



An Introduction to Graph Databases with GQLAlchemy and Python





This is Katarina

- Born at the Croatian coast, in the town of Šibenik
- Master's in Mathematics and Computer science from the Faculty of Science in Zagreb
- I work as a Developer Relations Engineer at Memgraph
- I love to travel, cook and eat tasty food (check out my Instagram)



The graph data model

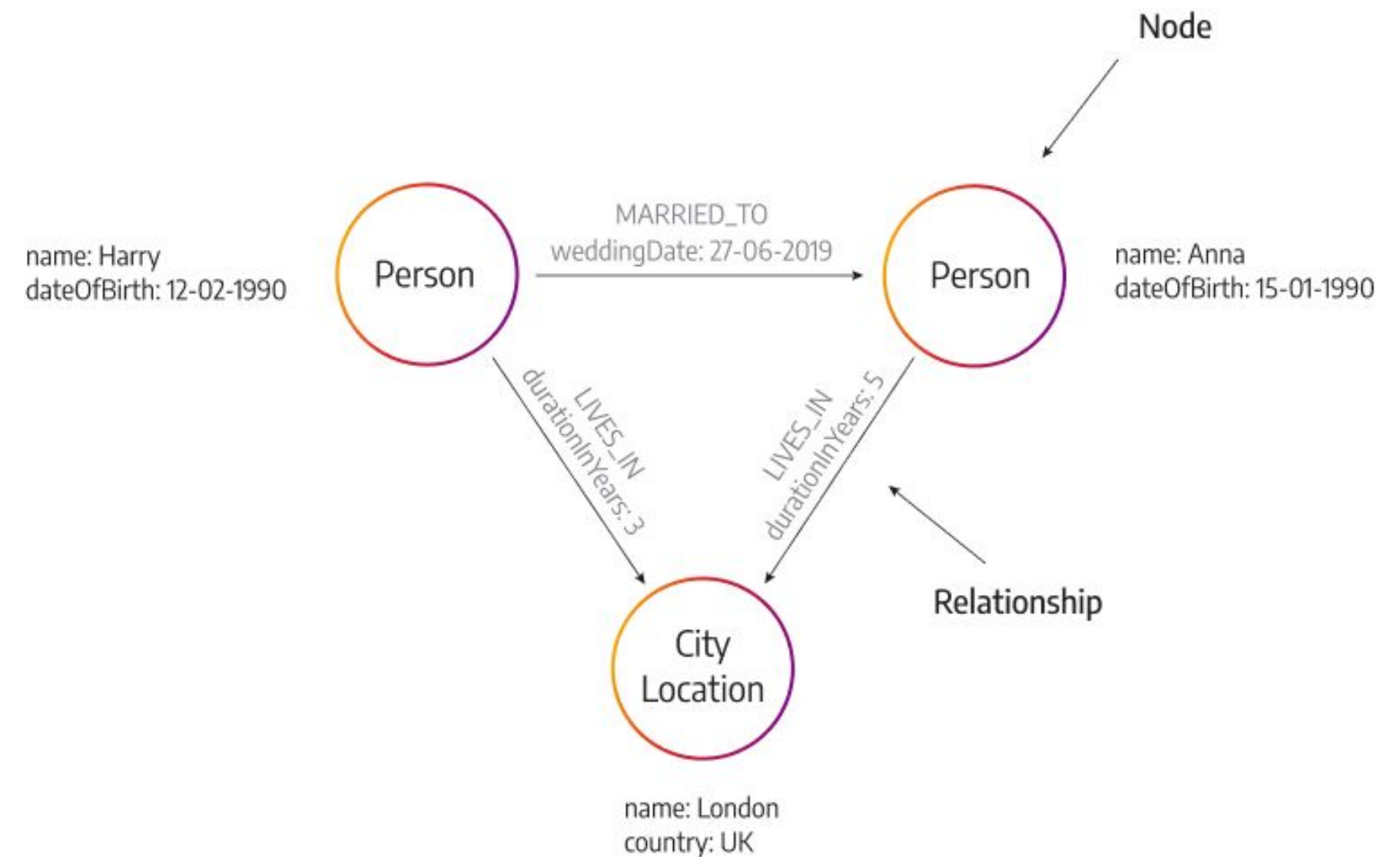
What are graphs and how to model a graph database?



What are graphs?

A graph is a network structure that consists of a set of nodes (vertices) and a set of relationships (edges) connecting them.

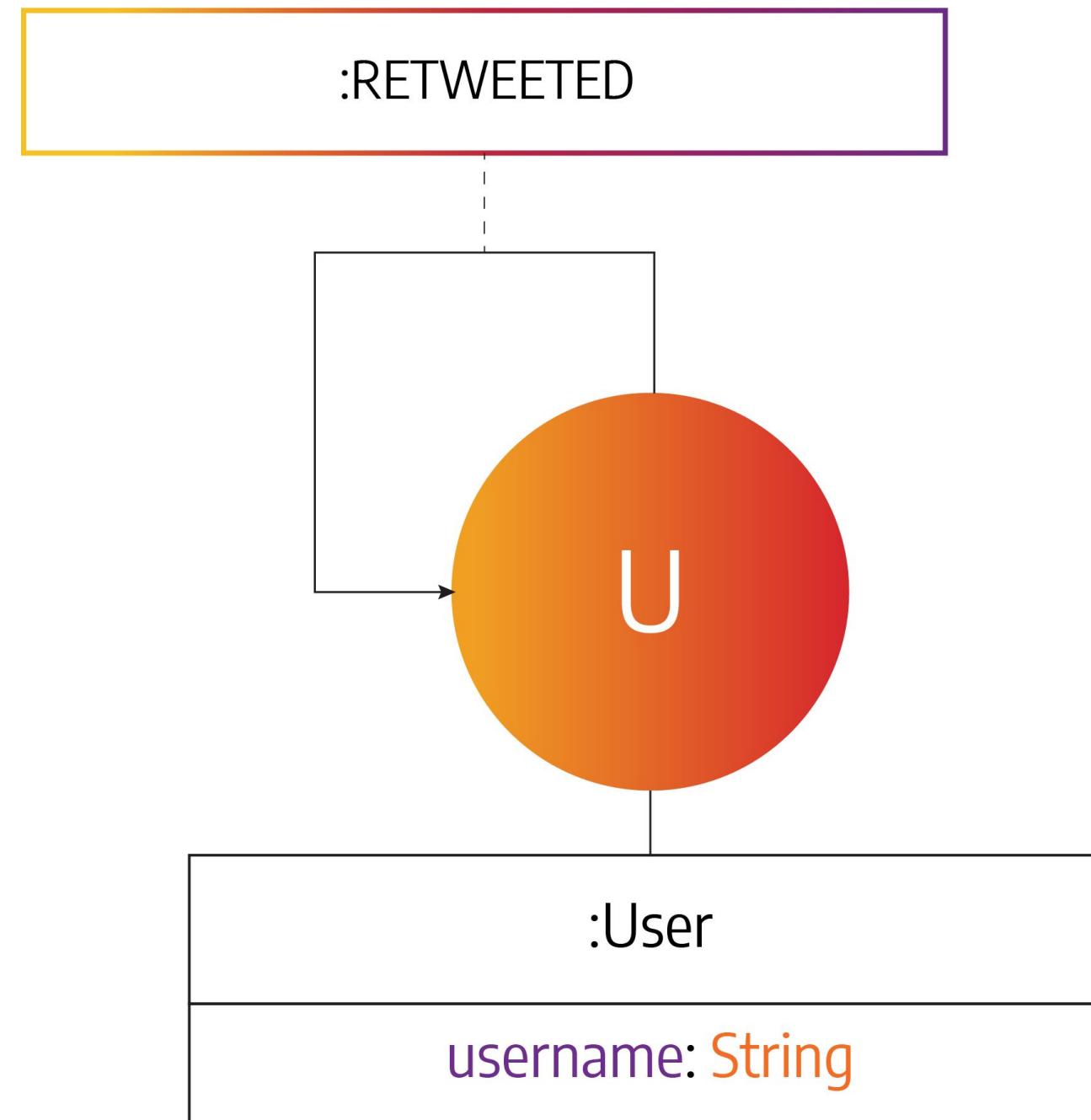
- **Nodes** - structures that represent entities
- **Relationships** - connections between these entities
- **Properties** - associated values (key-value pairs) belonging to either nodes or relationships



Labeled property graph model



The Twitter graph data model





Graph vs relational database

How does a graph database differ from a relational database?

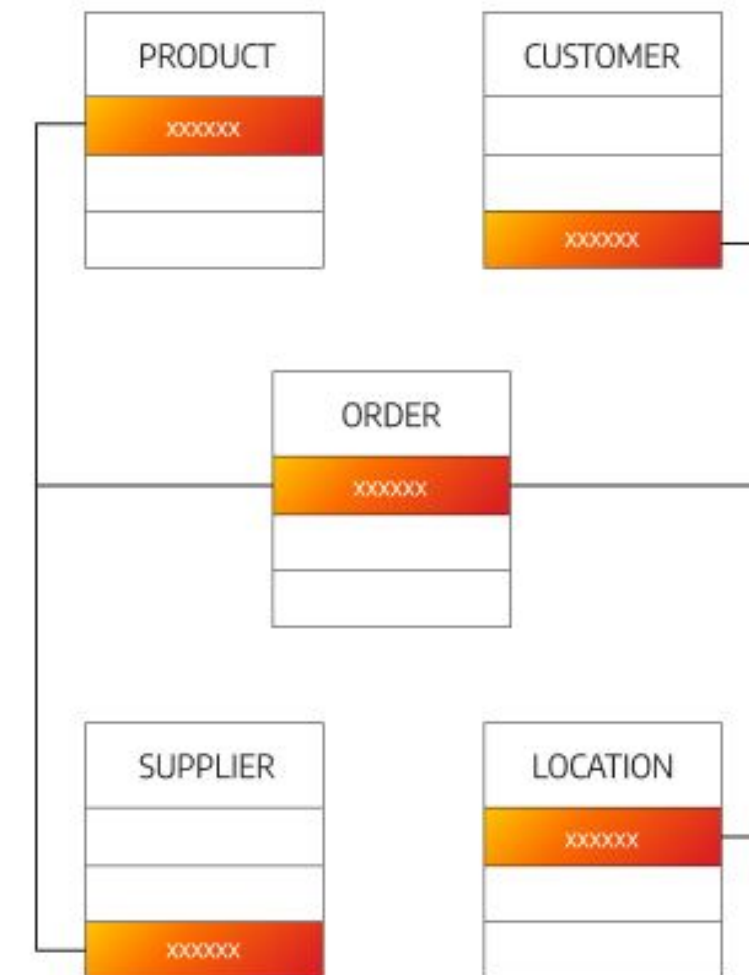


Graph database vs relational database

GRAPH DB



RELATIONAL DB





Cypher query language

Cypher is the most widely adopted, fully-specified, and open query language for property graph databases. It provides an intuitive way to work with property graphs.

Cypher contains:

- clauses such as `MATCH`, `DELETE`, `SET`, `RETURN`...
- functions such as `round()`, `cos()`, `toString()`...
- custom procedures written in Python, C/C++ and Rust



Cypher query language

```
MATCH (u:Customer{customer_id:'customer-one'})-[:BOUGHT]->
(p:Product)<-[:BOUGHT]-(peer:Customer)-[:BOUGHT]->
(reco:Product)
WHERE not (u)-[:BOUGHT]->(reco)
RETURN reco as Recommendation, count(*) as Frequency ORDER
BY Frequency DESC LIMIT 5;
```

SQL

```
SELECT product.product_name as Recommendation, count(1) as
Frequency
FROM product, customer_product_mapping, (SELECT
cpm3.product_id, cpm3.customer_id
FROM Customer_product_mapping cpm,
Customer_product_mapping cpm2, Customer_product_mapping
cpm3
WHERE cpm.customer_id = 'customer-one'
and cpm.product_id = cpm2.product_id
and cpm2.customer_id != 'customer-one'
and cpm3.customer_id = cpm2.customer_id
and cpm3.product_id not in (select distinct product_id
FROM Customer_product_mapping cpm
WHERE cpm.customer_id = 'customer-one')
) recommended_products
WHERE customer_product_mapping.product_id =
product.product_id
and customer_product_mapping.product_id in
recommended_products.product_id
and customer_product_mapping.customer_id =
recommended_products.customer_id
GROUP BY product.product_name
ORDER BY Frequency desc
```

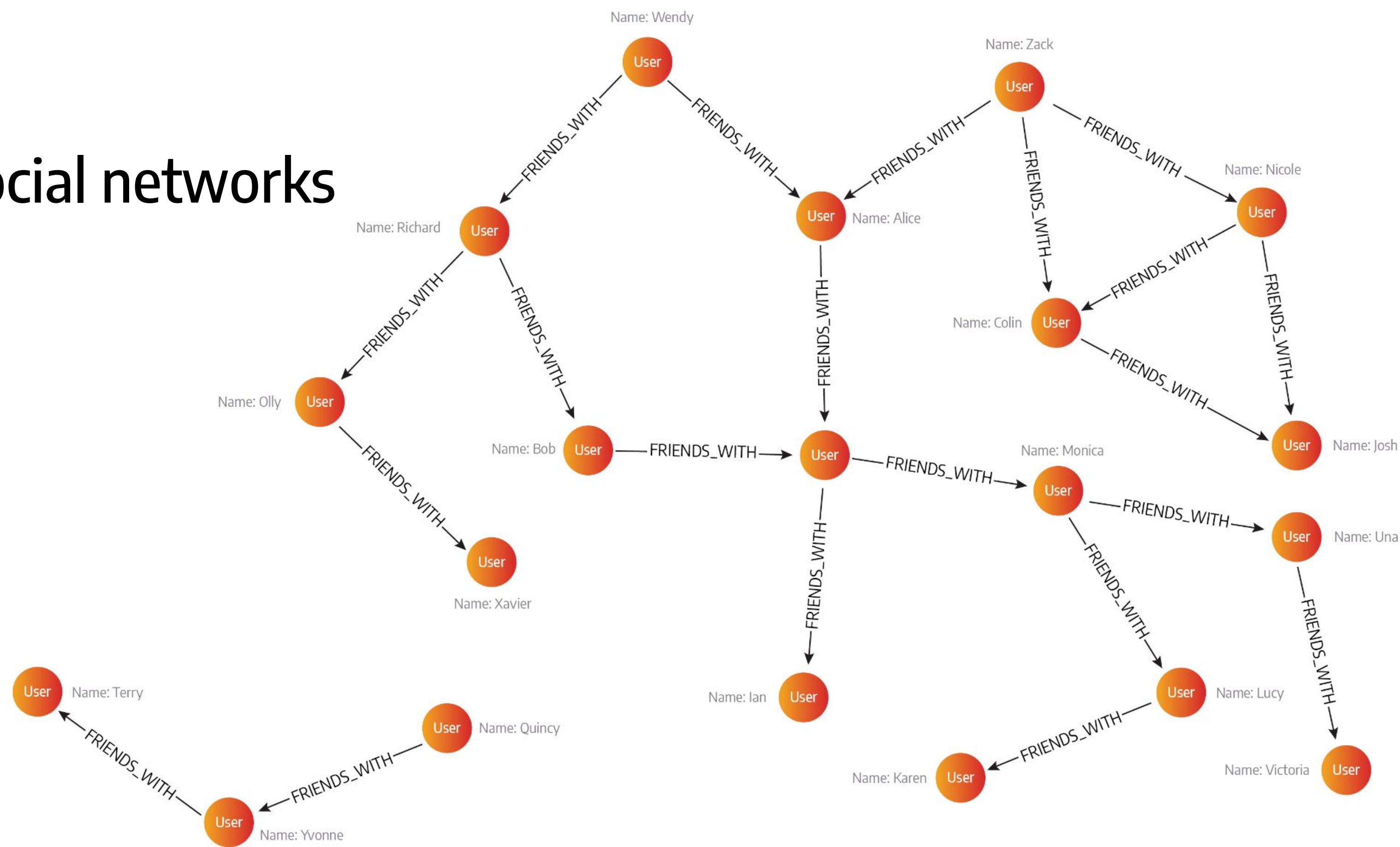


Graph database use cases

When to use a graph database?

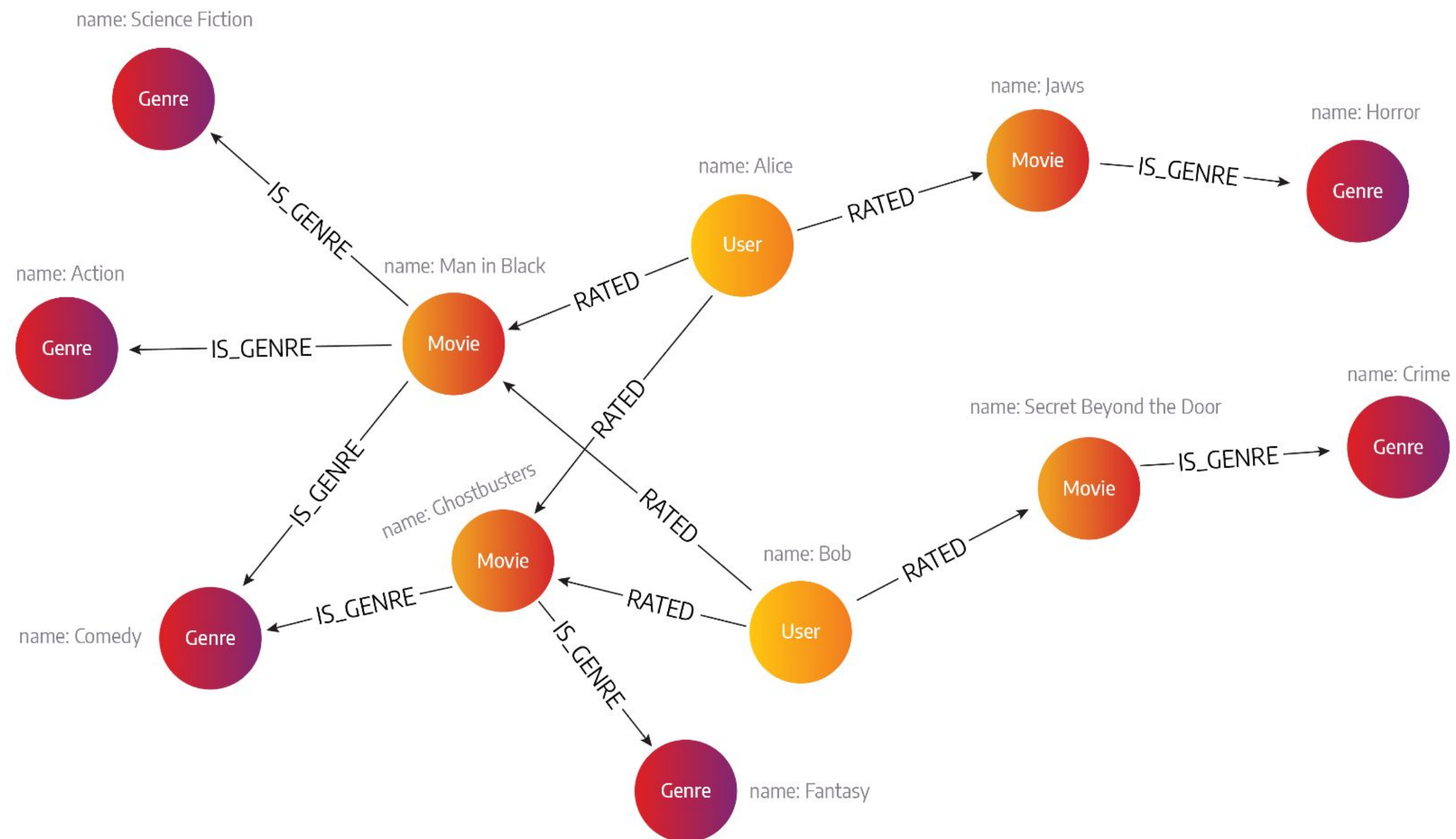


Social networks



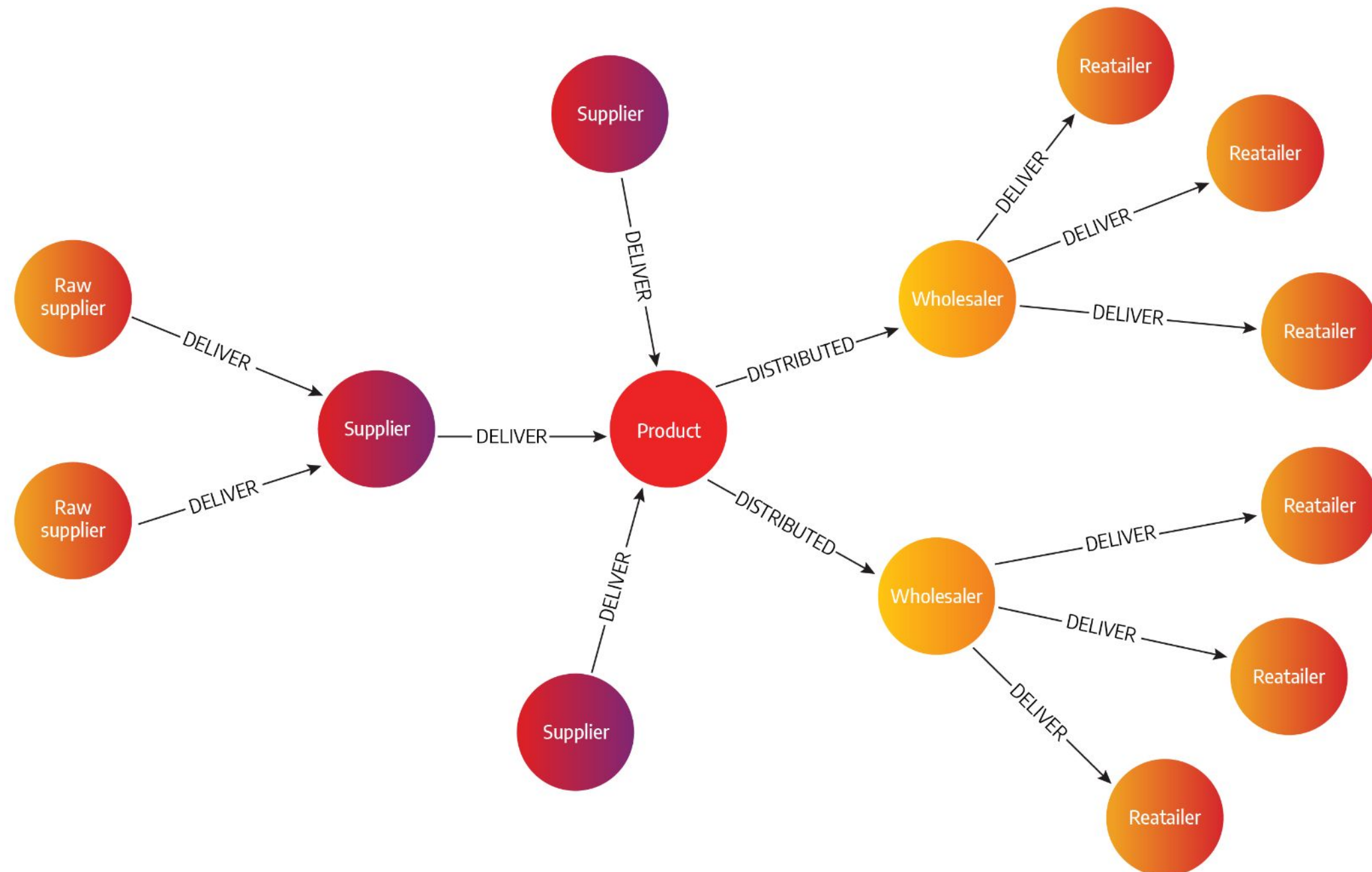


Recommendation engines



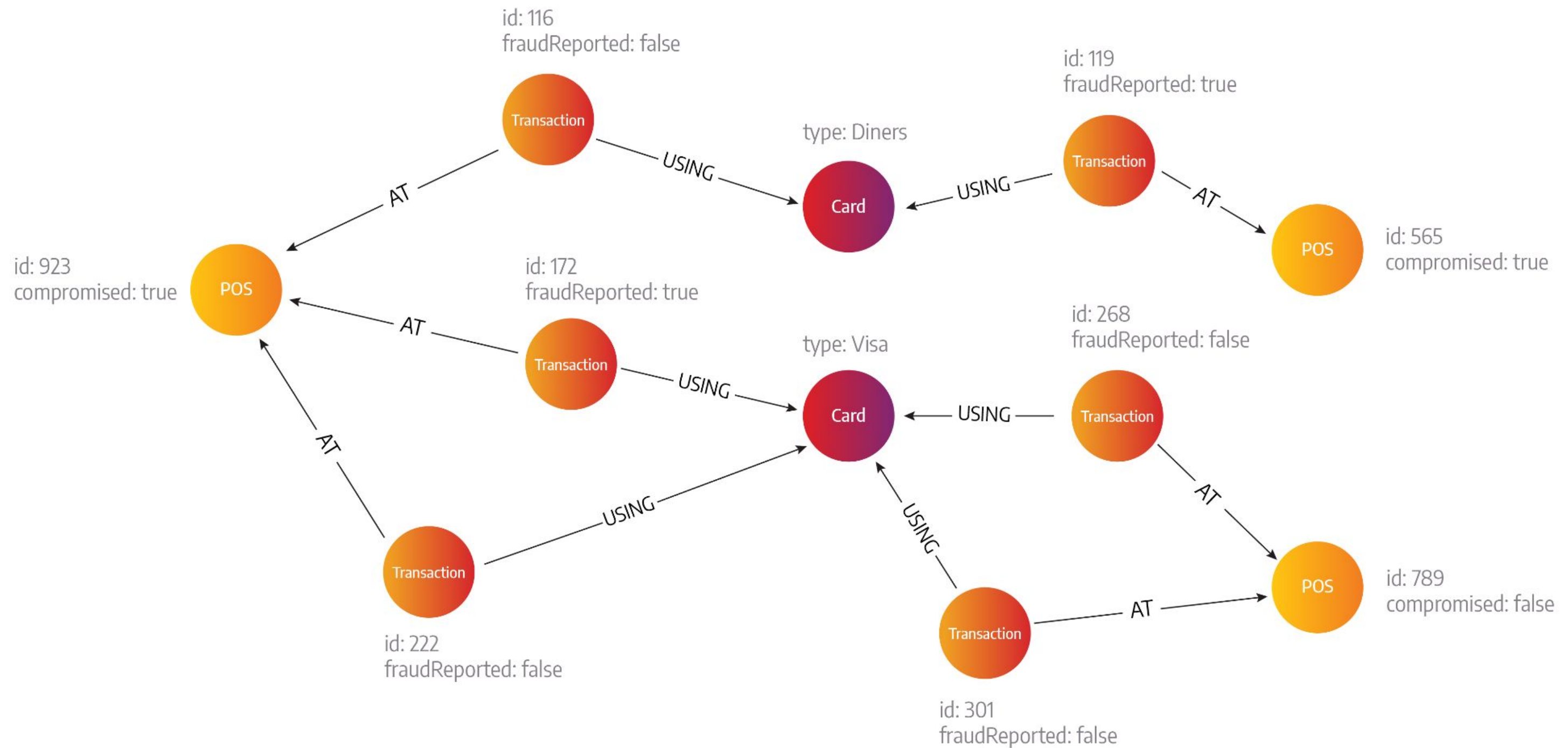


Supply chain management





Fraud detection







Memgraph Ecosystem

What is Memgraph?



Memgraph

Memgraph is a platform for **graph computation on streaming data** powered by an in-memory graph database.

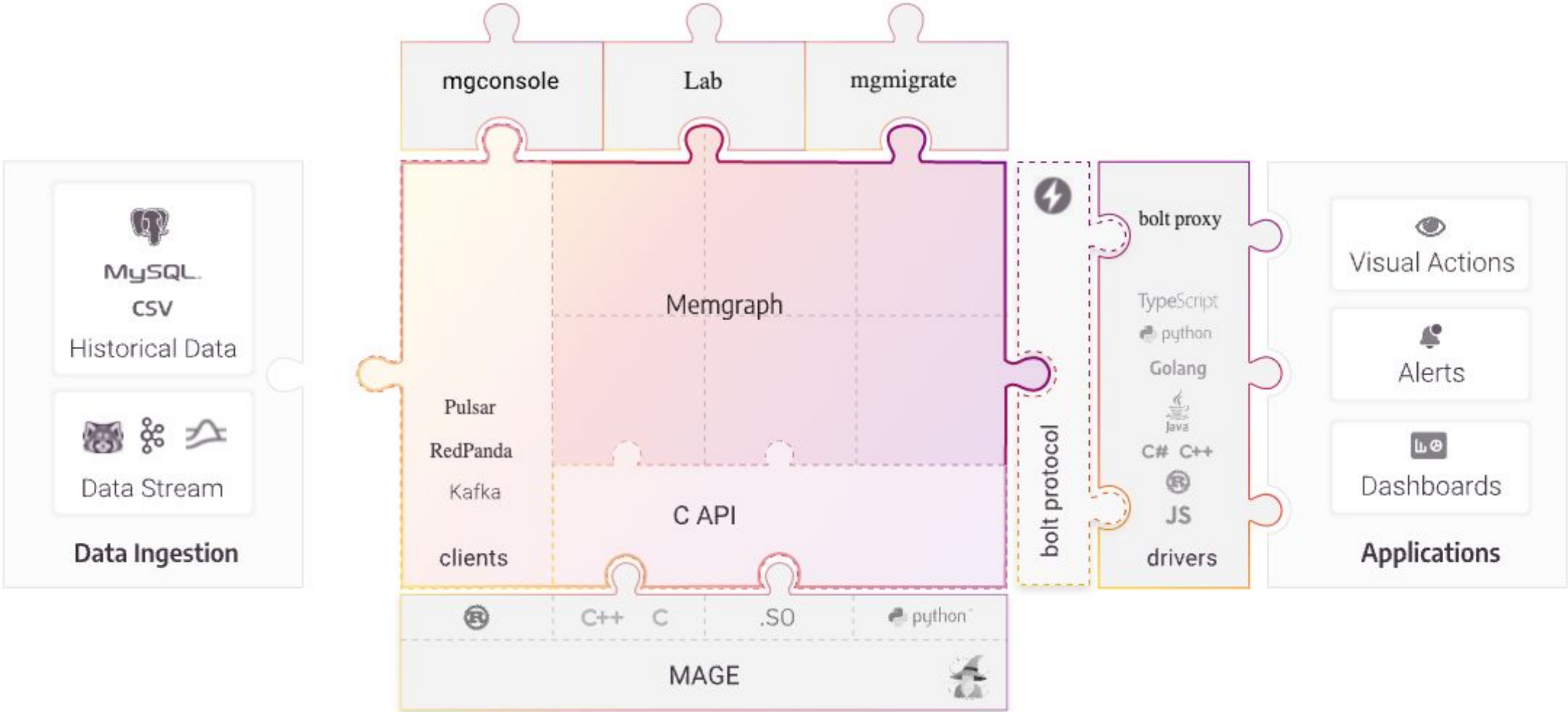
Memgraph is used to:

- Store graph data in memory
- Run graph analytics
- Analyze streaming data





Memgraph Ecosystem





GQLAlchemy

GQLAlchemy is a fully open-source **Python library**. It is an Object Graph Mapper (OGM) - a link between Graph Database objects and Python objects.

GQLAlchemy includes:

- OGM capabilities
- Query builder
- On-disk storage
- Graph schema validation

GQLAlchemy 1.1

A Python OGM (Object Graph Mapper) for graph databases



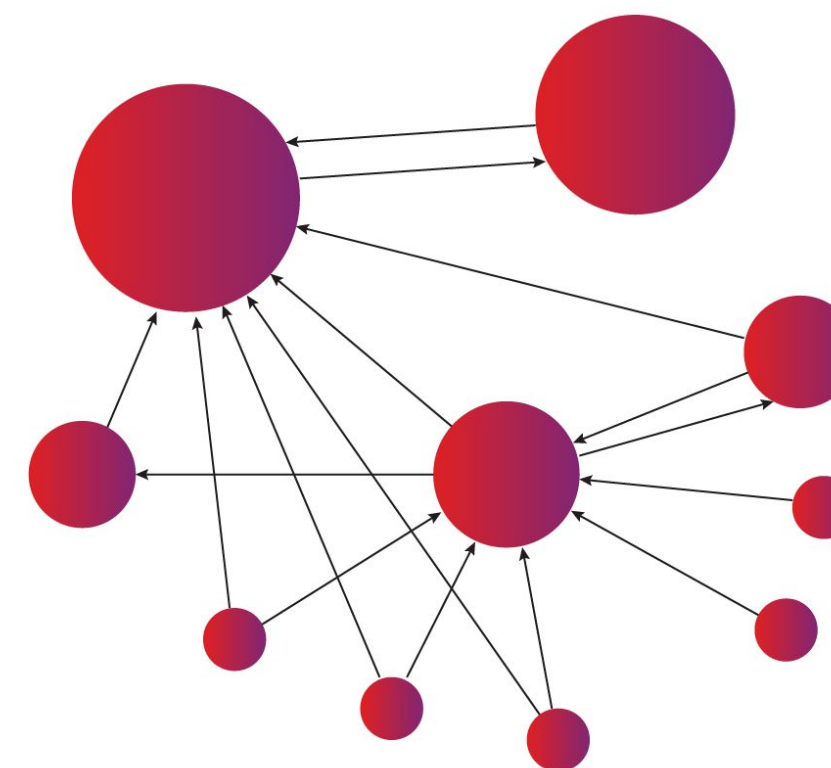
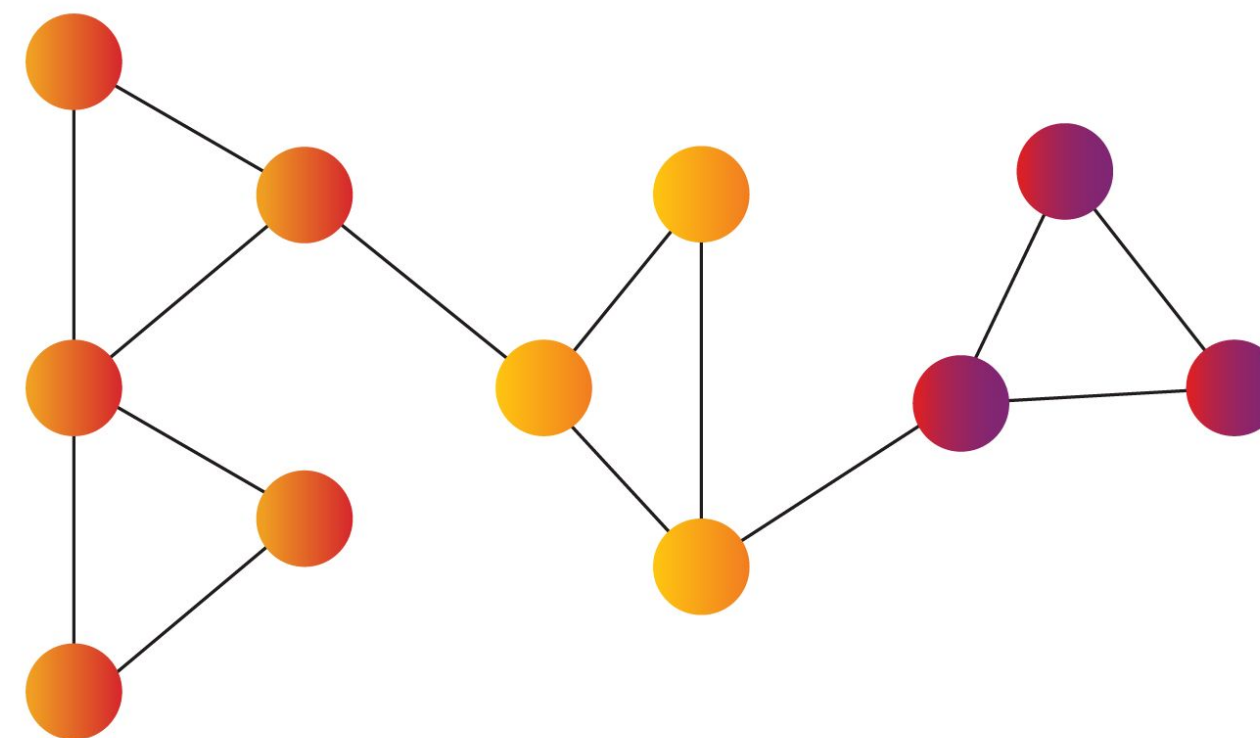


Graph analytics

Graph analytics, also called network analysis, generates insights hidden in the relationships of the network structure.

Some common algorithms include:

- Clustering & community detection
- Connected components
- PageRank
- Shortest path
- BFS & DFS





MAGE (Memgraph Advanced Graph Extensions)

- Open-source repository containing all available user-defined graph analytics modules and procedures
- Extends Cypher query language
- Implements popular graph algorithms such as PageRank, betweenness centrality, community detection, etc.
- Besides traditional graph algorithms, it also implements dynamic graph algorithms





Graph analytics in action



Google was built on the **PageRank** algorithm measuring the importance of web pages.



Facebook's social graph uses **Community Detection** to infer unknown data about their users based on similar network behavior of other users to power their ad-targeting engine.



Amazon uses **Collaborative Filtering** to deliver high quality real-time product recommendations.



Pinterest uses **Random Walks** and **Graph-Machine Learning** to deliver high-quality personalized recommendations responsible for more than 80% of all user engagements.



"Graph-Machine Learning features proved the most valuable of all other features when determining the quality and relevancy of our dish and restaurant recommendations."



Let's do some coding!



What we've learned?

- We use **graph database** when the data is highly connected and when we have lots of many-to-many relationships.
- Memgraph is a platform for **graph computation on streaming data** powered by an in-memory graph database.
- Before importing the data, we first have to create a **graph model**.
- The easiest way to import data is using the **LOAD CSV** Cypher clause.
- For Python developers, it's best to use **GQLAlchemy** to query Memgraph.
- If we want to get some valuable insights from the data, we can perform different kinds of **graph algorithms**, such as PageRank, betweenness centrality, etc.



Thank you for your attention!



If you like what we do,
throw us a star! ★

<https://github.com/memgraph/memgraph>

GQLAlchemy is an open-source library!
Feel free to contribute! 🧪

<https://github.com/memgraph/gqlalchemy>



www.memgraph.io



Join our Discord server!

memgr.ph/discord



www.memgraph.io

 [supe_katarina](https://twitter.com/supe_katarina)

 katarina.supe@memgraph.io