graphsummit

DUBLIN

neo4j

Building a Graph Solution

Using a real world digital twin data set

Marco De Luca
& Gal Bello

Neo4j Field Engineering Team

Goal of the Today

Demonstrate, with YOUR HELP :-), how to build a Graph Solution using a Digital Twin Dataset

Agenda

1. Logistics
2. Introduction
3. Use Case Explanation
4. Building the solution

BREAK (15min)

5. Discussion how to improve the Use Case further
6. Final Q&A
7. Done

Logistics

First things first …

Logistics

WIFI Access:	
Restrooms:	
Chargers:	
Material of the workshop:	https://github.com/neo4j-field/gsummit2023 SCAN ME

graphsummit

Introduction

A short overview of the Neo4j Product Portfolio

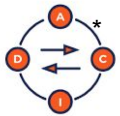
Neo4j Native Graph Database - What is it?

graphsummit

Neo4j is a NORMAL Database (DB)

- Comparable with other DBs like Postgres, MySQL / MariaDB, Oracle, HANA DB, etc.
- Securly save, insert, update and query data
- Backup + Recovery
- Highly scalable

Neo4j is “ACID compliant” and transaction save



- **ACID** (atomicity, consistency, isolation, durability)
- Data is stored consistently on transaction level without any losses during an outage
- **Most relational DB’s are ACID compliant, too.**

Neo4j is used 80%+ for OLTP Workloads

- > 80% of Neo4j Customers using the database as normal OLTP data store
- Data is stored, changed, deleted and queried
- With the same or even better performance compared to relational database systems

* Source: Medium.com - Link to the Article

neo4j

What is the Value Using a Native Graph-Database?

How data is stored on disk

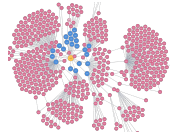
- Data is already stored connected
- Data is efficiently placed on disk using “Index-free Adjacency”
- A GDB stores **Nodes** and **Relations** instead of Rows and Columns
- **Semantics** may be build into the data!!

How to query the data

```
(:Product)-[:CONTAINS]->(:Part)
```

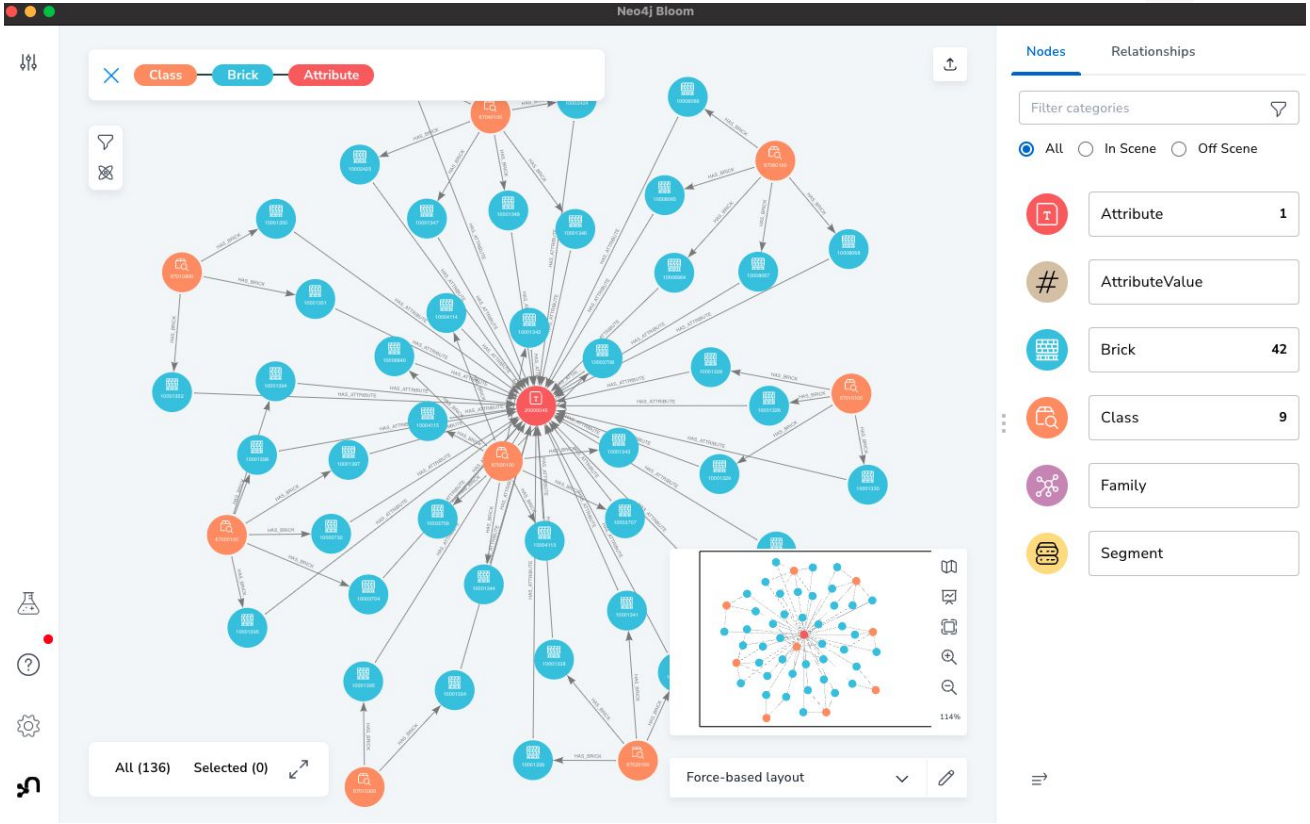
- Cypher Query Language vs. SQL (ISO -> GQL)
- Simple, lesser lines of code, easier to read and maintain
- Queries possible, that span 100+ Hops in the Graph, which is comparable with SQL Joins over 100+ Tables!

Storing complex data networks and semantics

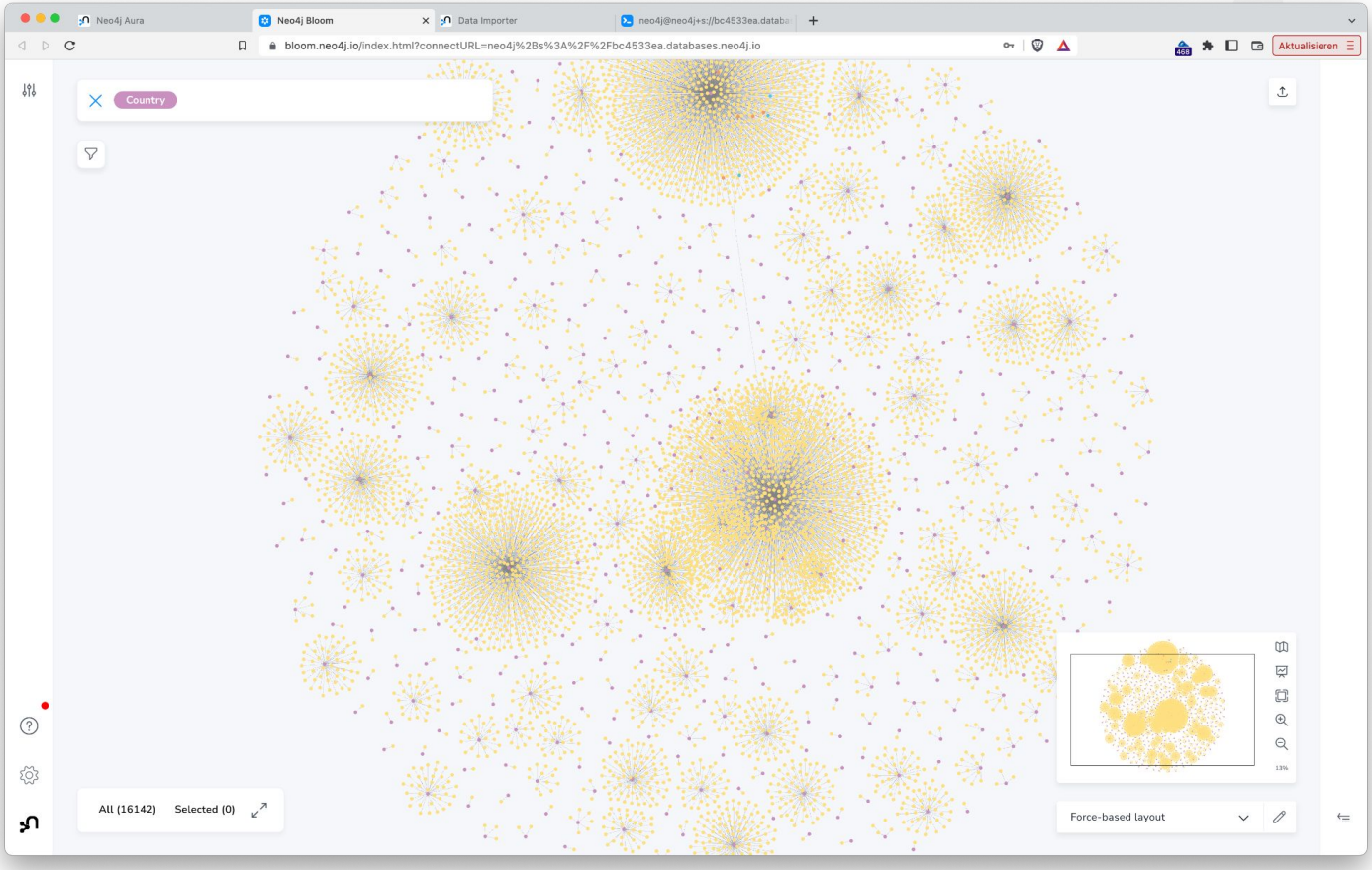


- Storing and analyzing complex relations between data
- Analyzing data from traditionally siloed data sources
- Extendable with data science algorithms and AI

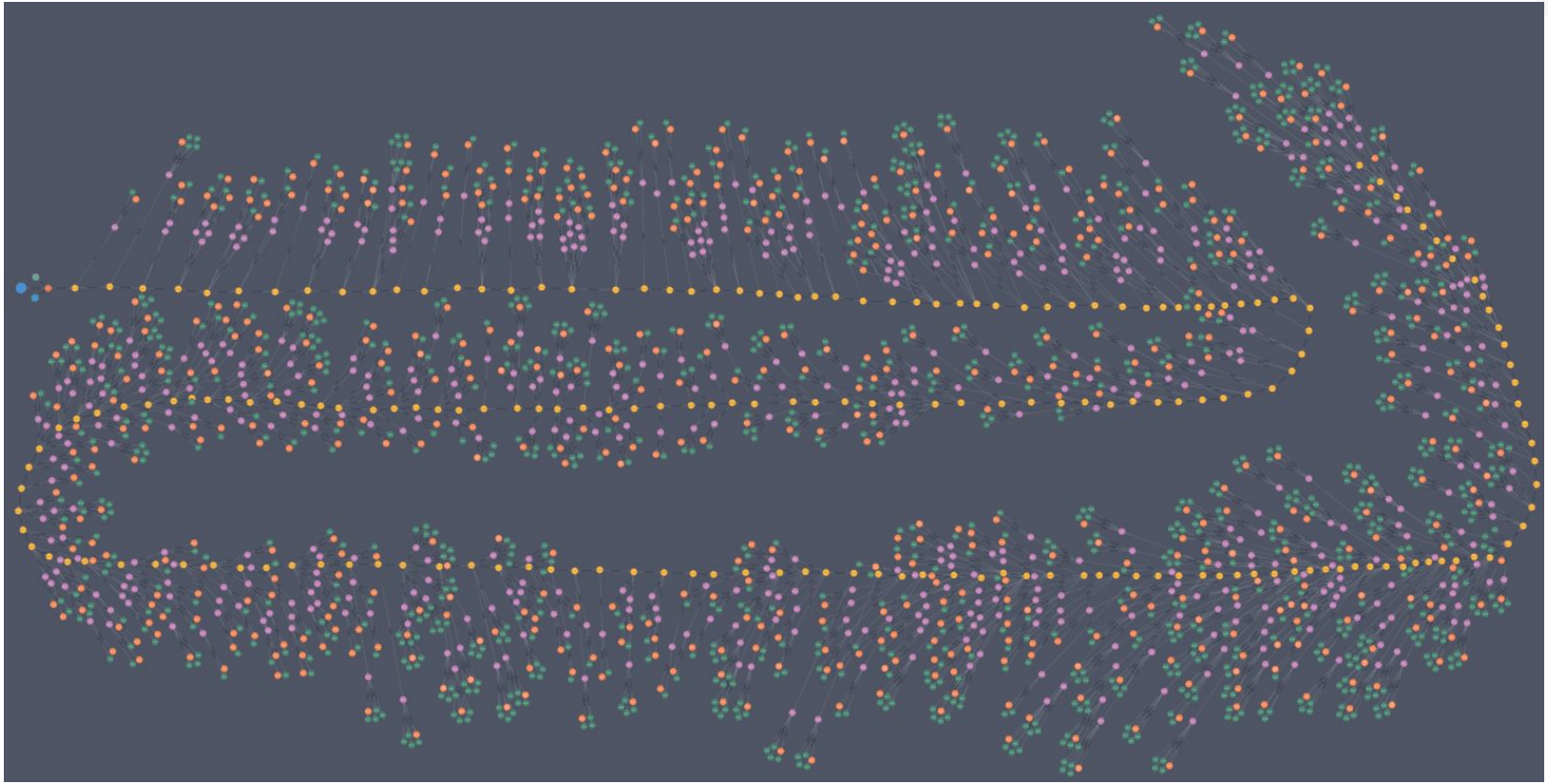
This is a **Graph …**



This is a **Graph, too …**



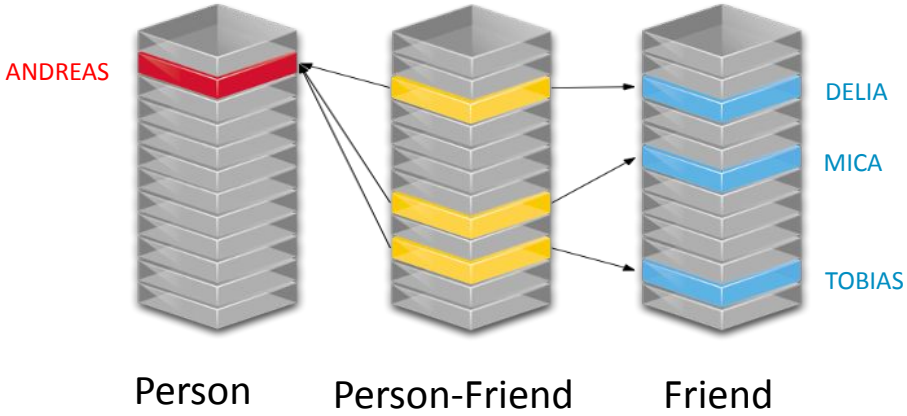
… and this is a Graph*! Guess, what graph could it be?



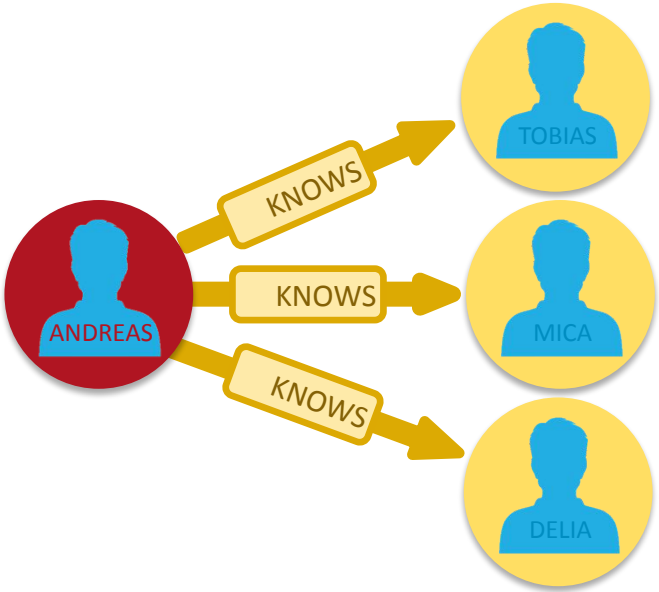
* https://www.graphable.ai/blog/patient-journey-mapping/

RDBMS vs Graph Model

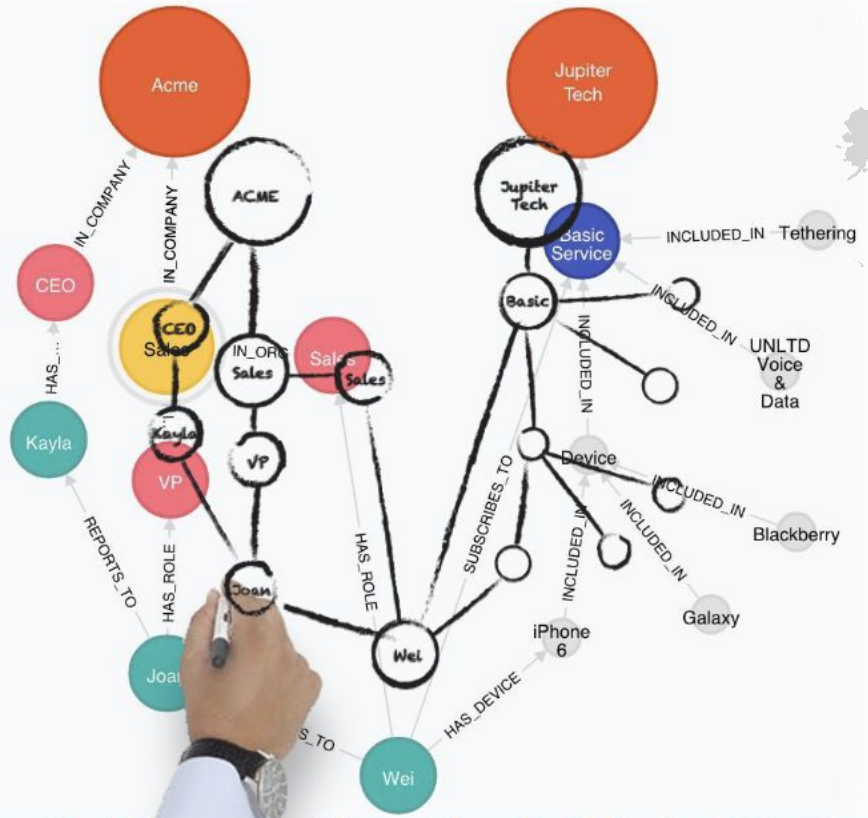
Relational Model



Graph Model



The Whiteboard Model Is the Physical Model

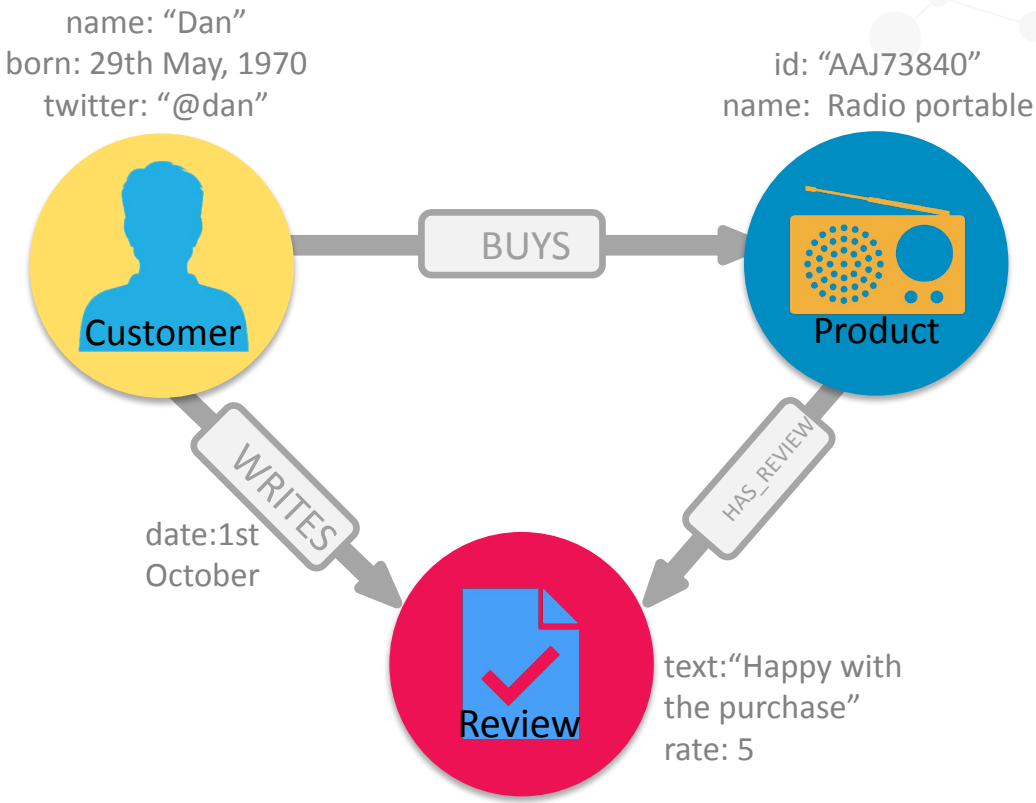


Labeled Property Graph Model

Nodes
- Can have *Labels* to classify nodes
- Labels have *native indexes*

Relationships
- Relate nodes by type and direction

Properties
- Attributes of Nodes & Relationships
- Stored as Name/Value pairs
- Can have indexes and composite indexes



What is **Graph Data Science**

Data science when ***relationships matter***

Queries
(Cypher)

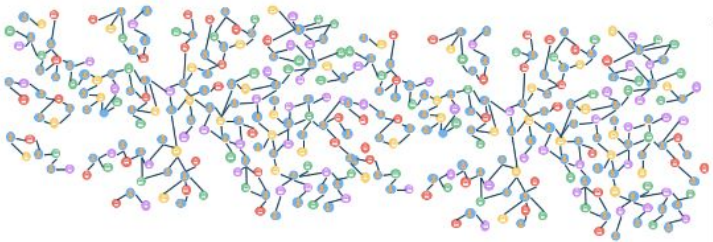
Fast, local decisioning and pattern matching

Graph Algorithms
(e.g. Neo4j graph data science library)

and iterations

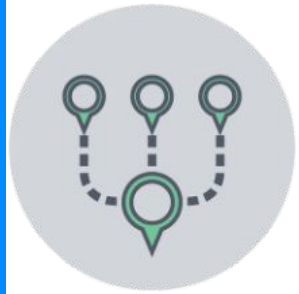
Local Pattern ⟵ ⟶ **Global Computation**

You know what you're looking for and making a decision



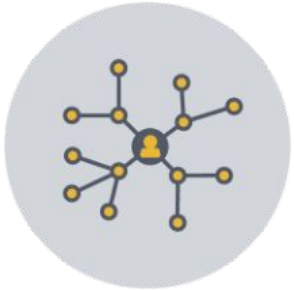
You're learning the overall structure of a network, updating data, and predicting

Graph Algorithms Verticals



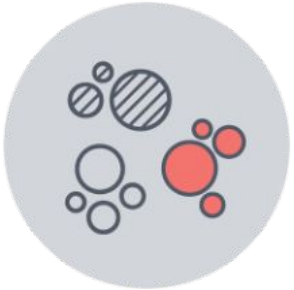
Pathfinding and Search

Finds optimal paths or evaluates route availability and quality.



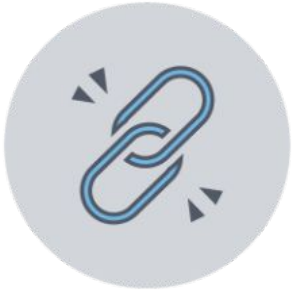
Centrality

Determines the importance of distinct nodes in the network.



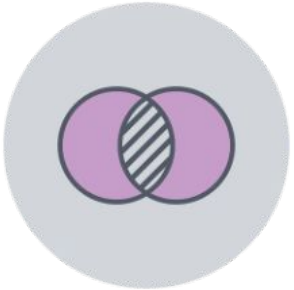
Community Detection

Detects group clustering or partition.



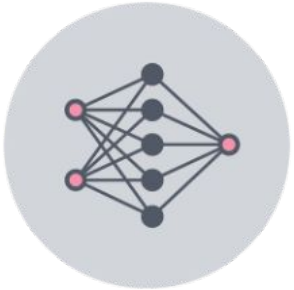
Heuristic Link Prediction

Estimates the likelihood of nodes forming a future relationship.



Similarity

Evaluates how alike nodes are by neighbors and relationships.



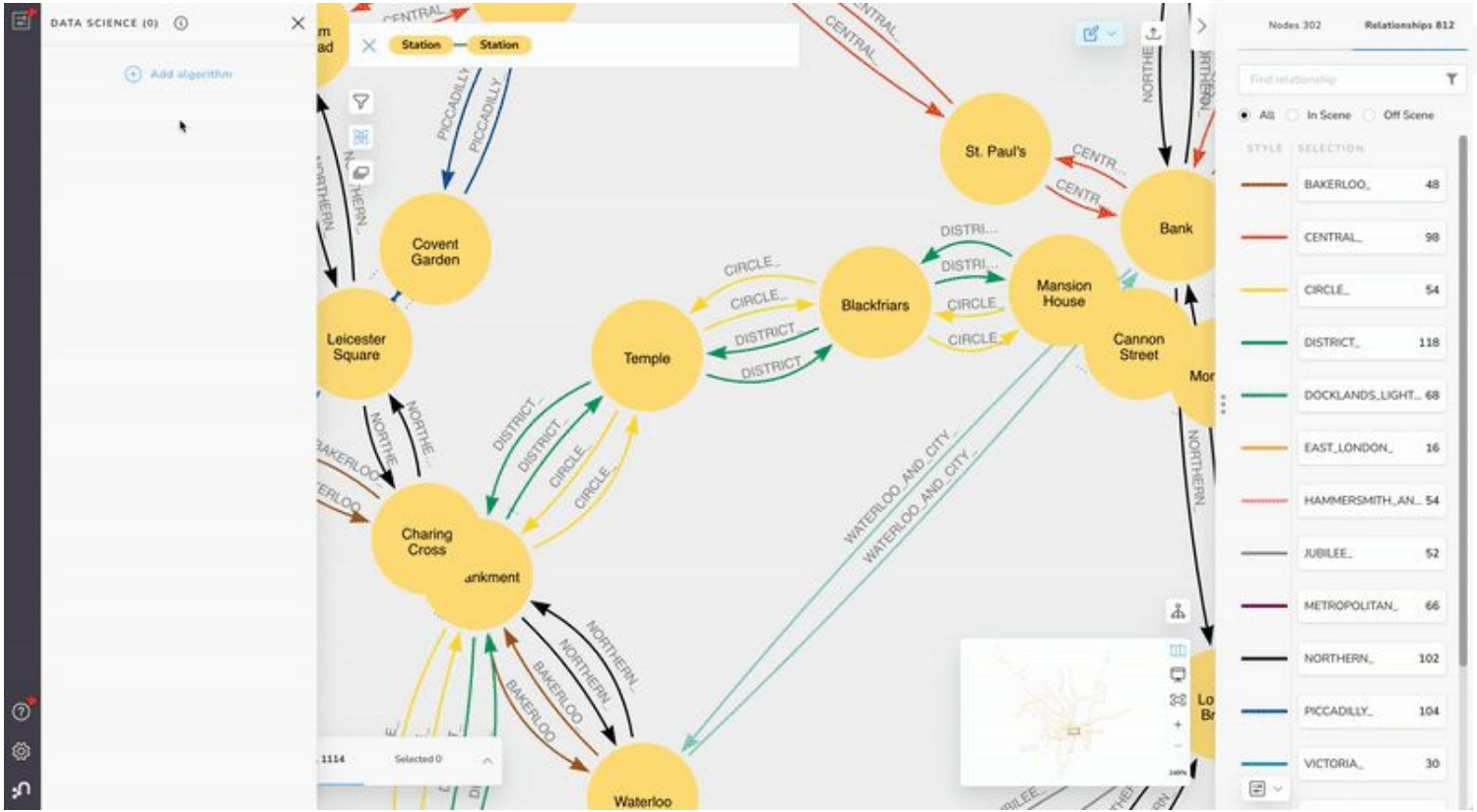
Embeddings

Learns graph topology to reduce dimensionality for ML

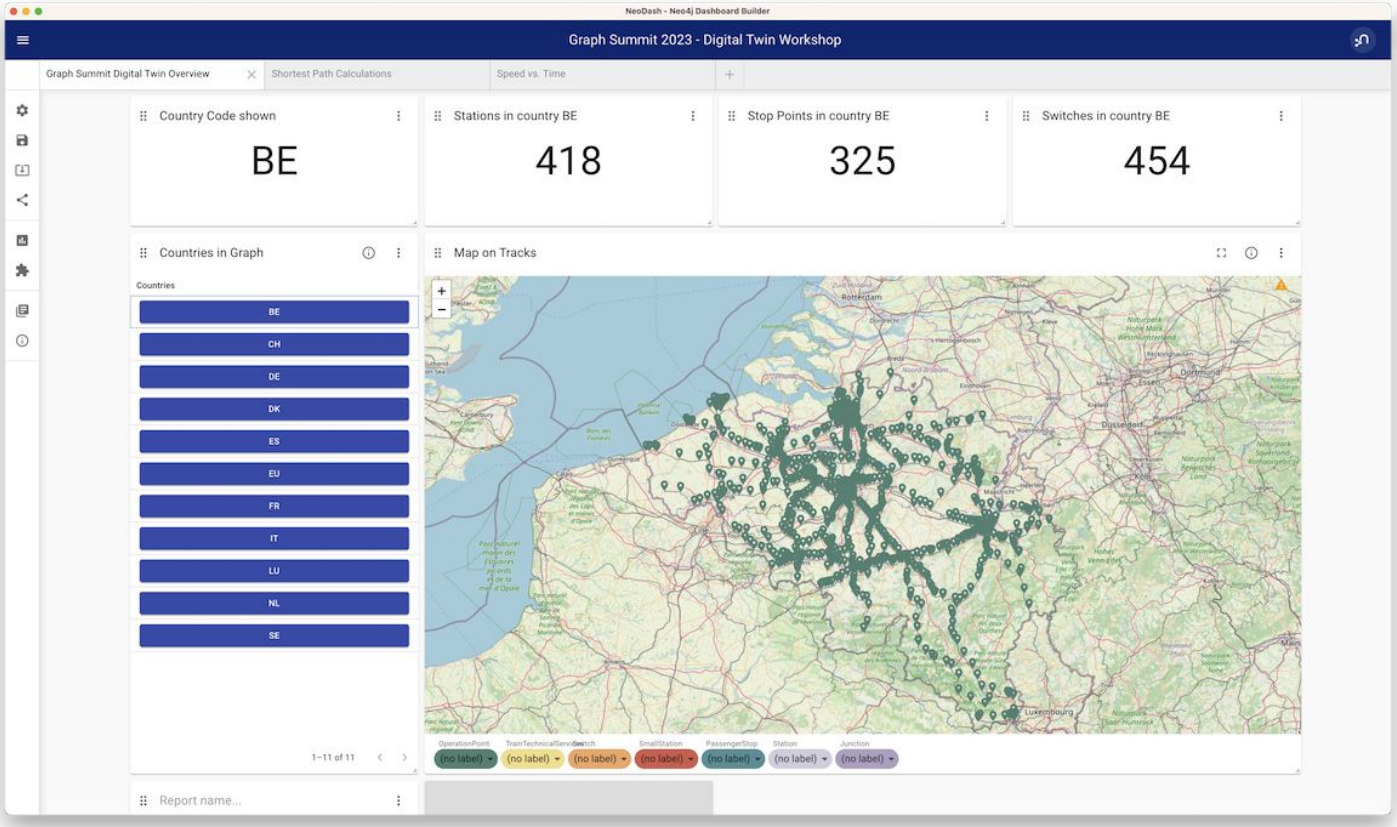
65+ graph algorithms in Neo4j

neo4j

Neo4j Visualisation Platform - Neo4j Bloom



NeoDash - Dash Boarding with Graph Data



graphsummit

Use Case Explanation

Digital Twin - An Overview

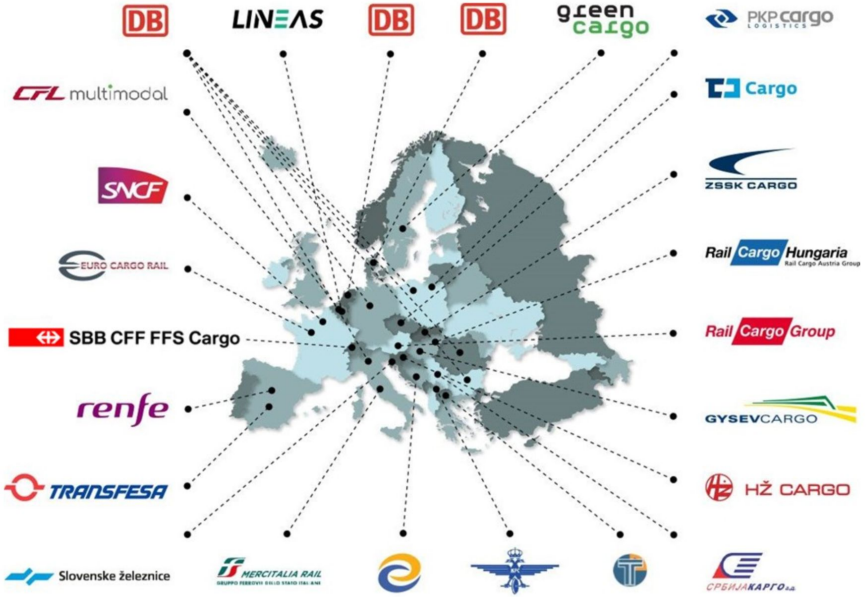
What is a **Digital Twin?**

A **Digital Twin** is a digital representation of an intended or actual real-world physical product, system, or process (a *physical twin*) that serves as the effectively indistinguishable digital counterpart of it for practical purposes, such as simulation, integration, testing, monitoring, and maintenance.

It has been done before

EU Railway Network - Track & Trace

- Challenge: **Legacy technology could not** track and analyze train journeys
- Solution: **Neo4j Knowledge Graph**
- **Identify and avoid bottlenecks (DB)**



Why do we need a Digital Twin (some reasons)

Improved efficiency

It can optimize its operations and reduce costs by simulating different scenarios and making data-driven decisions.

Enhanced safety

It can identify potential hazards and test safety measures to improve safety for passengers and employees.

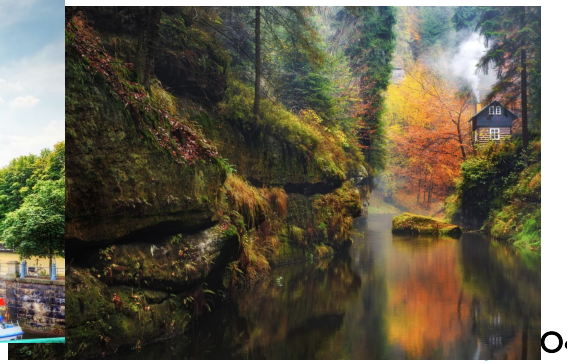
Predictive maintenance

It can monitor asset condition in real-time, predict maintenance needs, and increase asset lifespan.

Improved customer experience

A digital twin can simulate disruptions and help proactively address issues to enhance the customer experience and increase satisfaction.

POIs included



graphsummit

Unleash the power of the graph

neo4j

graphsummit

Abstracting from this use case

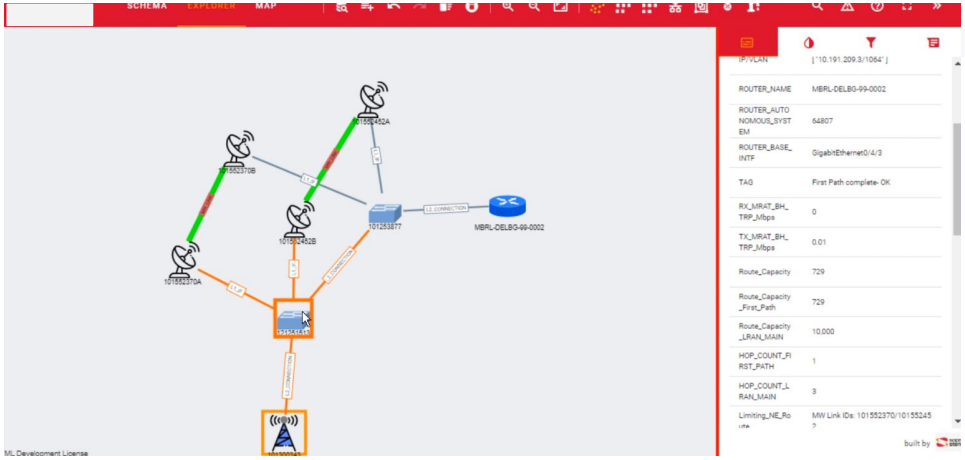
Digital Twin and Networks are everywhere

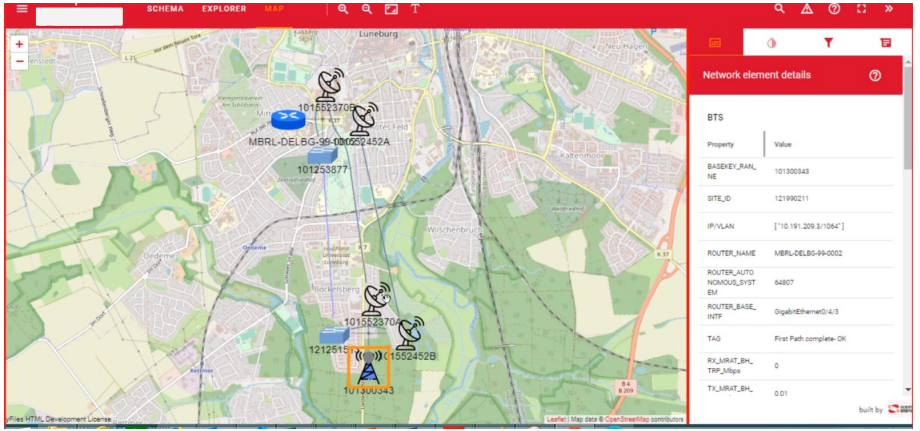
Many other use cases are networks, too!

- Supply Chain
- Mobile phone networks
- Power Grids / Gas Grids / Fibre Networks
- Social Networks
- Patient Journeys
- etc.

Mobile Network Operations - Large Mobile Provider

- Digital Twin of Mobile Network
- Planning and operation
  - Low signal strength of tower
  - Moving equipment around
- Simulation of possible failure scenarios
- Repair & maintenance support
  - Supply Chain Impact





"We wanted to create a solution that exploits artificial intelligence, integrates current and historical AIS positions as well as multiple data feeds and APIs. Such an effort would require a world-class infrastructure. That's why we selected Neo4j."

David Levy
Chief Marketing Officer
OrbitMI

Customer Case Study:
Logistics and Supply Chain

Plan maritime routes based on distances, costs, and internal logic.

Results:

- **Subsecond** maritime routes planning
- Reduce global carbon emissions **60,000 tons**
- **12-16M ROI** for OrbitMI customers

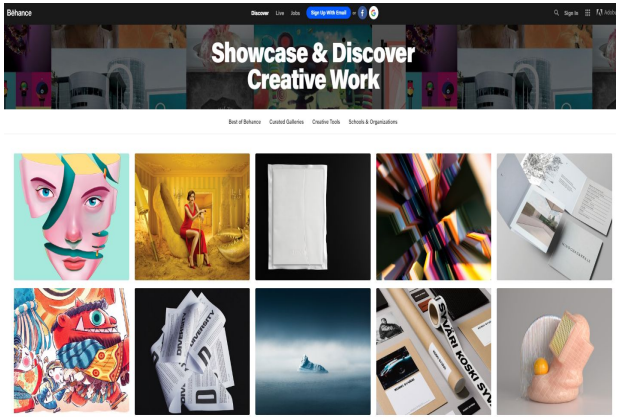
OrbitMI

Adobe

graphsummit

Adobe Behance

Social Network of 10M
Graphic Artists



Background

- Social network of 10M graphic artists
- Peer-to-peer evaluation of art and works-in-progress
- Job sourcing site for creatives
- Massive, millions of updates (reads & writes) to Activity Feed
- *150 Mongos to 48 Cassandras to 3 Neo4j's!*

Business Problem

- Artists subscribe, appreciate and curate "galleries" of works of their own and from other artists
- Activities Feed is how everyone receives updates
- 1st implementation was 150 MongoDB instances
- 2nd implementation shrunk to 48 Cassandras, but it was still too slow and required heavy IT overhead

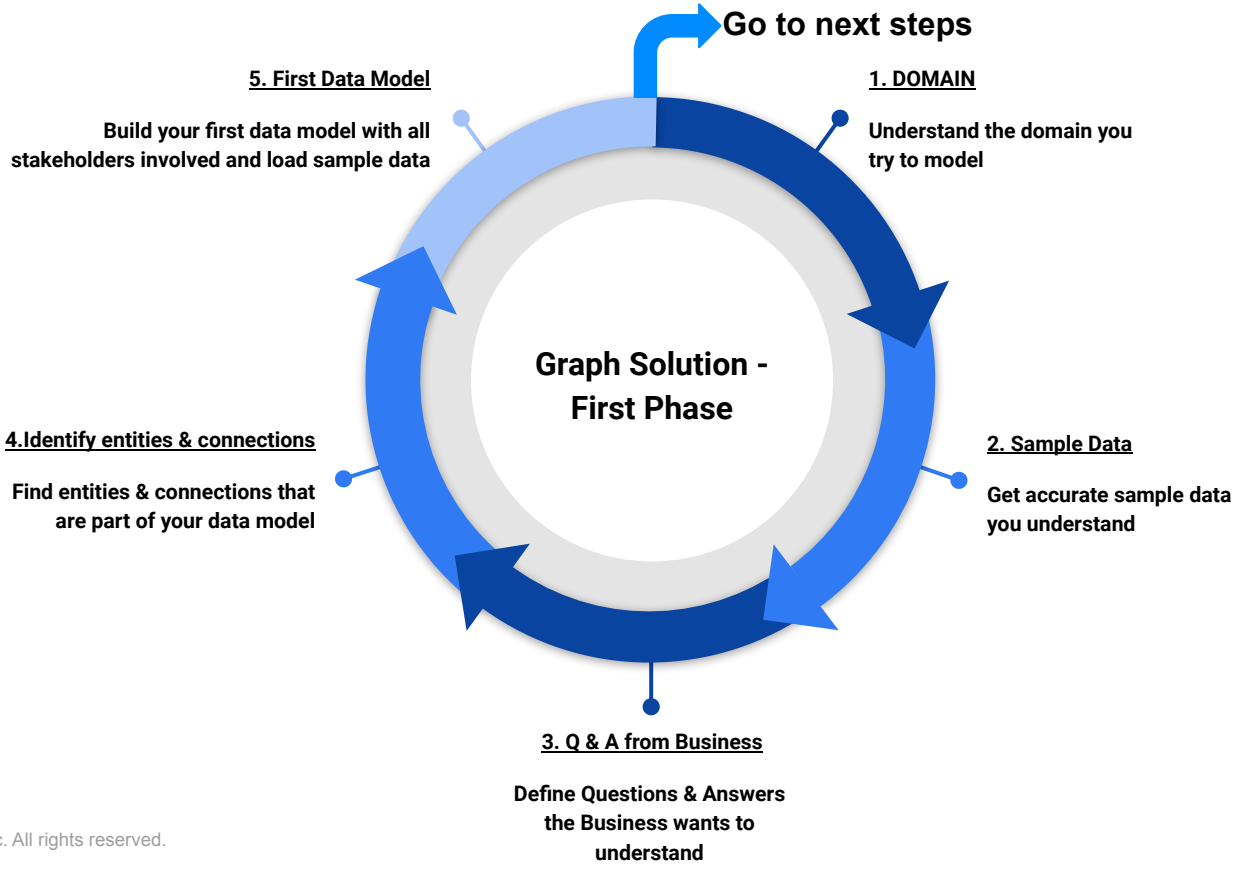
Solution and Benefits

- 3rd implementation shrunk to 3 Neo4j instances
- Saved over $500k in annual AWS fees
- Reduced data footprint from 50TB to 40GB
- Significantly easier to introduce new features like, "New projects in your Network"

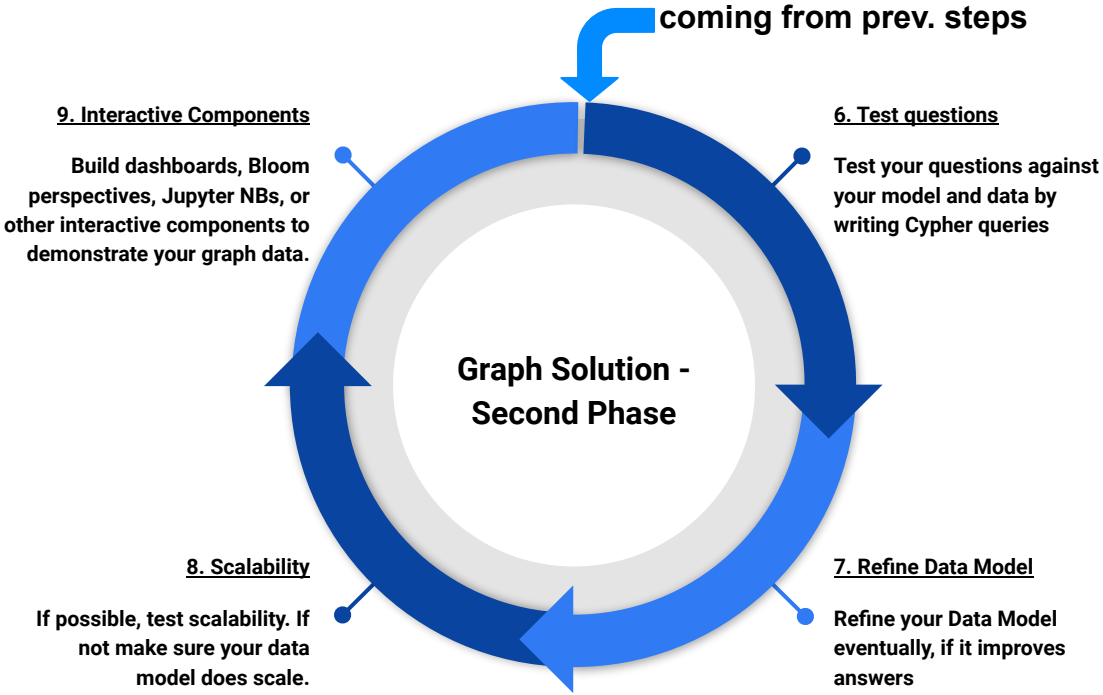
neo4j

Building our solution

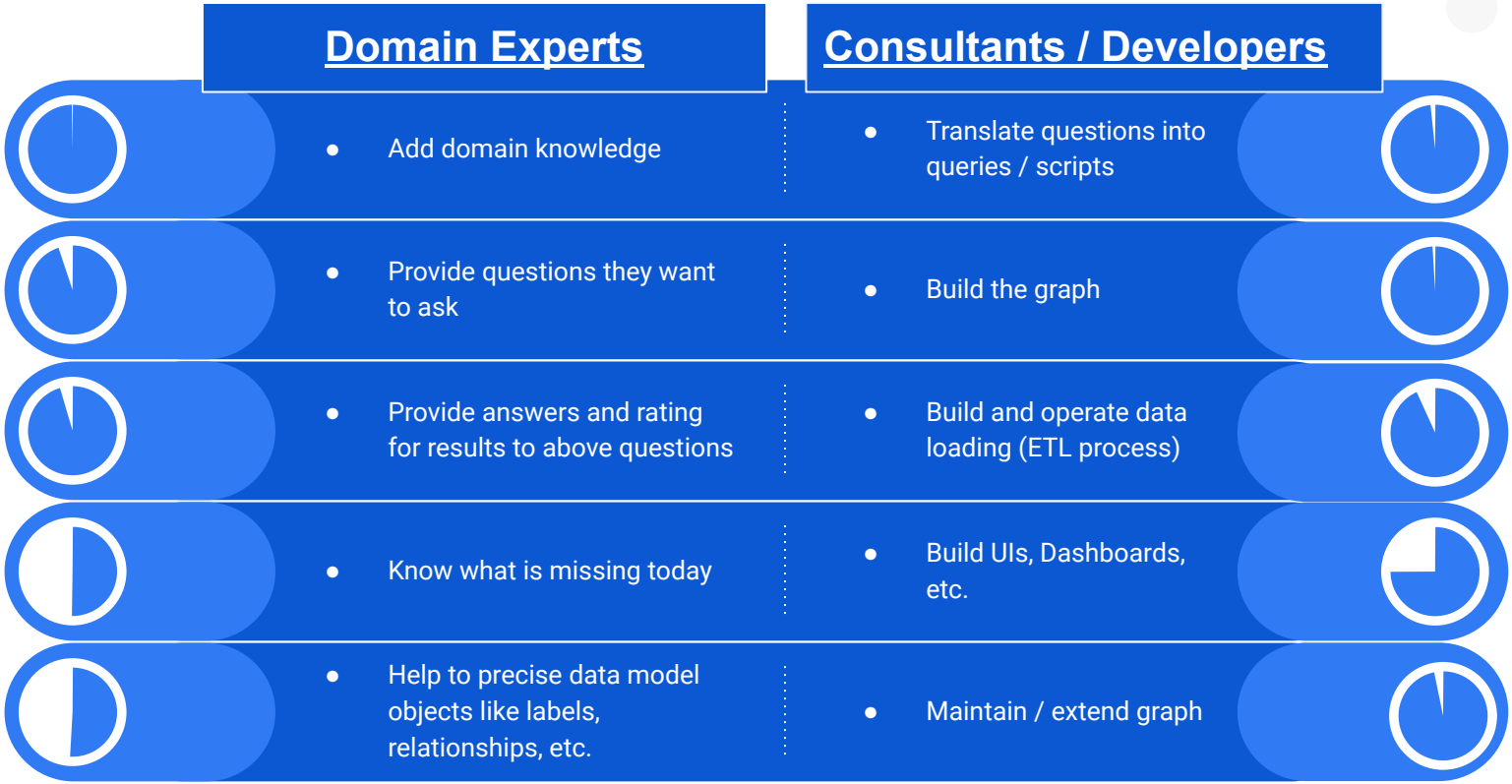
High Level Approach building a Graph Solution



High Level Approach building a Graph Solution



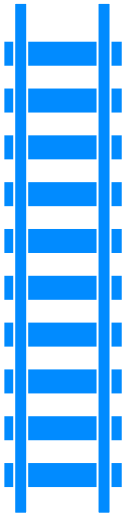
Stakeholders recommended to build a graph solution



graphsummit

The Sample Data Set

Sample Data: Railway Network Digital Twin dataset



Provided Data:

- Consists of a Railway **Track** Network + **Operation Point (OP)**
- **It includes attributes like:**
  - Tracks with Track Numbers,
  - Stations and other OPs,
  - Geo location information,
  - Track (**Section**) length and Speed,
  - other parameters

- Add on from us: **Point of Interest** (POI) along tracks

- **Format**: CSV → Generated from XML
- Origin of Data: https://data-interop.era.europa.eu/search

What Does the Sample Data Looks Like?

- Operation Points Data*:
- Tracks & Speed Data*:
- Point-of-Interest Data:
  - https://github.com/neo4j-field/gsummit2023/tree/main/data

Origin of data:

- https://data-interop.era.europa.eu/search

The data set is public available and can be used under the license and conditions mentioned by OpenDB. We have translated the data to make it better understandable to everyone outside Germany.

*** Translation of Shortcuts and certain Operational Units into english done by the Neo4j team!**

The Data Explained - In an Easy Way

“The data can be seen as a highway that has ramps throughout its entire path.

Each ramp can be an **Operation Point** (OP) like a Station, a Switch, etc.

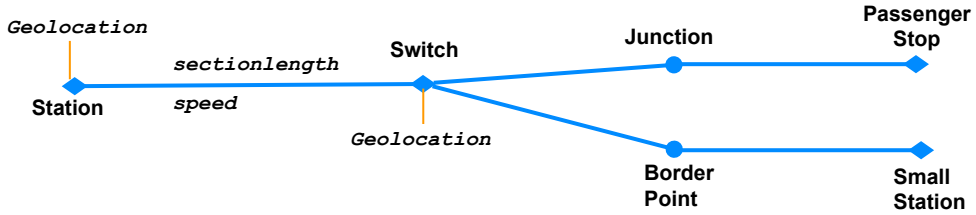
Tracks have a length and a speed associated. **Tracks have a start point and an endpoint and will be inter-connecting Operation Points**.

An Operation Point has a name and is referred to by a relationship with a country code.

Operation Points (OP) - Data Explanation (Nodes)

CSV Header titles:

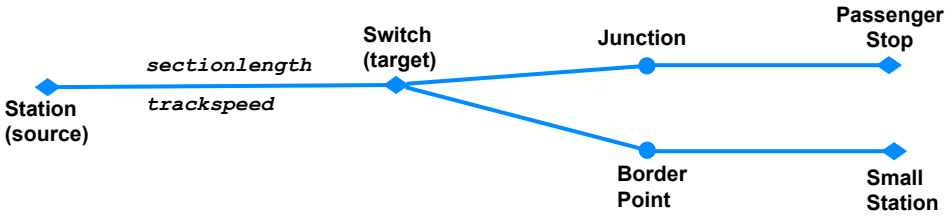
- **id**: the internal number of the OP
- **extralabel**: the kind of OP we deal with, e.g. Station, Junction, Switch, etc.
- **name**: the name of a OP
- **latitude**: of the OP
- **longtitude**: of the OP



Section Connection Data Explanation (Relationships)

CSV Header titles:

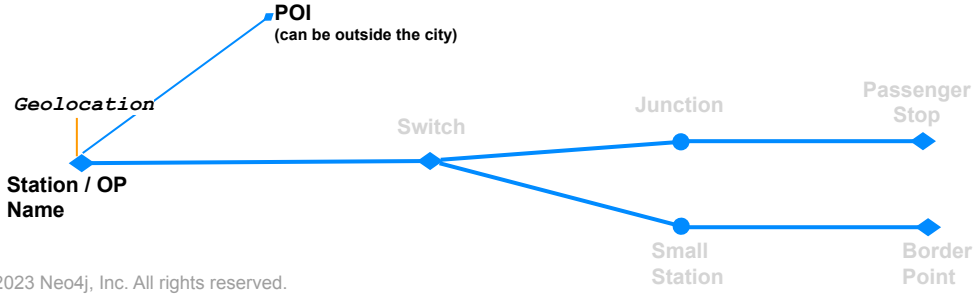
- **source**: start OP for this section
- **target**: end OP for this section
- **sectionlength**: the length in km of that section
- **trackspeed**: max speed allowed on that section



Point of Interest (POI) Data Explanation

CSV Header titles:

- **CITY:** City the POI is in or close by
- **POI_DESCRIPTION:** A short description of of the POI
- **LINK_FOTO:** a short name
- **LINK_WEBSITE:** is the name of the track corresponding to the shortcut
- **LAT:** Latitude of the POI
- **LONG:** Longitude of the POI
- **SECRET:** True if not well know, False if well known (e.g. Berlin)



graphsummit

Data Modeling - Writing down Questions & Answers

Data Modeling - The Questions are the Input

1. How long is the **journey** from **station** X to station of **station** Y?
2. Do I have alternative routes if a **station** in my journey is down?
3. What is the *shortest distance/path*, between **stations** X and Y?
4. Can I **geolocate** stations in my rail track network?
5. What are the stations expecting the most traffic?
6. What **POI**s **has** a station (city) along my route?

Note: Aggregations are also possible, but we focus on the “graphy” queries here!

Expected Answers

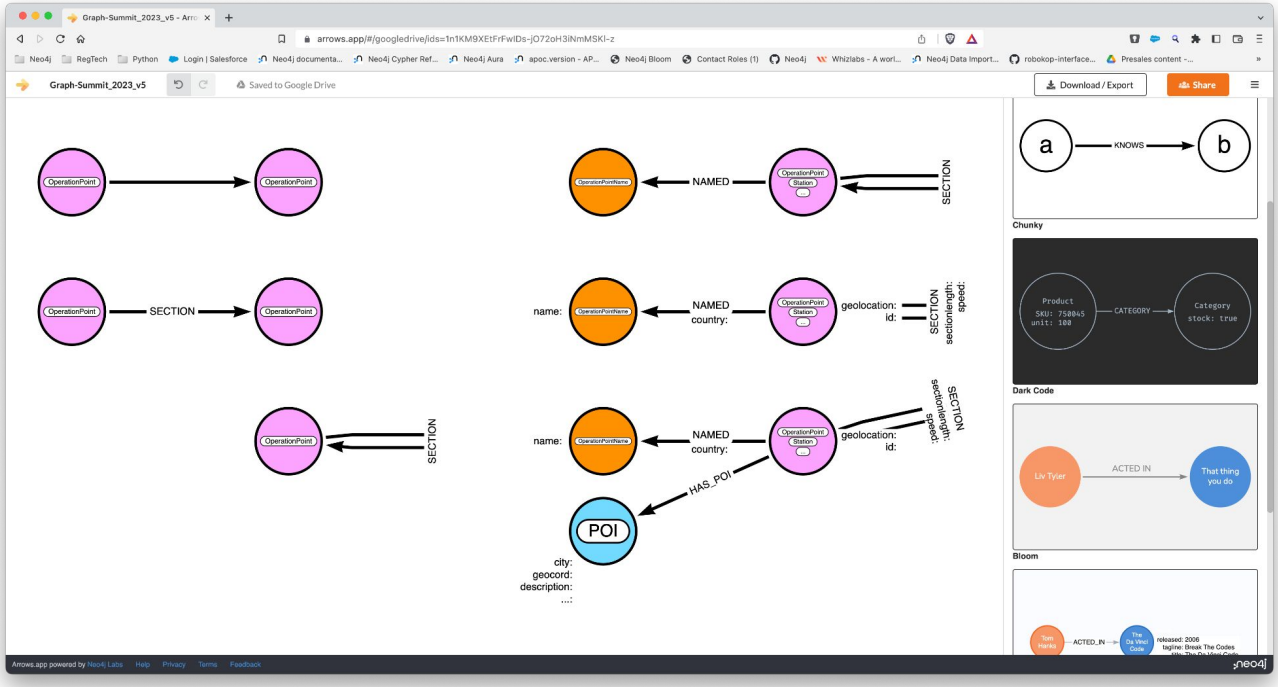
1. Your journey is 30 stations and 239km long
2. Your alternative route is 36 stations and 271km long
3. The shortest route between X and Y is XXX km
4. The geo location of your station is Lat / Long
5. The station with the most traffic is … ?
6. The following POIs are along your route …

graphsummit

Data Modeling - Building the Model

Tools for Graph Data Modeling

Arrows Tool → https://arrows.app



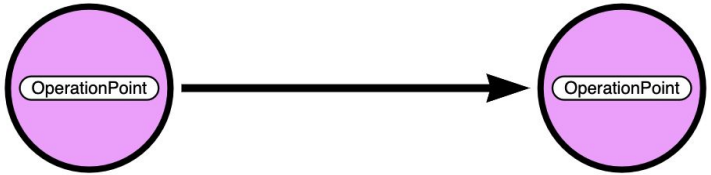
Data Modeling Using the Questions as Input

Remember those questions?

1. How long is the **journey** **from** **station** X **to** station of **station** Y?
2. Do I have alternative routes if a **station** in my journey is down?
3. What is the ***shortest*** *distance/path*, **between** **stations** X and Y?
4. Can I **geolocate** stations in my rail **track** network?
5. What are the stations expecting the most traffic?
6. What **POI**s **has** a station (city) along my route?

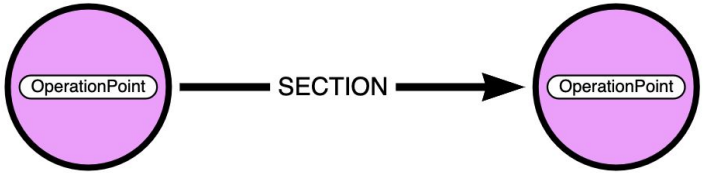
Building a First Data Model 1/6

- Nouns in your questions are the nodes, verbs are the relationships
- Nodes have a **Label**, Relationships MUST have a **Type**
- Properties should help to uniquely identify Nodes and/or answer questions



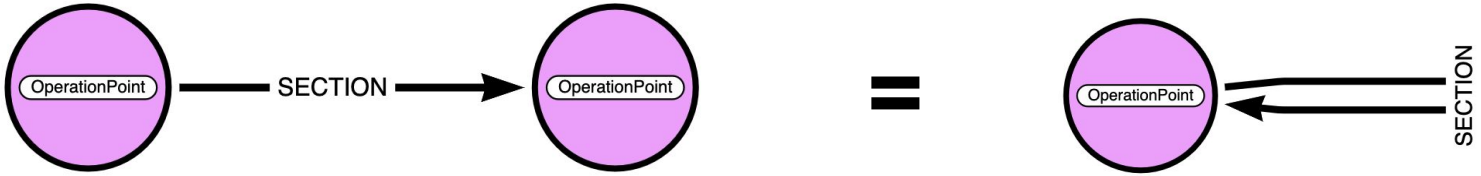
Building a First Data Model 2/6

- Nouns in your questions are the nodes, verbs are the relationships
- Nodes have a **Label**, Relationships MUST have a **Type**
- Properties should help to uniquely identify Nodes and/or answer questions



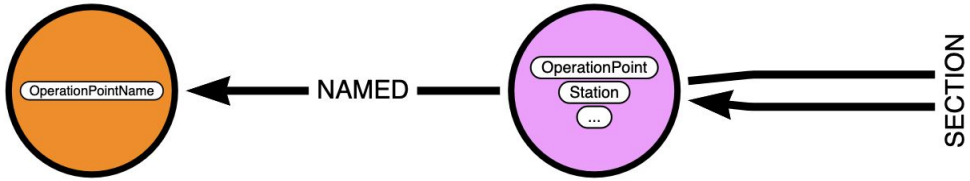
Building a First Data Model 3/6

- Nouns in your questions are the nodes, verbs are the relationships
- Nodes have a **Label**, Relationships MUST have a **Type**
- Properties should help to uniquely identify Nodes and/or answer questions



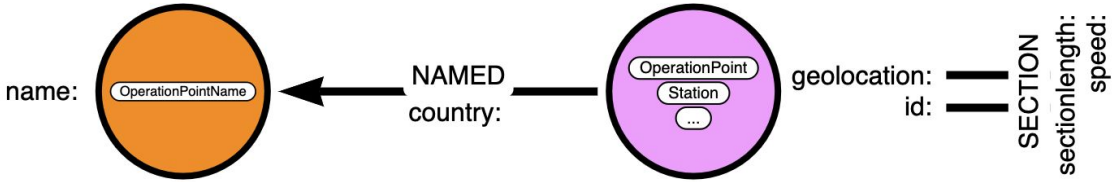
Building a First Data Model 4/6

- Nouns in your questions are the nodes, verbs are the relationships
- Nodes have a **Label**, Relationships MUST have a **Type**
- Properties should help to uniquely identify Nodes and/or answer questions



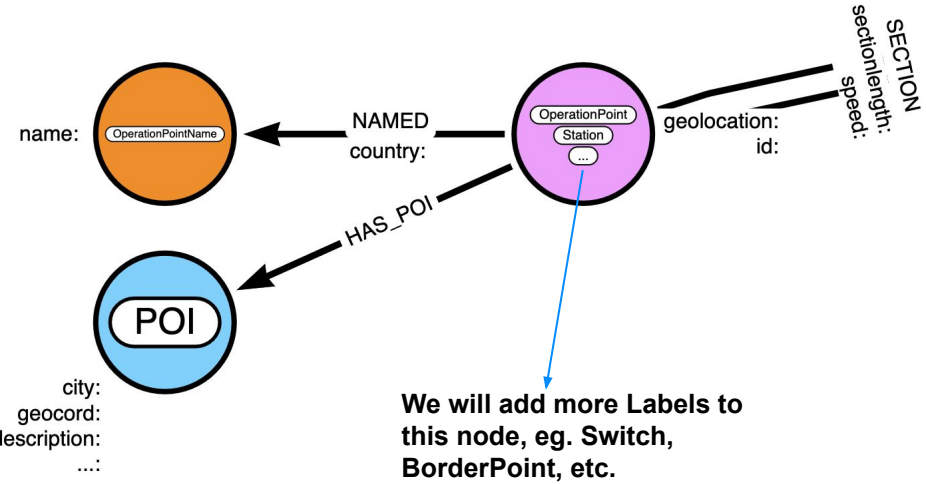
Building a First Data Model 5/6

- Nouns in your questions are the nodes, verbs are the relationships
- Nodes have a **Label**, Relationships MUST have a **Type**
- Properties should help to uniquely identify Nodes and/or answer questions



Building a First Data Model 6/6

- Nouns in your questions are the nodes, verbs are the relationships
- Nodes have a **Label**, Relationships MUST have a **Type**
- Properties should help to uniquely identify Nodes and/or answer questions

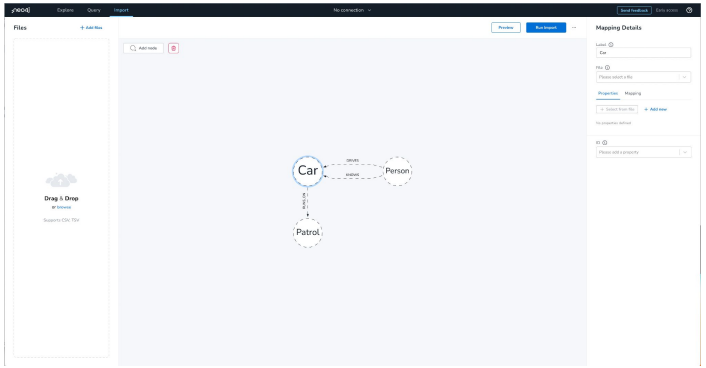


graphsummit

Data Loading

Data Loading - How Can I Do That?

- **Cypher CSV Load**
- APOC Library
- neo4j admin import command
- ETL Tool
- Python API
- Neo4j Workspace*
- 3rd Party Tools (Apache Airflow or HOP, Cloud provider tools, etc.)



Neo4j Workspace*

More about Data loading, etc. **on the Neo4j Website**

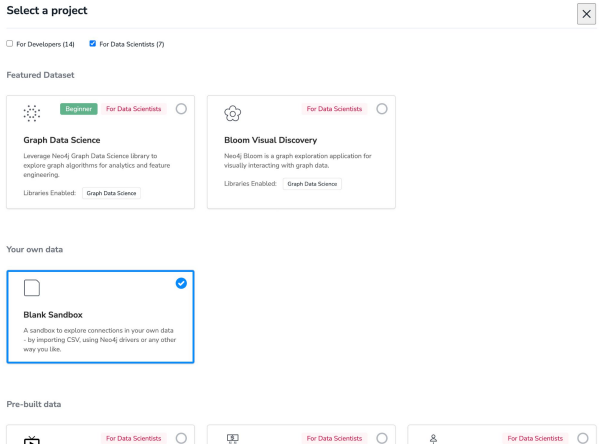
Workshop Time

Let’s build the solution together …

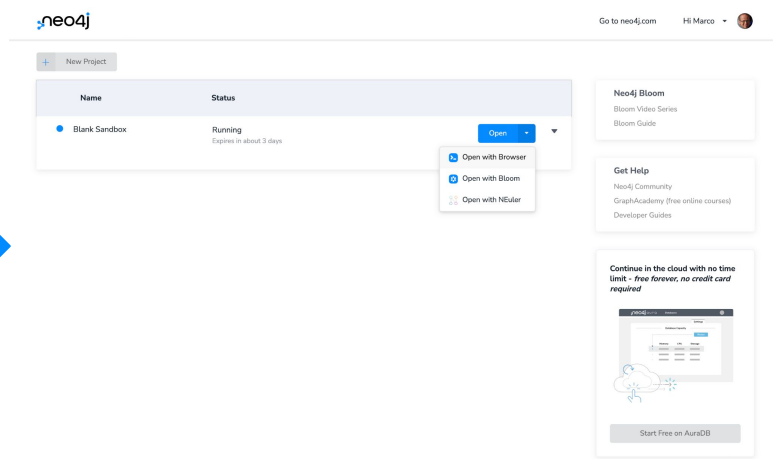
Lets do it together …

1. Open the Github page at: https://github.com/neo4j-field/gsummit2023
2. Open https://sandbox.neo4j.com/ and create a Neo4j Sandbox (Google login required!)

a)



b)



3. Open Neo4j Browser and login with your credentials
4. **Later** on open NeoDash at: https://neodash.graphdatabase.ninja/

Improving the Solution further

To add even more value to your business …

graphsummit

How could this “Mini Digital Twin” be enhanced?

Extending the Graph is easy, e.g. with further track information:

- For example with possible **pace on track**, **track quality metrics**, **mobile network coverage**, etc.
- Sensor information about switches to quickly find problems and re-route traffic
- The more related information is added, the more use cases could benefit from the digital twin

Going even further:

- Additional information about **trains running on the tracks** enable additional benefit:
  - Track and schedule optimization
  - Simulation of new track (to grow traffic)

neo4j

Q & A

Thanks for Graphing with us!

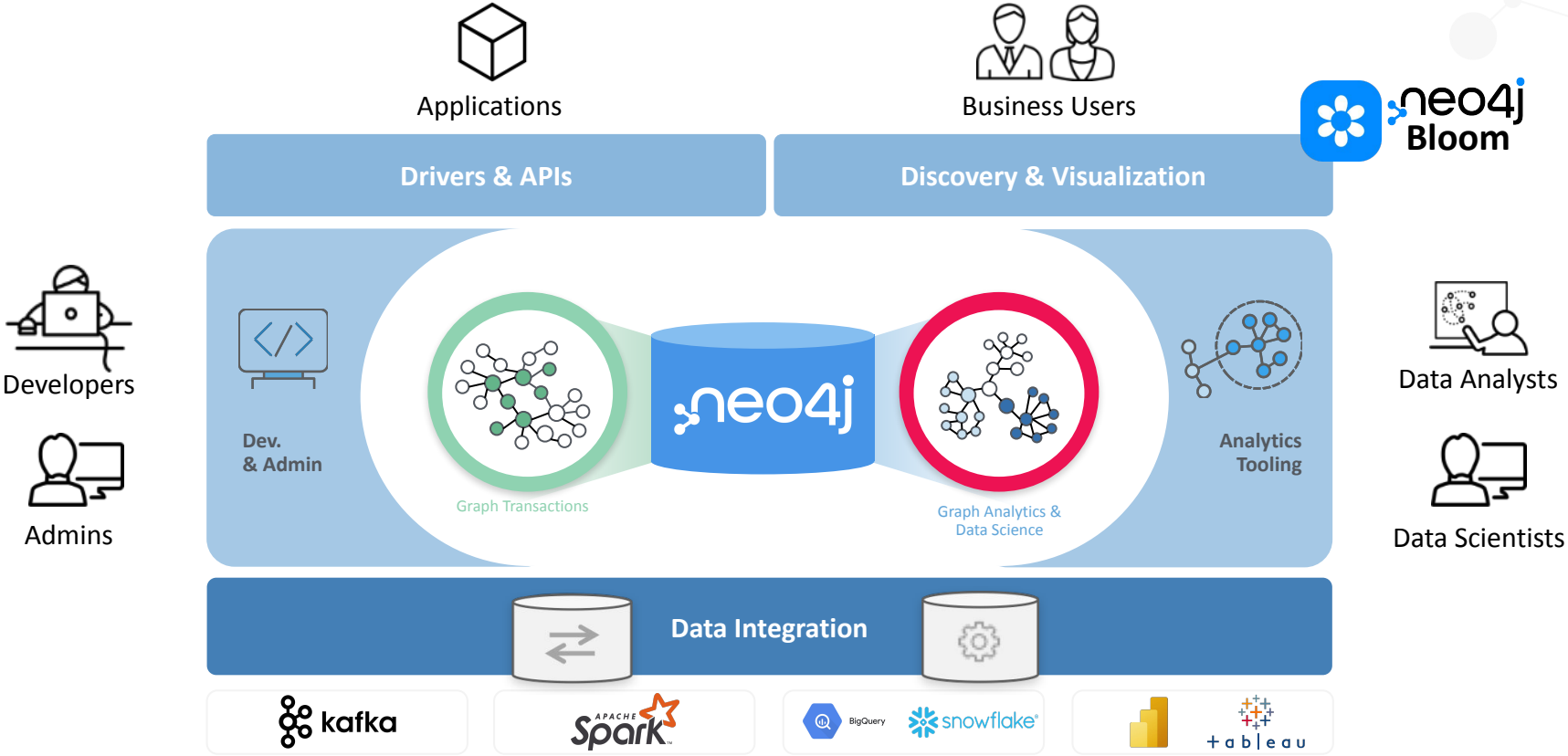
Contact us:
Marco.Deluca@neo4j.com
Luis.Salvador@neo4j.com
Gal.Bello@neo4j.com

graphsummit

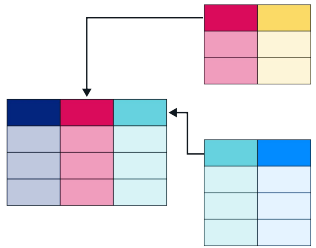
Contact us at
sales@neo4j.com

Additional Information

Neo4j Graph Data Platform Overview



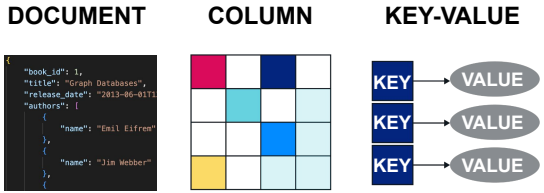
Use context to unlock insights



Relational Databases

***Don't* handle relationships well**

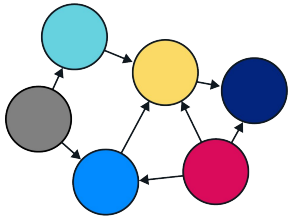
Practically limited to three degrees of separation



Other NoSQL Databases

Don't* handle relationships *at all

No language or structures for handling relationships: joins done through the application

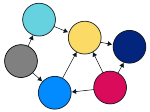
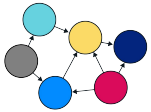
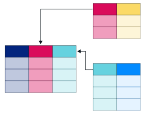
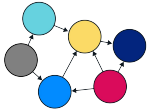
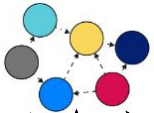
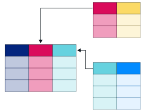
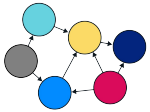
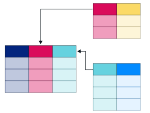


Graph Databases

***Natively* store & query relationships**

Queries run 1000x faster at scale, up to 1000+ hops

How Neo4j differentiates from other databases

	Native Graph DB	Non-Native Graph DB	RDBMS
Visualization			
Queries	Cypher (graphs)-[are]->(everywhere)	openCypher, Gremlin, Proprietary	SQL
Processing			
Storage			

Cypher

```
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
```

How to deploy Neo4j - 3 deployment models available

DB-as-a-Service

neo4j aura

Fully-managed SaaS
Consumption-based pricing

Cloud-native

Self-service deployment

No access to underlying infrastructure and systems

Cloud Managed Services

White-glove managed service by Neo4j experts

Fully customizable deployment model and service levels

Operate In own data centers or Virtual Private Cloud

Self-hosted

For private, hybrid or lift-and-shift cloud
Bring-your-own-license

Full control of your environment

Run in any cloud, in your account