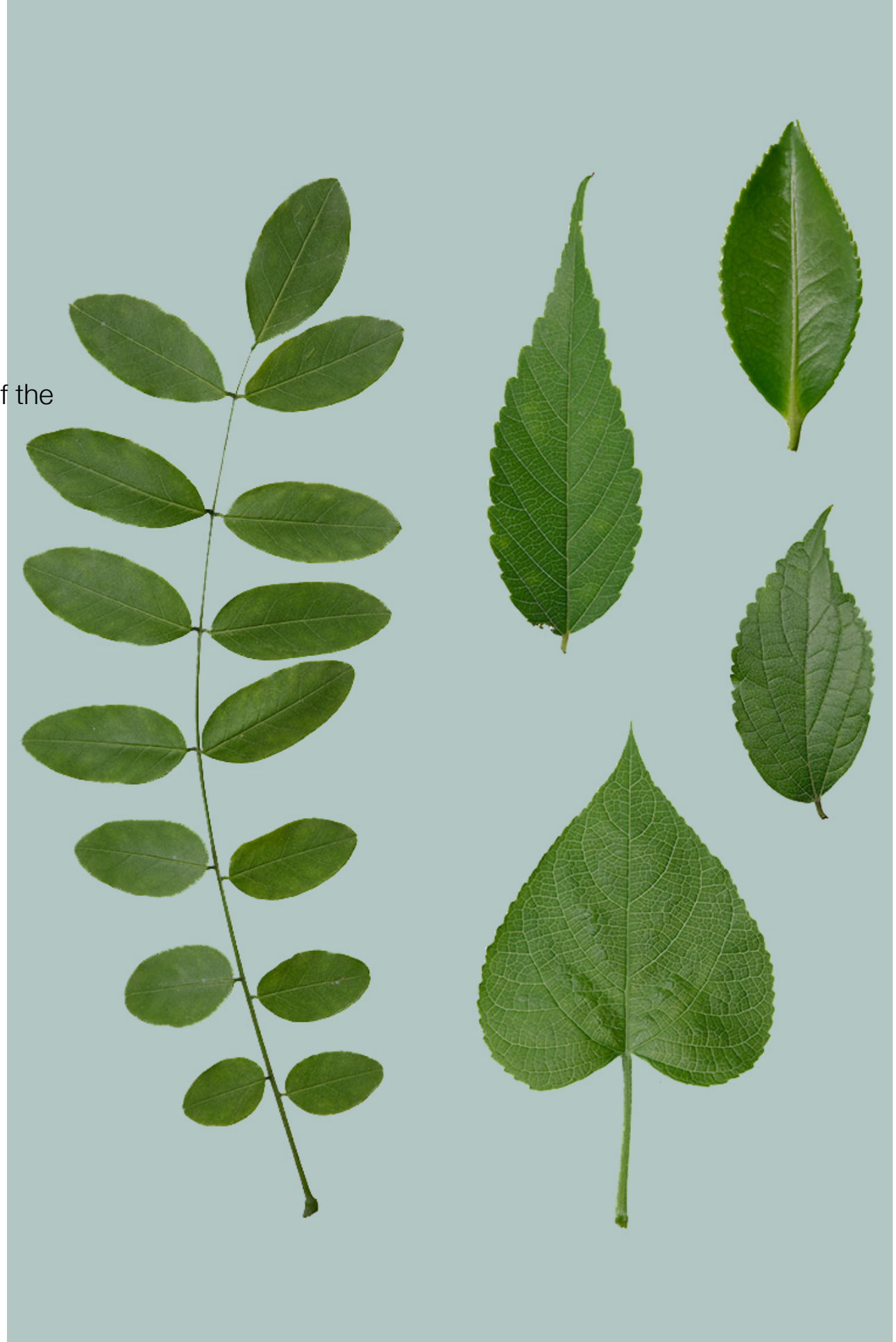
What We Need

Supporting Knowledge Graph

- Ontology layer adding a knowledge domain on top of the physical indexes structure

- Query semantics (cypher) that focus on the domain entities and hide the underlaying complexities

- Logical & physical execution planner that optimise execution on top of the actual data



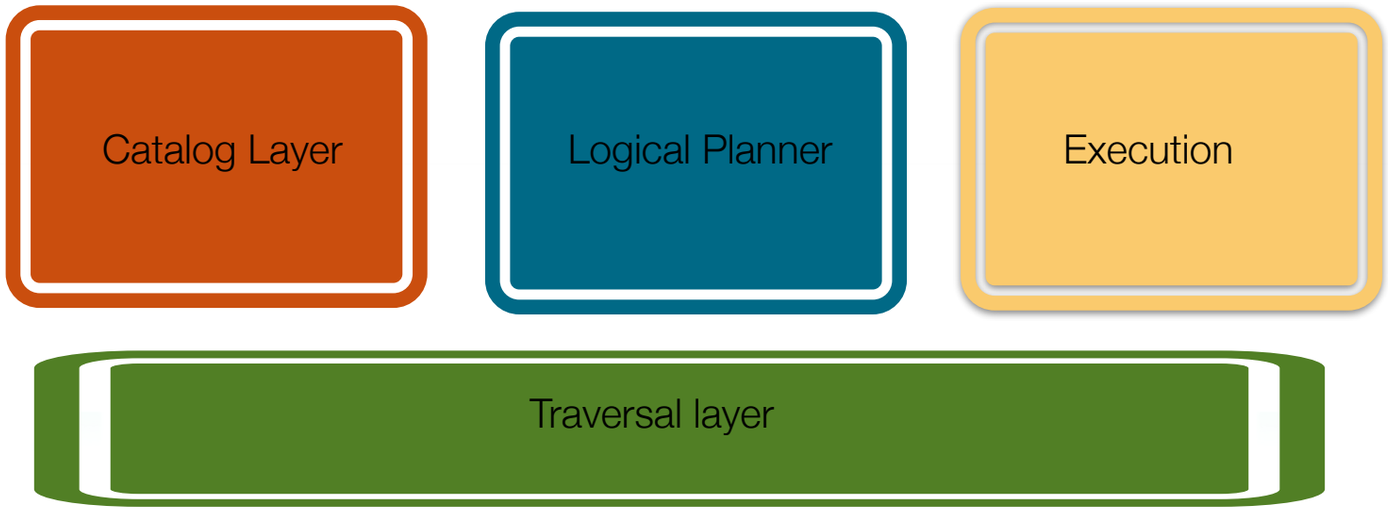
Why is it complicated ?

- Currently focus on documents / indexes
- Indexes can’t be joined / intersected
- Document mapping and the query DSL are highly coupled
- Changing physical mapping mostly needs reindexing
- Needed Parallel Join capabilities for optimised execution

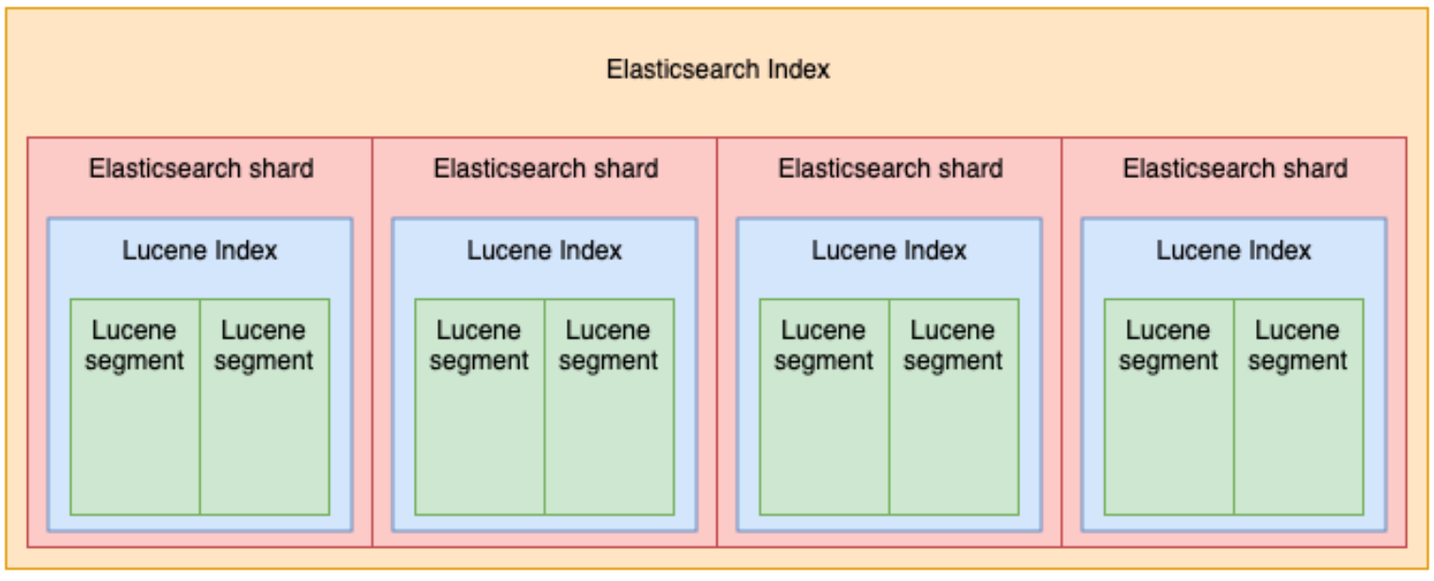
How Its Done...

https://github.com/YANG-DB

{ REST }



Gremlin
$G = (V, E)$



Demo

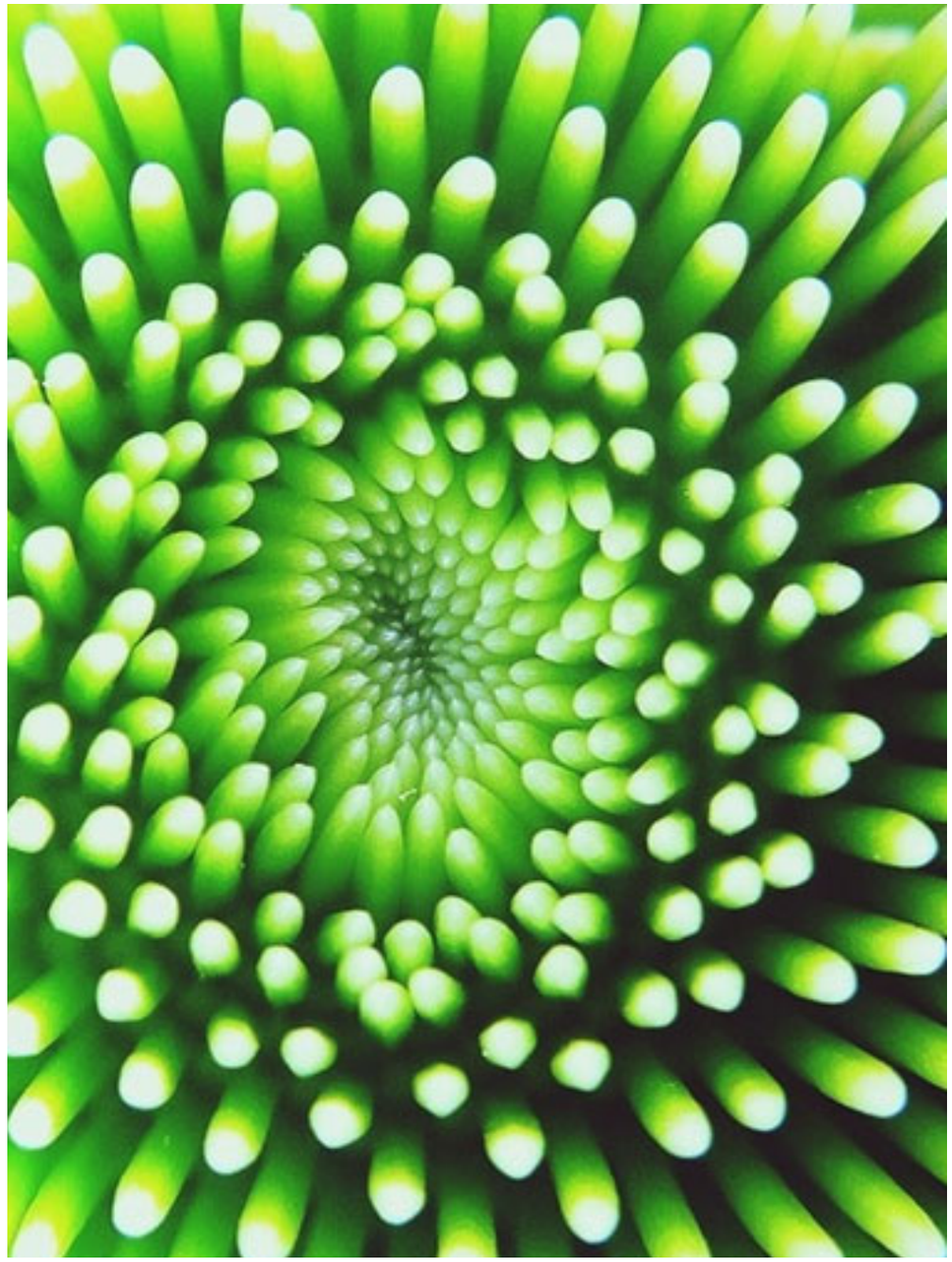
Road map for integration

User Experience
- add ontology (schema) viewer for kibana (ERD like)
- add PGQL query builder capable of auto-complete labels from the graph

schema
- add graph query results viewer for graph type results projections
- add explain/analyse viewer for understanding query execution / performance

Language
- PGQL Parser (ANTLR https://github.com/oracle/pgql-lang/)
- Defining a LPG (Label Property Graph) on top of existing indices
- Creating queries in PGQL
- Catalog different ontologies

Engine
- Converge into a unified codebase
- Supported as an open-search Plugin
- Graph schema catalog
- Cost Based Optimiser for best performance (https://github.com/YANG-DB)
- Distributed execution drivers capable of splitting and gathering the query tasks
- Add benchmark support capabilities



Thank You