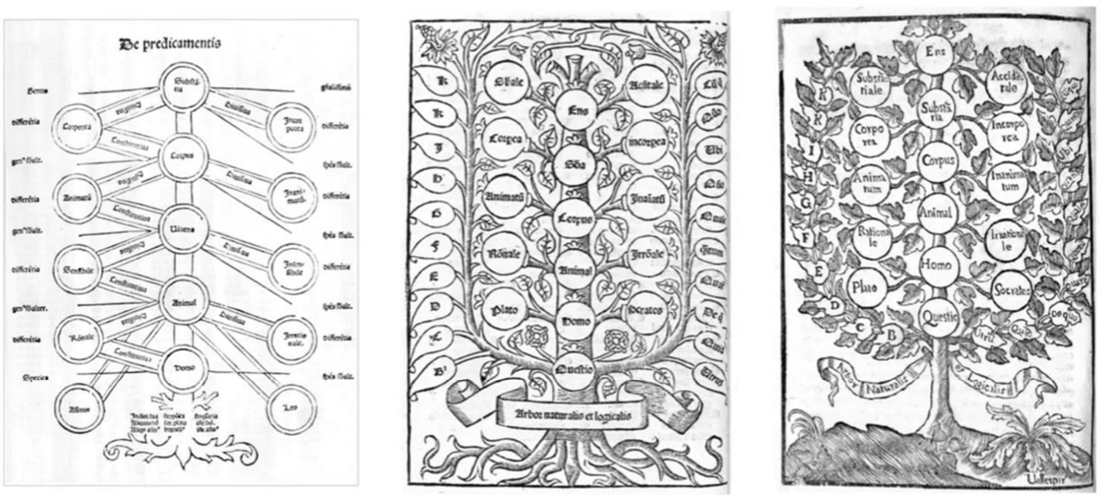
Generative AI Bootcamp

Knowledge Graphs, Graph RAG

Knowledge Graphs, Graph RAG

- Knowledge Graphs
- Graph Database (Neo4j)
- Cypher query language
- Graph RAG with LangChain

Knowledge Graphs have been a key part of mapping data, throughout human history



Porphyrian Tree (oldest Tree of Knowledge)

A visual history of human knowledge

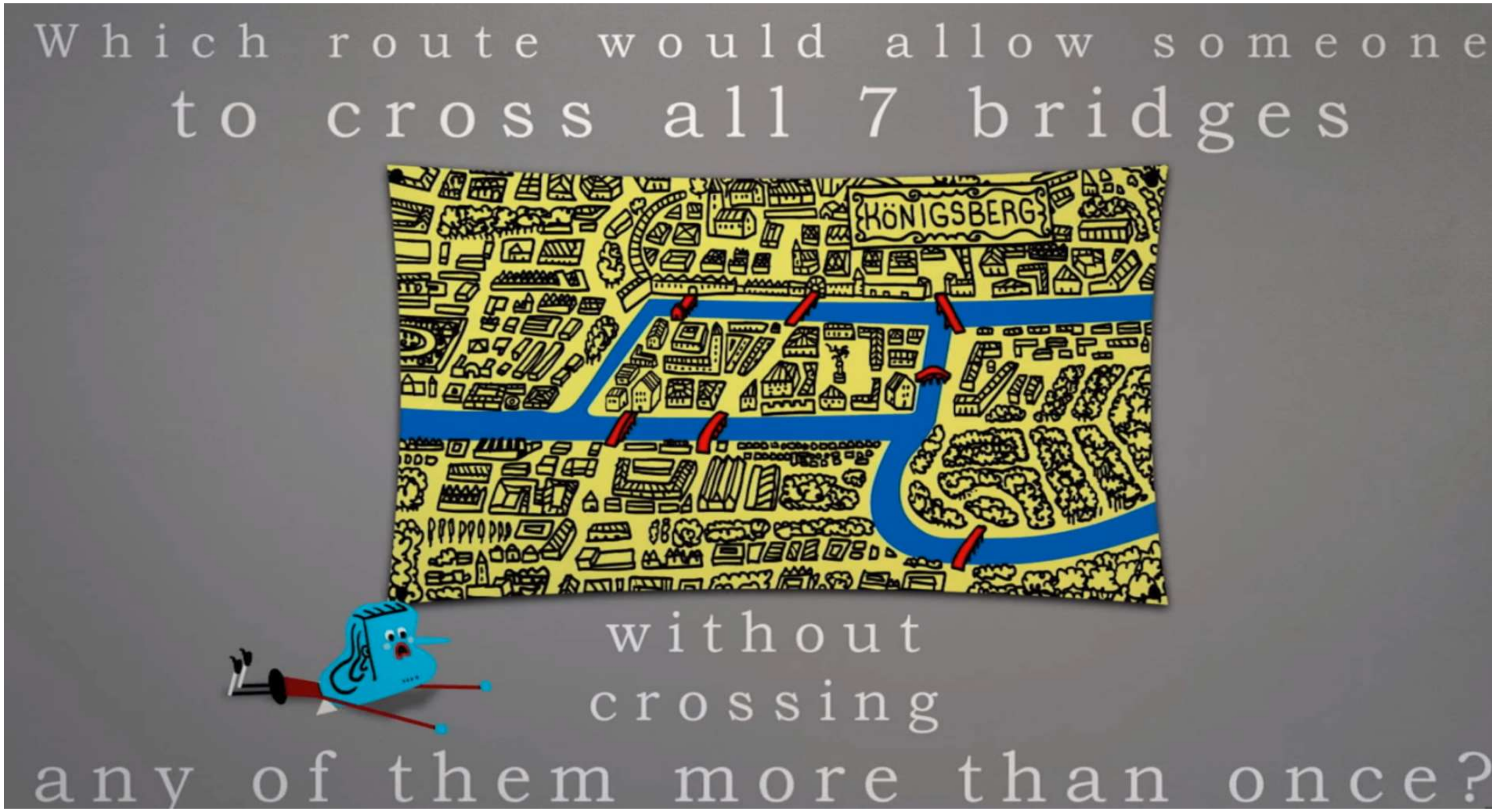
2,077,878 plays | Manuel Lima | TED2015 • March 2015

https://www.youtube.com/watch?v=BQZKs75RMqM

https://www.ted.com/talks/manuel_lima_a_visual_history_of_human_knowledge?language=en

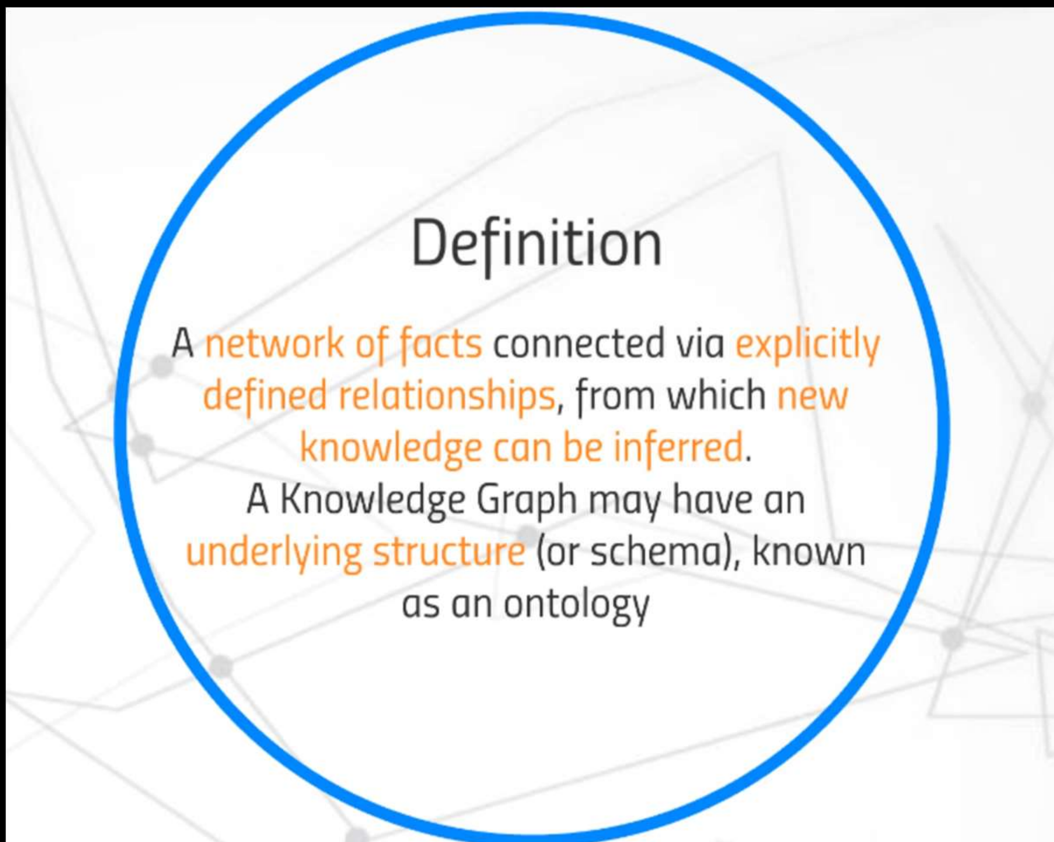
Graph Theory origins

1736 – Leonhard Euler

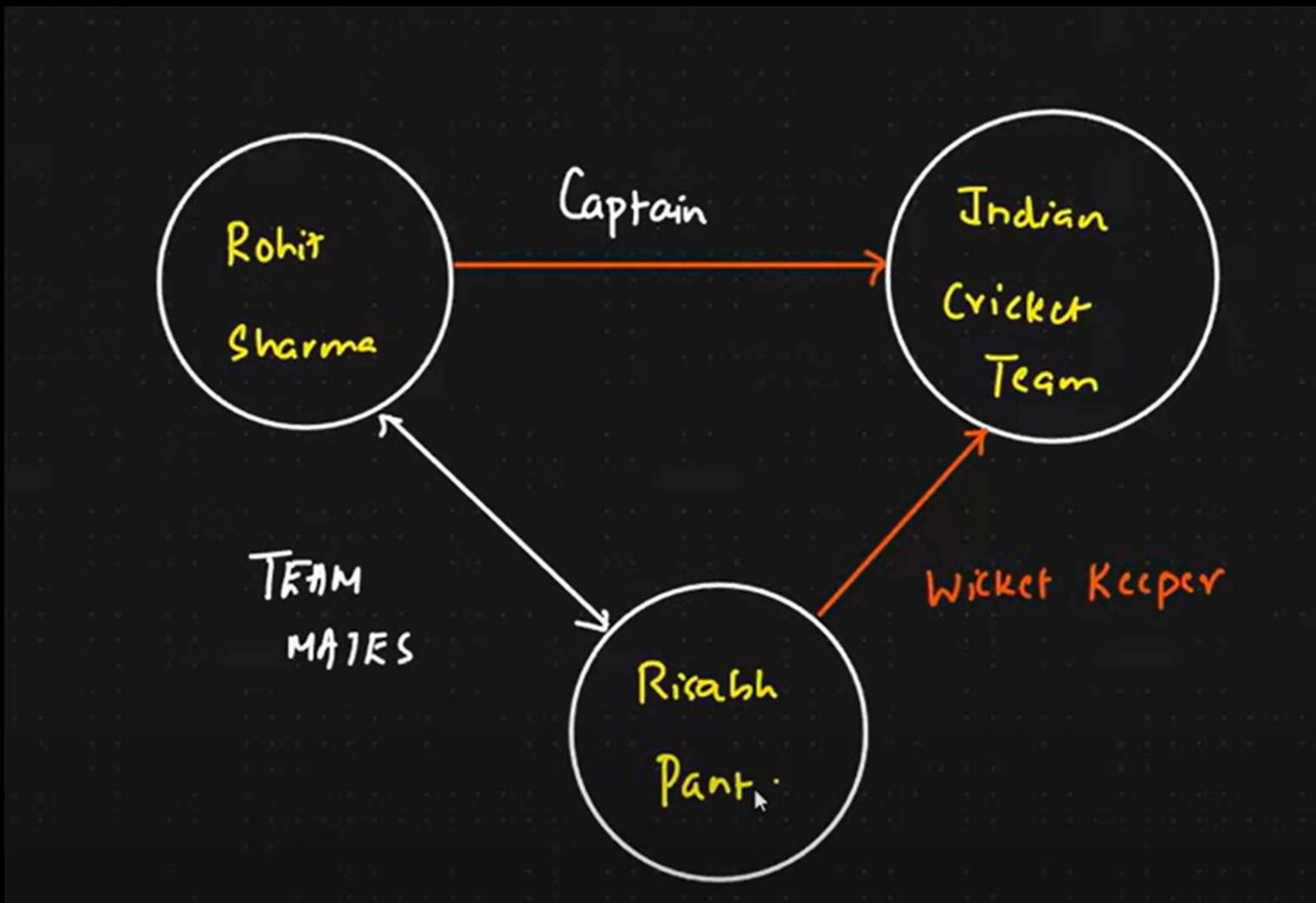


[Seven Bridges of Königsberg – Wikipedia]
[How the Königsberg bridge problem changed mathematics - Dan Van der Vieren]
https://www.youtube.com/watch?v=ycRuO-u6rt8

Knowledge Graph - definition



Knowledge Graph – nodes and relations



Key components of a knowledge graph (a.k.a. KG)

- Nodes
  - Contain labels and properties
  - e.g., PERSON, PLACE
- Relationships (Edges)
  - Directional, e.g., FRIENDS_WITH
- Labels
  - Used to group nodes of a certain type, for e.g., all person nodes can have label Person
- Properties
  - Both nodes and relations can have properties, which are key-value pairs. For e.g. a Person node may have name="Vishal", age="30"
- Relationship Type
  - Type of relationship, foe e.g. WORKS_FOR

KG use-case: Proactive network maintenance and RCA

Scenario

A major mobile operator is experiencing intermittent service degradation in a dense urban area, leading to a spike in customer complaints about slow data speeds and dropped calls, particularly during peak hours.

How Knowledge Graphs Help

Instead of engineers manually sifting through alerts from dozens of siloed monitoring systems (for radio access network, transport, and core network), a knowledge graph provides a unified, real-time view of the entire network topology and its dependencies

KG use-case: Proactive network maintenance and RCA

Specific example

A major mobile operator is experiencing intermittent service degradation in a dense urban area, leading to a spike in customer complaints about slow data speeds and dropped calls, particularly during peak hours.

The knowledge graph can instantly visualize the relationships between

- a specific cell tower (Cell_ID_123)
- the services running on it (e.g., 5G_NSA, VoLTE)
- the connected user devices,
- and the underlying transport network.

When an anomaly is detected, the graph can trace the impact upstream and downstream. For instance, it might reveal that a specific router port (Router_XYZ, Port_GigaEthernet0/1) is experiencing high packet loss, which is impacting not just one, but three different cell sites.

This allows engineers to pinpoint the root cause in minutes rather than hours, leading to faster resolution and preventing a wider outage.

KG use-case: Supply Chain disruption analysis

Scenario:

Telecom Company: Ericsson (Sweden)

Chip Supplier: Broadcom Inc. (USA)

Chip Manufacturing Sites: Taiwan, Vietnam, Malaysia

SITUATION:

Ho-chi-minh is hit with sever flooding resulting in stoppage of chip production

KG use-case: Supply Chain disruption analysis

🤔 Ericsson's Question:

- *"How will the flood in Vietnam impact our chip deliveries from Broadcom?"*
- *"Do we have alternate sources or buffer capacity?"*
- *"Which of our final products will be delayed?"*

KG to the rescue

```
Ericsson → sources chip → Broadcom
Broadcom → outsources packaging → ASE Group
ASE Group → has site → Ho Chi Minh City, Vietnam
Ho Chi Minh → affected by flood → Severe Disruption
Chip X → used in → Ericsson 5G Router Y
```

Popular graph databases

Name	Strengths	Weaknesses	Licence / Deployment model	Query-language support
Neo4j (Neo4j Inc)	• Mature ecosystem, rich tooling (APOC, Bloom, drivers) • Powerful graph algorithms & GDS library • Strong community & training resources	• Some features in enterprise-only edition (clustering, RBAC) • High memory footprint for very large graphs • GPL v3 for Community can deter closed-source redistribution	Community Edition — GPL v3; Enterprise — commercial subscription (Graph Database & Analytics)	Native Cypher (plus preview ISO GQL in v5.x)
Amazon Neptune	• Fully-managed, auto-patched & backed-up • Choice of property-graph and RDF models • Deep integration with the AWS stack	• Vendor lock-in to AWS • Separate endpoints for PG/RDF • Pricing driven by instance size + I/O	Closed-source, AWS-managed PaaS (AWS Documentation)	openCypher, Gremlin, SPARQL (AWS Documentation)
ArangoDB (ArangoDB Inc)	• Multi-model (documents + KV + graphs) in one engine • Flexible JOIN-like traversals via AQL • Good built-in clustering & sharding	• Smaller graph-specific ecosystem • Licence changed to BSL 1.1 for future versions (cloud-hosting restriction)	Apache 2.0 ≤ v3.11; BSL 1.1 (falls back to Apache 2.0 after 4 yrs) from v3.12+ (ArangoDB, ArangoDB)	Native AQL

Popular graph databases (contd.)

Name	Strengths	Weaknesses	Licence / Deployment model	Query-language support
TigerGraph (TigerGraph Inc)	• High-performance distributed engine tuned for real-time analytics • Parallel, Turing-complete **GSQL** language • Built-in graph data-science and vector search	• Proprietary closed source (free dev edition only) • Learning curve vs SQL/Cypher • Smaller community than Neo4j	Proprietary, node-locked licence; free single-node Dev edition (TigerGraph Documentation)	GSQL (plus optional openCypher subset) (TigerGraph Documentation)
JanusGraph (Linux Foundation)	• 100 % open-source, vendor-neutral • Back-end pluggable (Cassandra, HBase, Bigtable, etc.) • Massively scalable with external storage & index layers	• Requires external DB + search engine (ES/Solr/Lucene) • Operational complexity vs all-in-one stores • No native query DSL beyond Gremlin	Apache 2.0 (JanusGraph)	Gremlin (TinkerPop); SPARQL via plugin (JanusGraph)
Azure Cosmos DB (Gremlin API)	• Globally distributed with multi-region writes • Millisecond P99 latency SLA • Multi-model (graph, docs, KV, column) under one service	• RU/s cost model can be hard to predict • Gremlin feature set lags behind on-prem engines • Closed-source cloud service	Proprietary, fully-managed PaaS by Microsoft Azure (Microsoft Azure)	Gremlin API (plus SQL, Mongo, Table, Cassandra APIs) (Microsoft Azure)

Example – nodes, relations, properties creation using Cypher

CREATE (p:Person {name: "Alice", age: 30, city: "New York", email: "alice@example.com"})

CREATE (c:Company {name: "Acme Corp", location: "San Francisco", industry: "Technology"})

CREATE (p)-[r:WORKS_FOR {since: 2015, role: "Software Engineer", department: "R&D"}]->(c)

CREATE (p)-[r:FAN_OF]->(c)

CREATE (p:Person {name: “Bob", age: 30, city: "New York", email: "alice@example.com"})

CREATE (p)-[r:WORKS_FOR {since: 2015, role: "Software Engineer", department: "R&D"}]->(c)

- How many nodes?
- How many Node Labels?
- How many relationship types?
- How many relations?

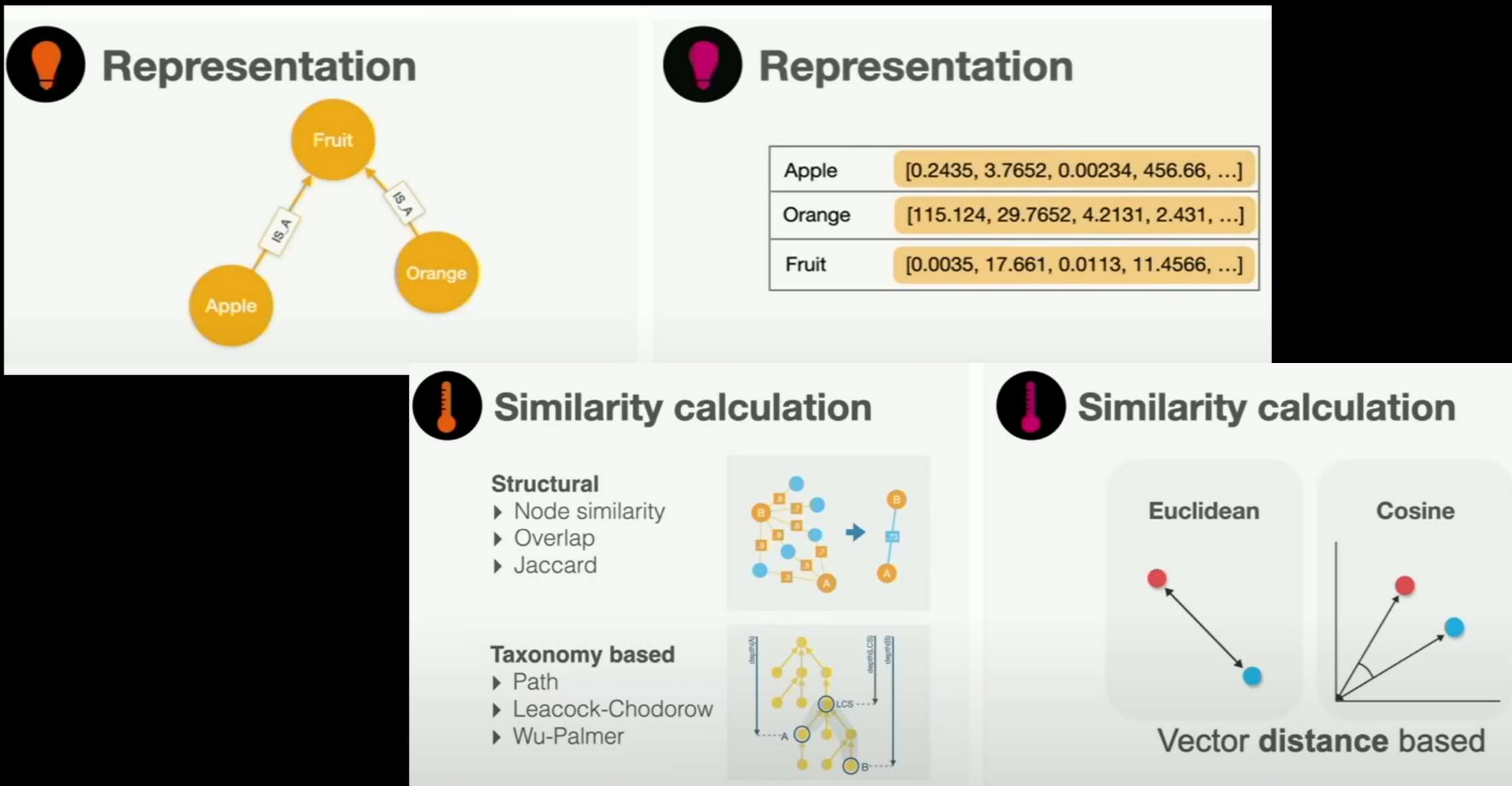
- Nodes: 3, Node labels: 2 (Person, Company)

Language families used by popular graph databases

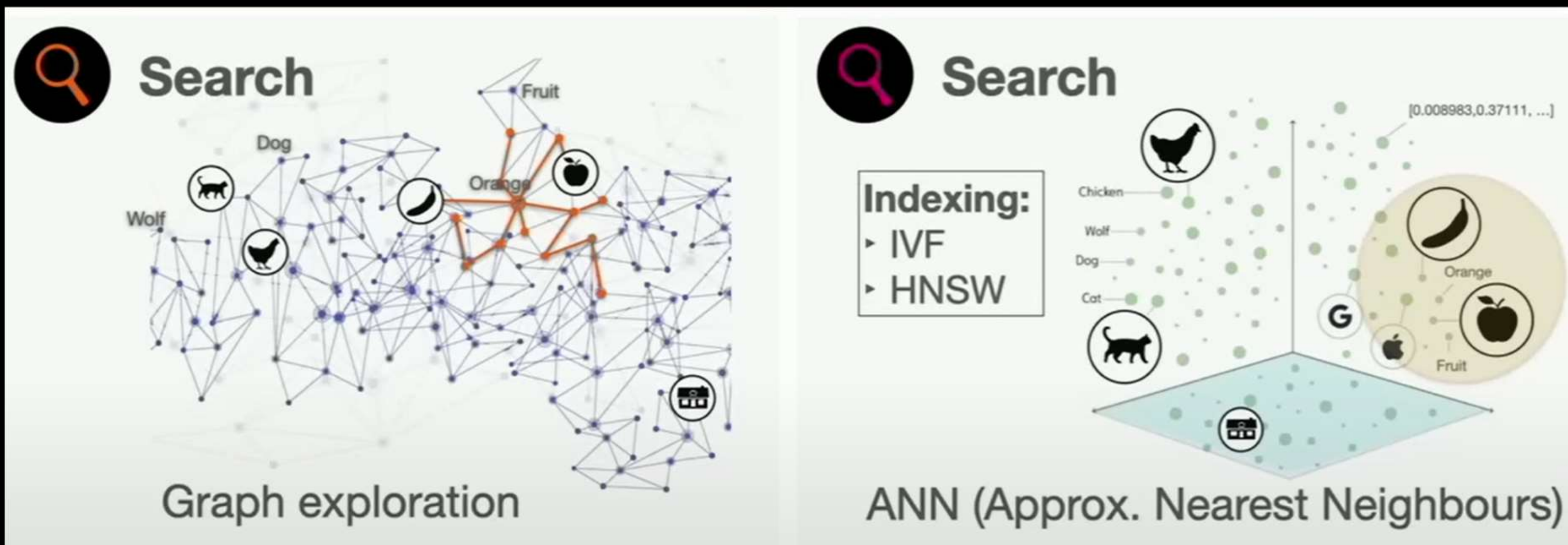
Language family	Typical database engines[1]	Notes
openCypher / Cypher	Neo4j, RedisGraph, Memgraph, SAP HANA Graph, Amazon Neptune (preview), NebulaGraph, Apache AGE	De-facto “property-graph” lingua franca, but implementation details diverge.
Gremlin (Apache TinkerPop)	JanusGraph, Azure Cosmos DB (Gremlin API), Amazon Neptune, OrientDB, DataStax Enterprise Graph	Imperative traversal style; vendor-neutral framework.
SPARQL (W3C standard)	GraphDB, Blazegraph, Virtuoso, Stardog, Neptune (RDF mode)	Targets **RDF** triple stores rather than property graphs.
GSQL	TigerGraph	Declarative, graph-native analytics extensions.
AQL	ArangoDB	Multi-model (documents + graphs) query language.
PGQL	Oracle PGX, Oracle Database 23c	SQL-like, one of the inputs to ISO GQL.
SQL/PGQ (part 16 of SQL:2023)	Rolling into mainstream relational DBs (e.g., PostgreSQL prototypes, Oracle 23c)	Lets you run property-graph pattern matching *inside* SQL. Peter Eisentraut ISO

Demo with neo4j

KG Vs regular RAG

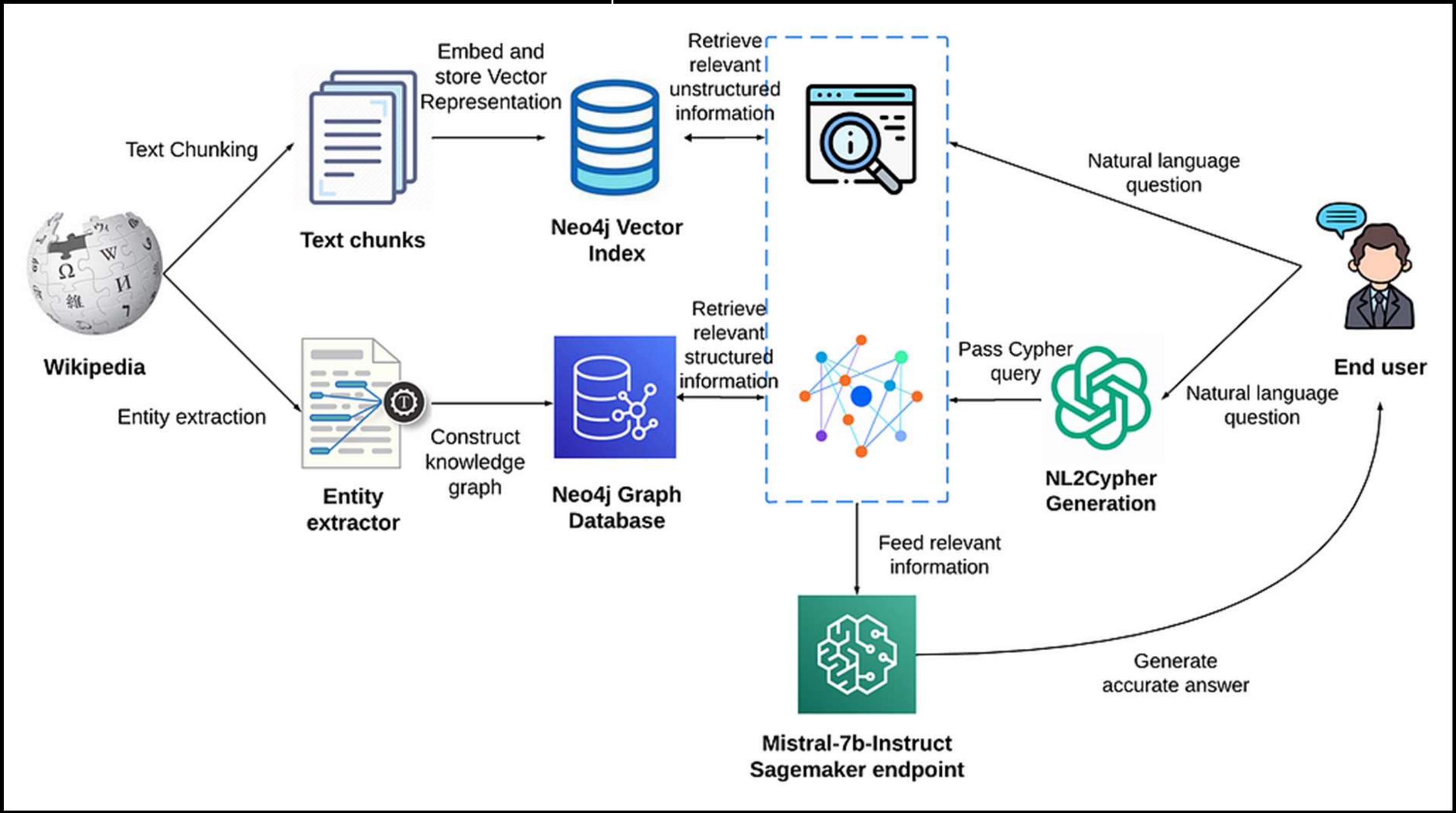


KG Vs regular RAG



Managing unstructured text and KG in neo4j

Indexing | Retrieval



Integrated graph retrieval system

Initial Filtering (Keyword search)

A quick keyword search to narrow down a large corpus to a manageable set of candidates.
Such as metadata filtering

Semantic search and ranking

Apply embeddings and cosine similarity to rank the candidates based on their semantic relevance.

Relationship Analysis

Finally, incorporating graph relationships to refine or contextualize the search results, such as ranking based on node connectivity or importance.