neo4j

Modeling and querying for graph databases

Stefano Ottolenghi

! a cemetery

slido

Join at slido.com
#3244815

ⓘ Click **Present with Slido** or install our Chrome extension to display joining instructions for participants while presenting.

slido

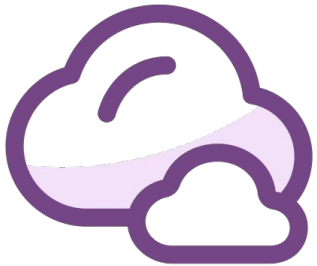
How is your Neo4j instance deployed?

ⓘ Click **Present with Slido** or install our Chrome extension to activate this poll while presenting.

Why should *you* care about graph databases?

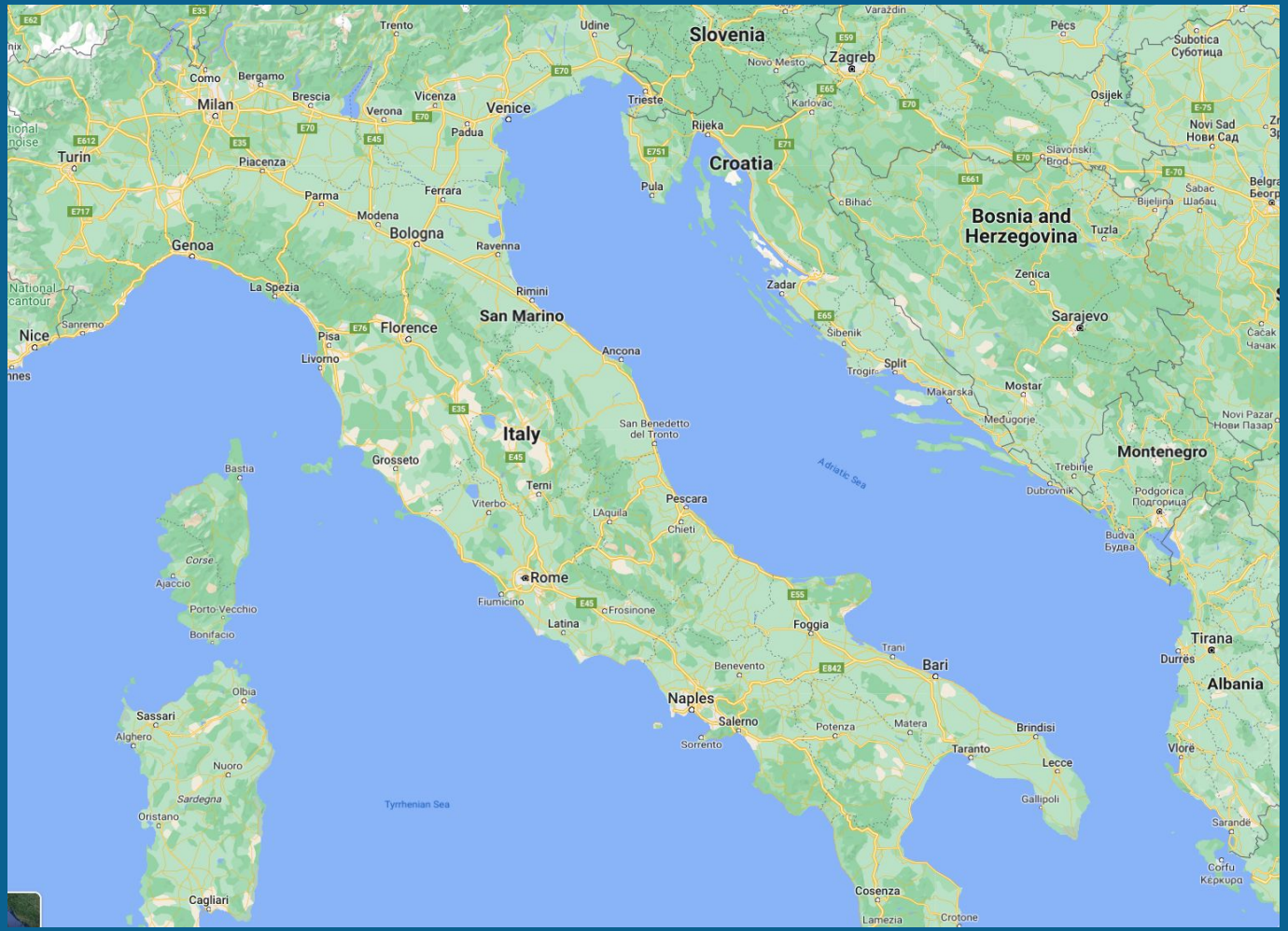
If you were to make up your job description, what would it be?

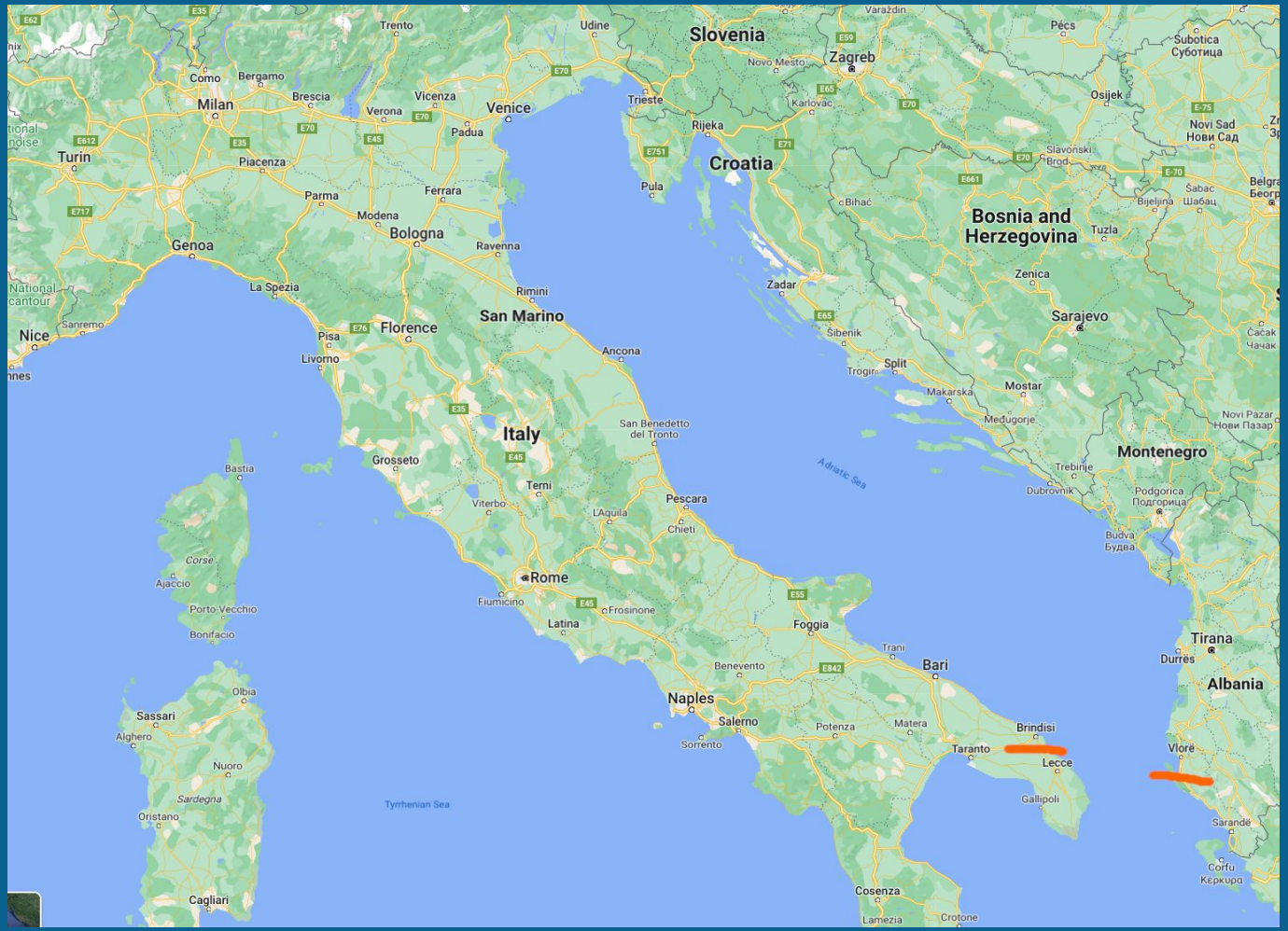
slido

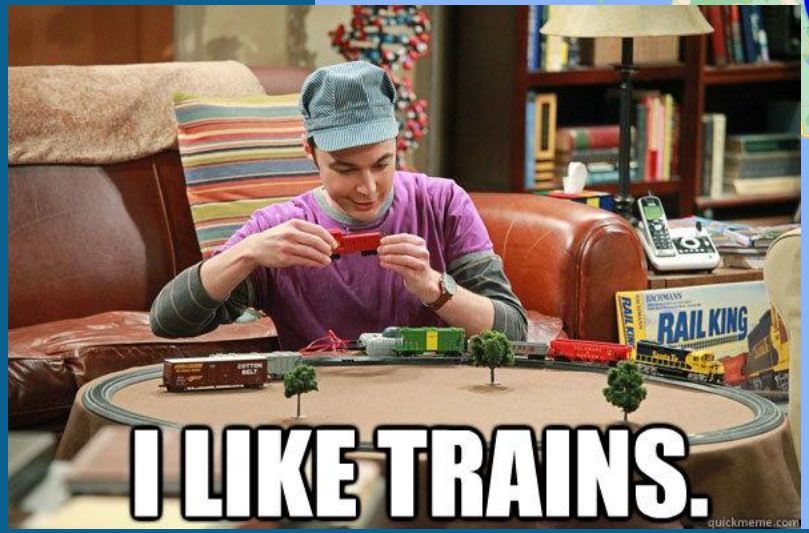


What's your dream job description?

Click **Present with Slido** or install our Chrome extension to activate this poll while presenting.







Think of yourself as a ...

The more tools you know, the more problems you can solve (*efficiently*).

Questions?

I may have answers.

Twenty minute introduction to graph databases

Find the fraudster

Ash gonna catch them ‘ll

Not Sinatra

Blue Eyes

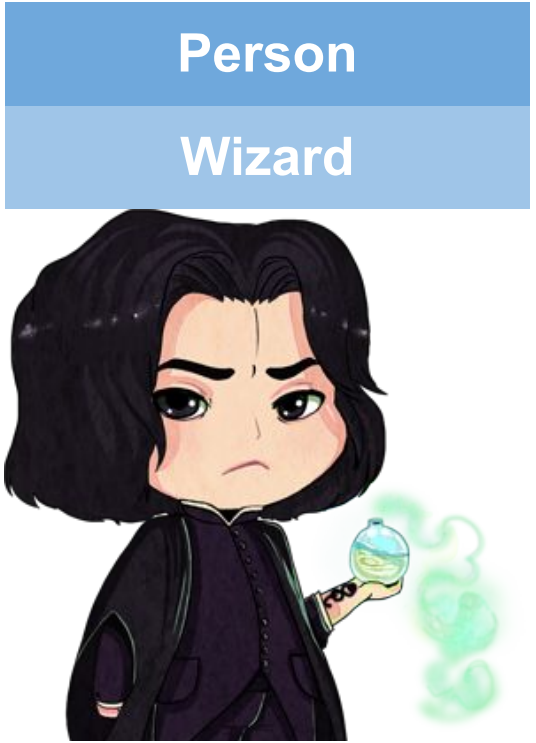
Terminology

Node - vertex, entity, object, thing

Terminology

Node - vertex, entity, object, thing

Label - concept, class(ification) of a node



Terminology

Node - vertex, entity, object, thing

Label - concept, class(ification) of a node

Relationship - edge, a physical link between two nodes

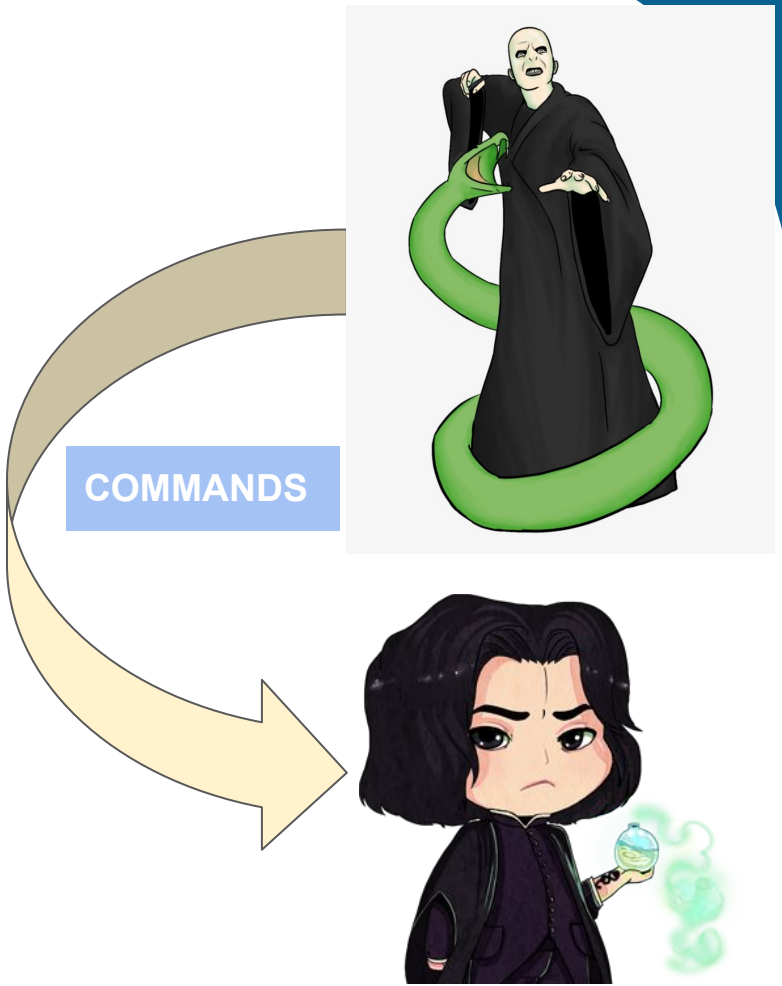
Terminology

Node - vertex, entity, object, thing

Label - concept, class(ification) of a node

Relationship - edge, a physical link between two nodes

Type - class(ification) of a relationship



Terminology

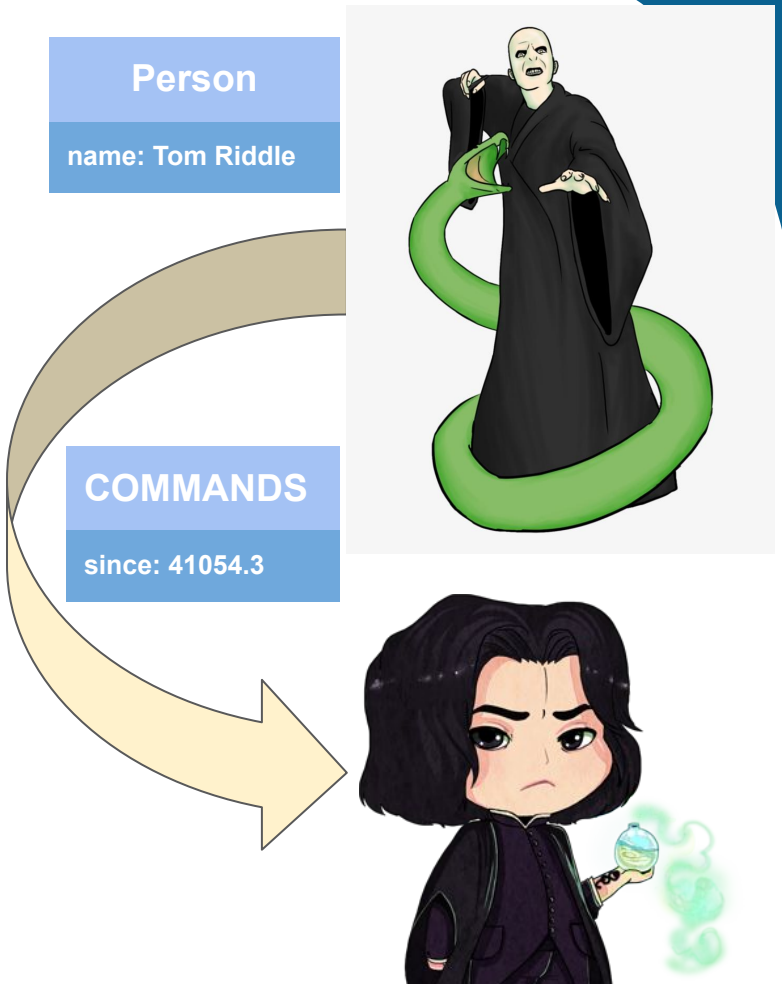
Node - vertex, entity, object, thing

Label - concept, class(ification) of a node

Relationship - edge, a physical link between two nodes

Type - class(ification) of a relationship

Property - key-value pair



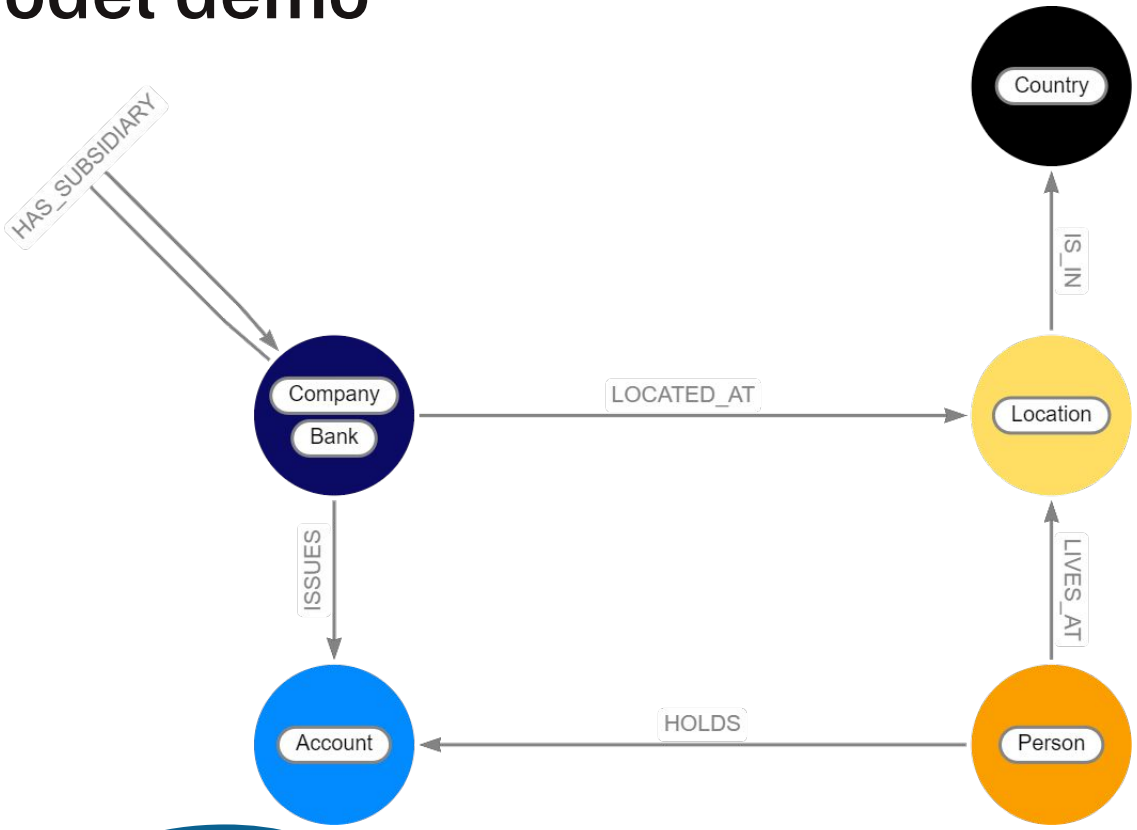
Demo(n)

Play along!

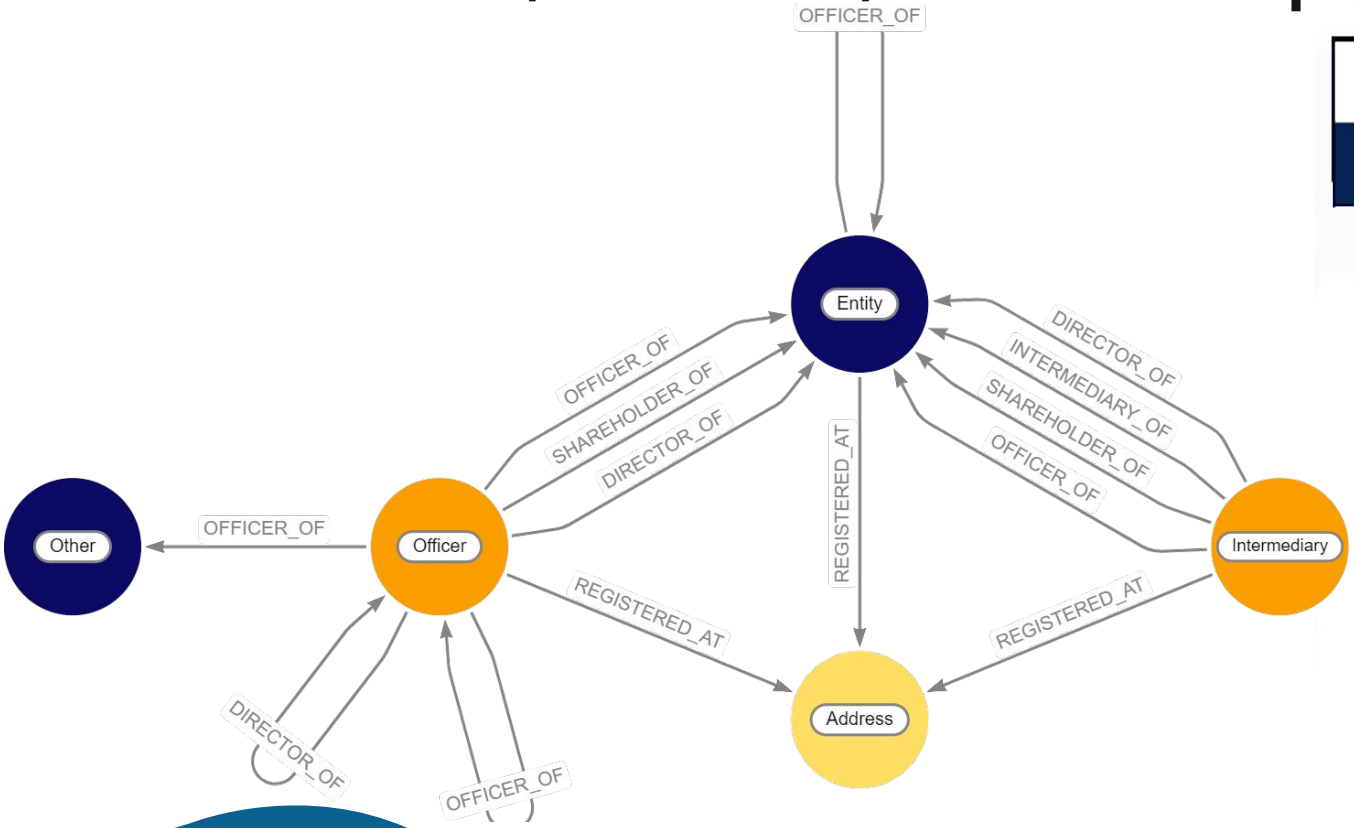
bit.ly/neo4junige

Drag unige-querying.csv into “Saved Cypher”.

Model demo



Model Panama/Paradise/Pandora Papers



Takeaways

Connected data = differentiator & disruptor
Google, Facebook, LinkedIn, …

A graph database helps make sense of connected data
The connections are *real*

It’s not about the complexity of the *model*, it’s about the complexity of the *query*

Questions?

I may have answers.

Neo4j in a

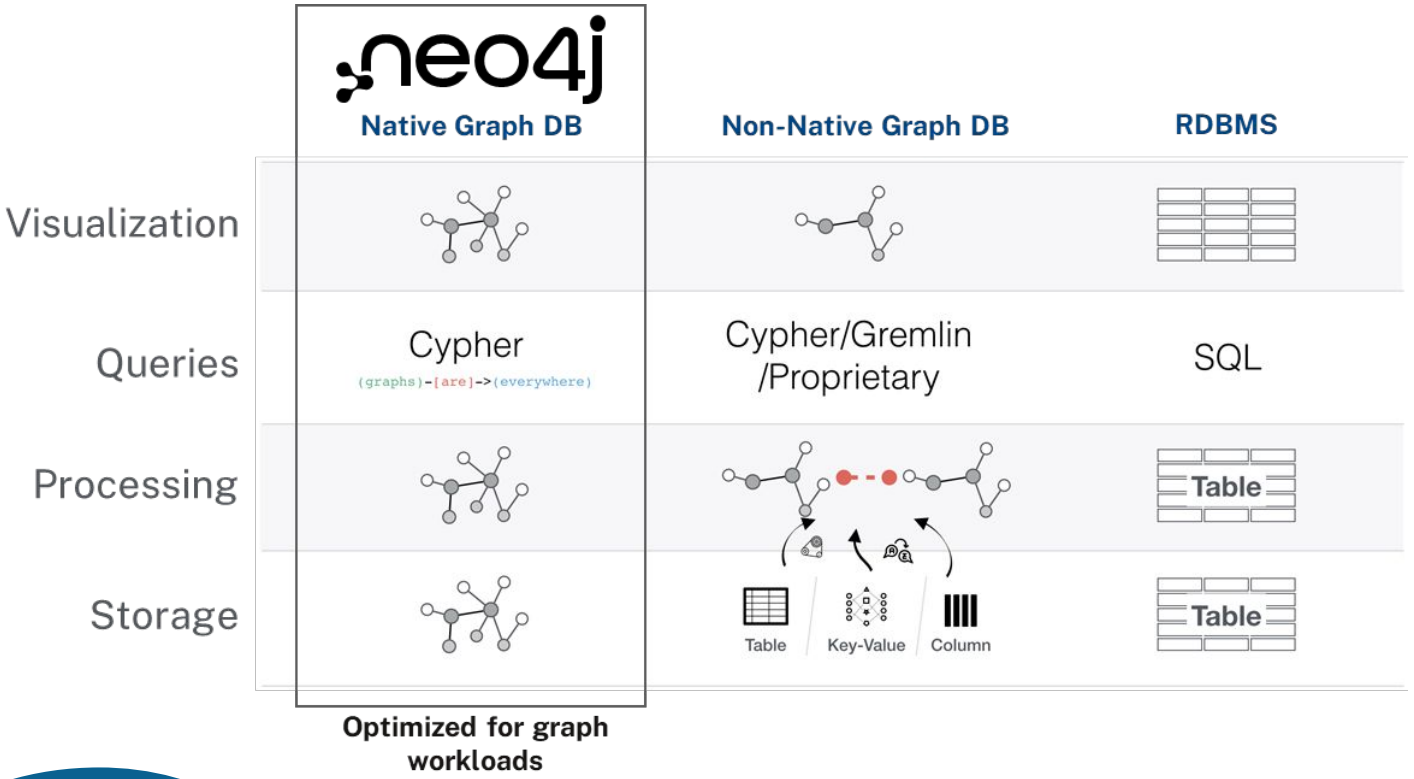
Secret Ingredients

Graph native

Mister Ping

Noodle Expert

Native versus non-native



Secret Ingredients

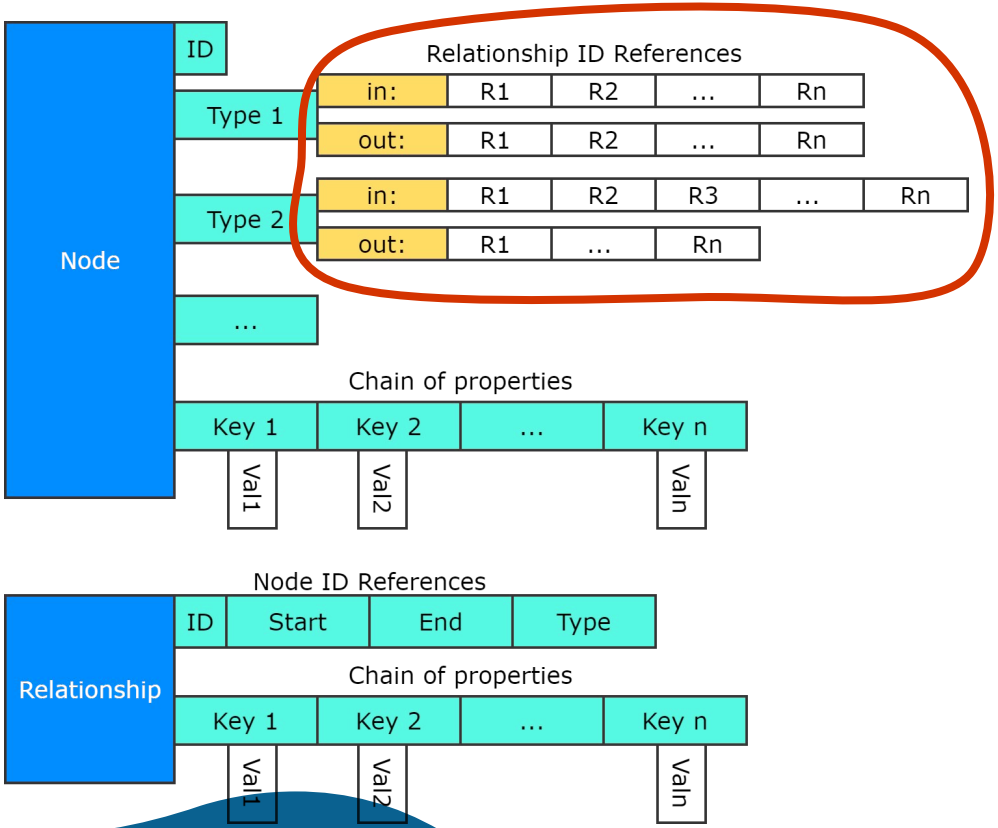
Graph native

Index-free adjacency

Mister Ping

Noodle Expert

Index-free adjacency



- Pointer hopping instead of index lookups
- Fixed size objects
- Joins are done on creation

Secret Ingredients

Graph native

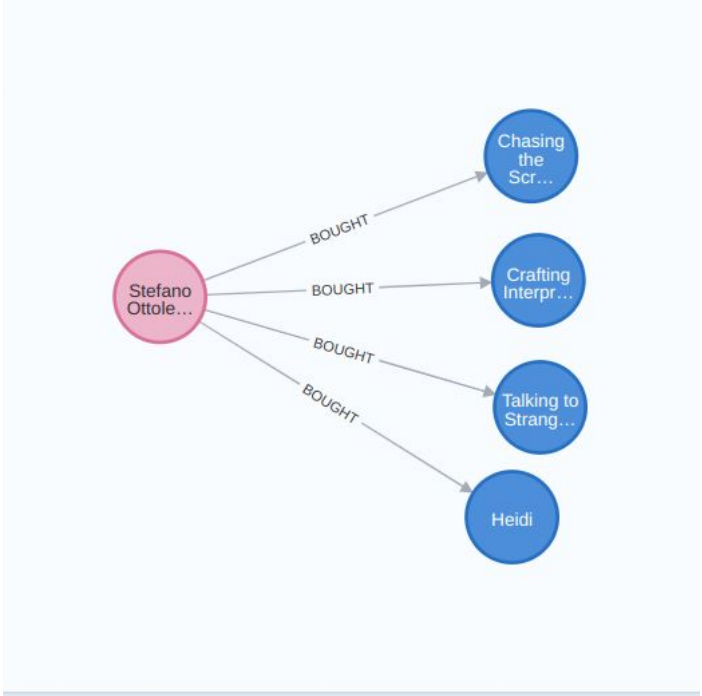
Index-free adjacency

Graph locality

Mister Ping

Noodle Expert

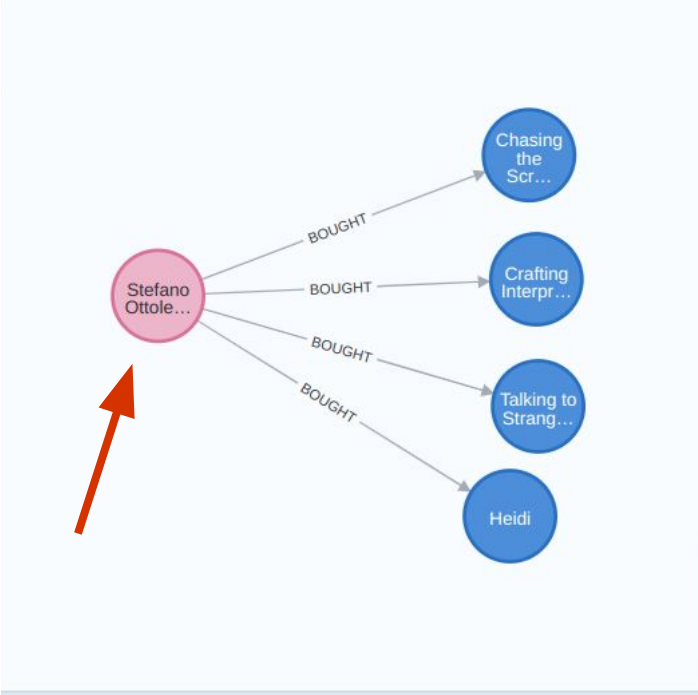
Graph locality



It doesn't matter if the data sits in a **huge** graph or a **small** graph.

A query performs in the same way (no big table problem) in all cases.

Gimme gimme gimme … an entrypoint



Relational

1 + 2x index lookups

O(n logn)

VS

Graph

1 index lookup

x pointer hops

O(1)

Downsides

slido

Name two possible downsides of the Neo4j Secret Ingredients

ⓘ Click **Present with Slido** or install our Chrome extension to activate this poll while presenting.

Use cases

Real-Time Recommendations

Fraud Detection

Tracking mail delivery

Graphs enabling glycoscience

Knowledge Graph Based Chatbot

Identity & Access Management

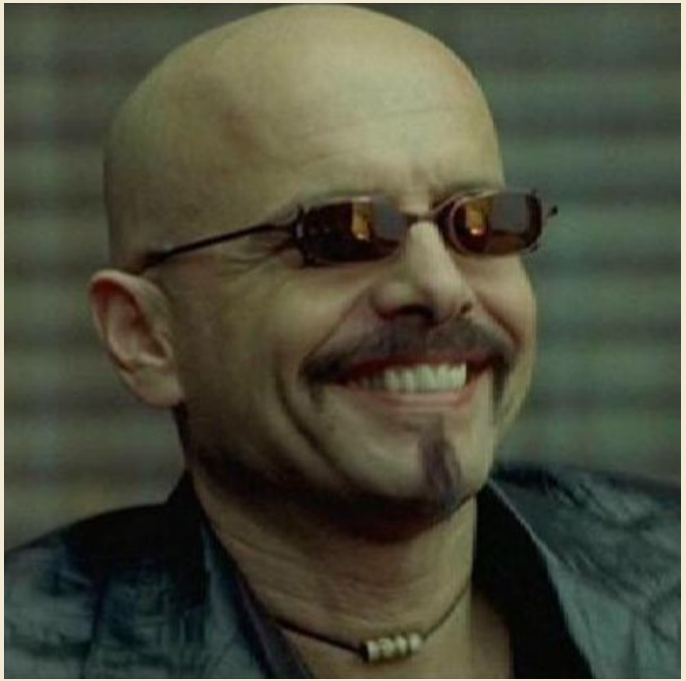
Questions?

I may have answers.

Cypher

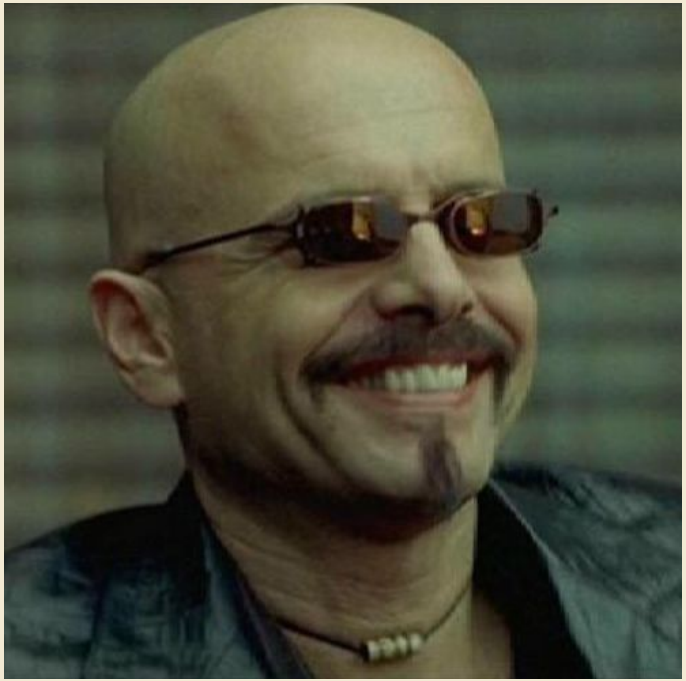
A new query language

Who is Cypher?



This is **Cypher**

Cypher ASCII art



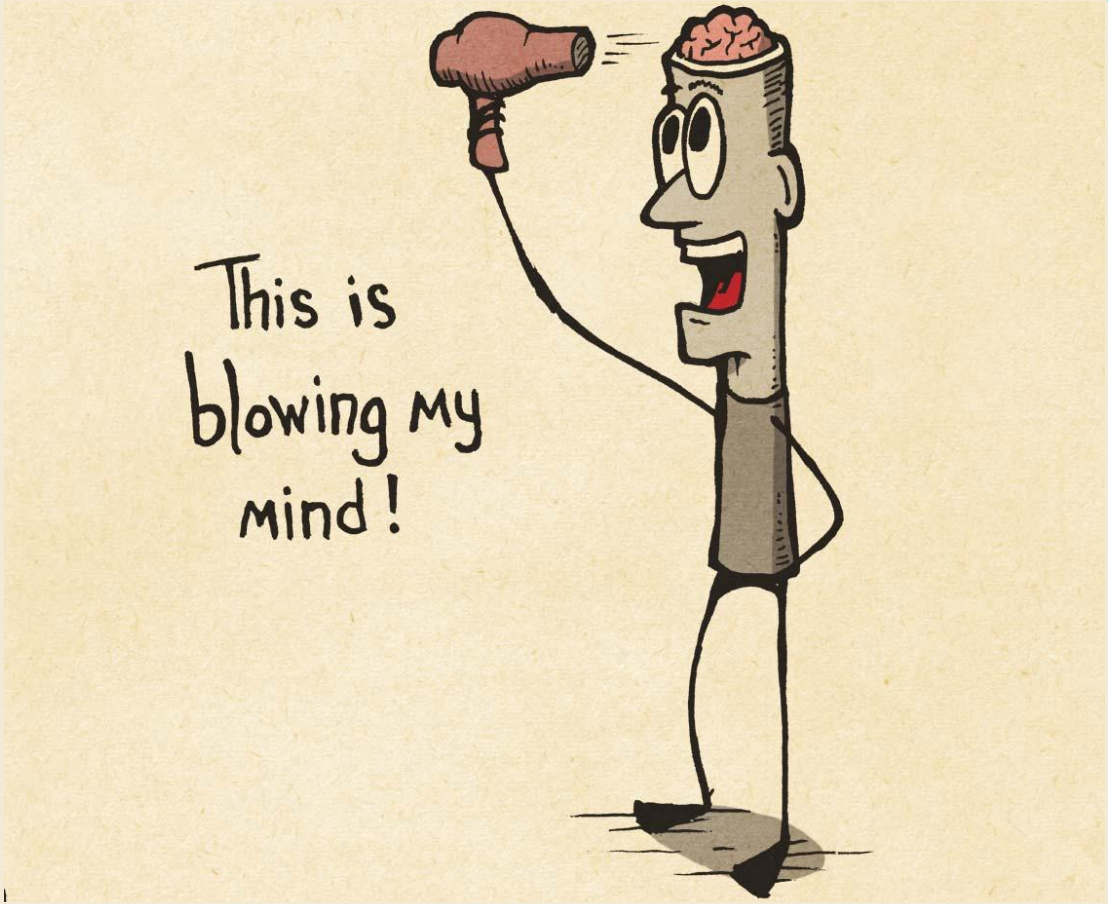
*This is **Cypher***

\+

```
("`-''-/").___..--''"`-._
 `6_ 6  )   `-.  (     ).`-.__.`)
 (_Y_.)'  ._   )  `._ `. ``-..-'
   _..`--'_..-_/  /--'_.'
  ((((.-''  ((((.'  (((.-'
```

*This is **ASCII Art***

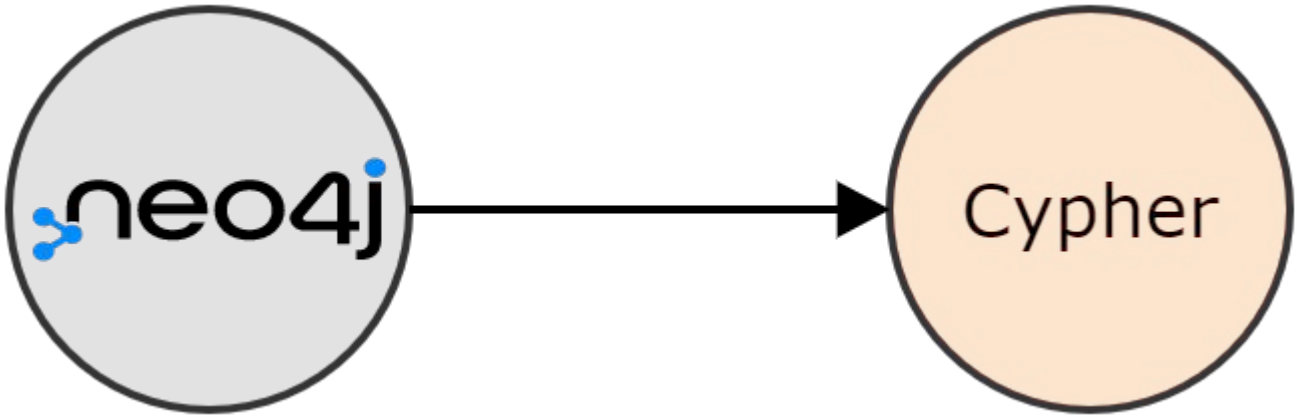
Cypher ASCII art



What is Cypher?

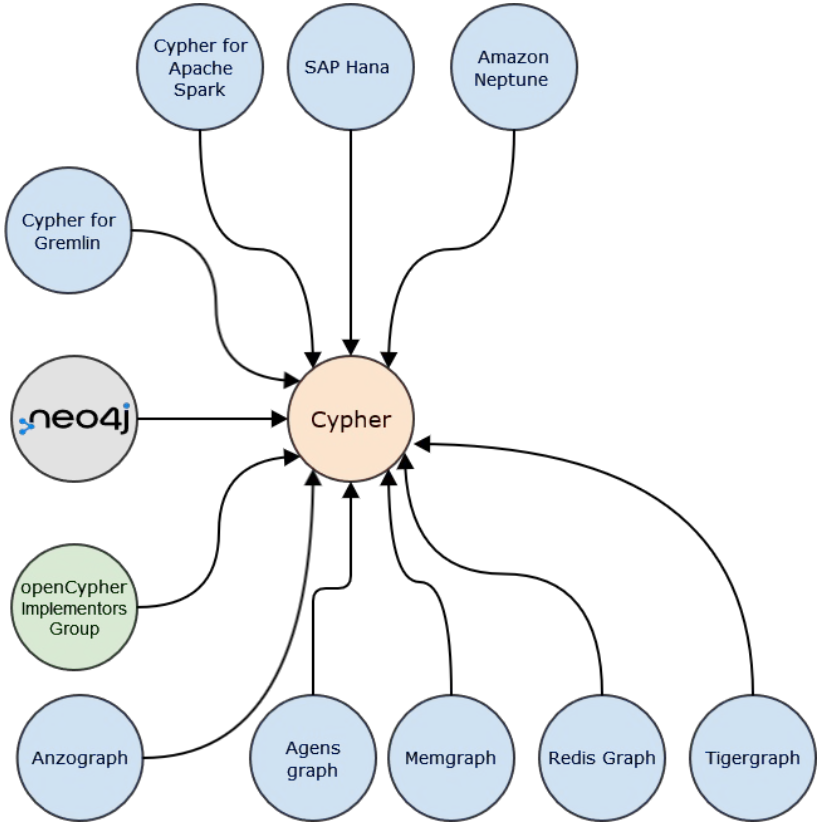
- Declarative (focus on *what*, not *how*) query language for property graphs
- Uses ASCII Art to visually describe patterns in a graph

Where is Cypher going?



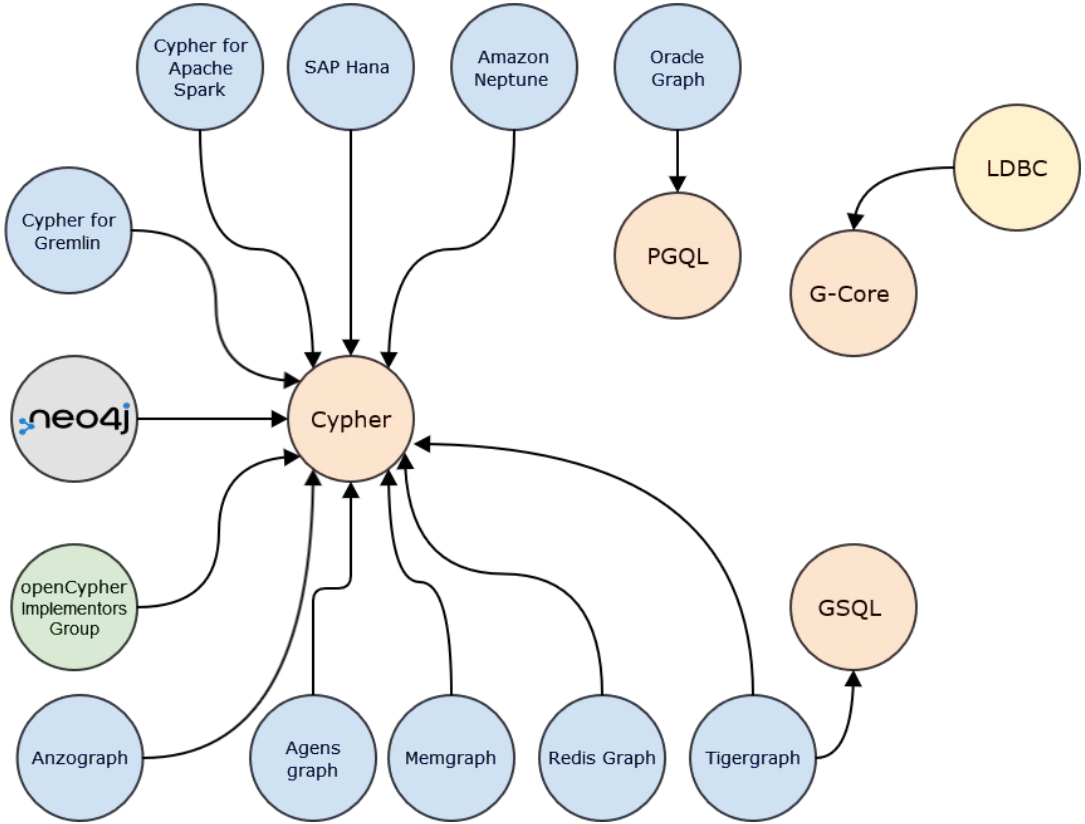
Where is Cypher going?

Cypher has been open from the start and has quite a few implementations!



Where is Cypher going?

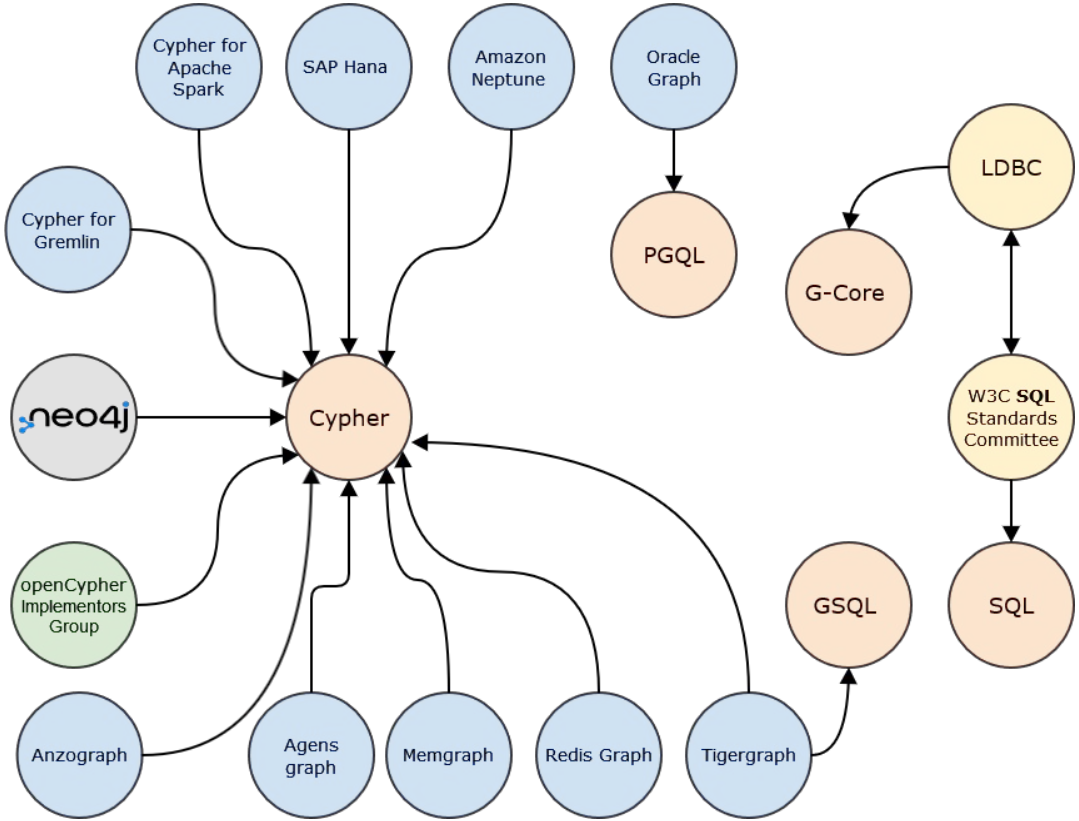
Realisation that a property graph requires a specific query language is much wider!



Where is Cypher going?

What does the Alpha & Omega of standards think?

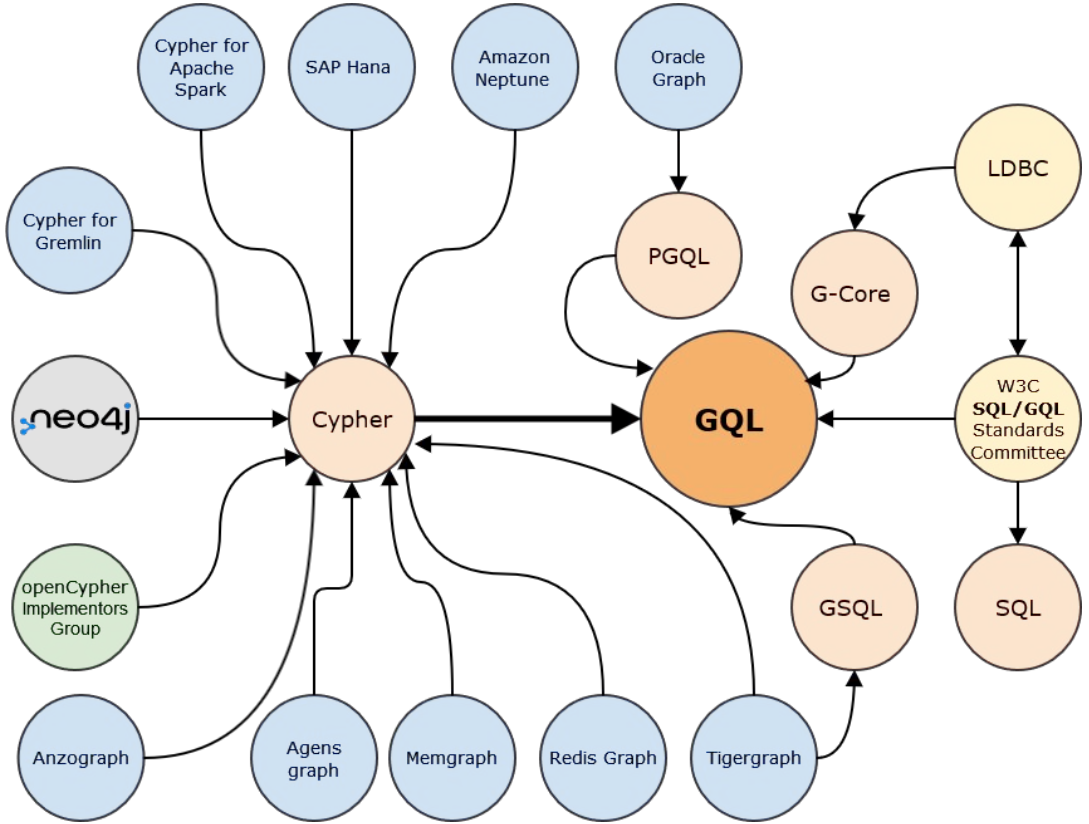
That committee hasn't budged in over three decades ...



Where is Cypher going?

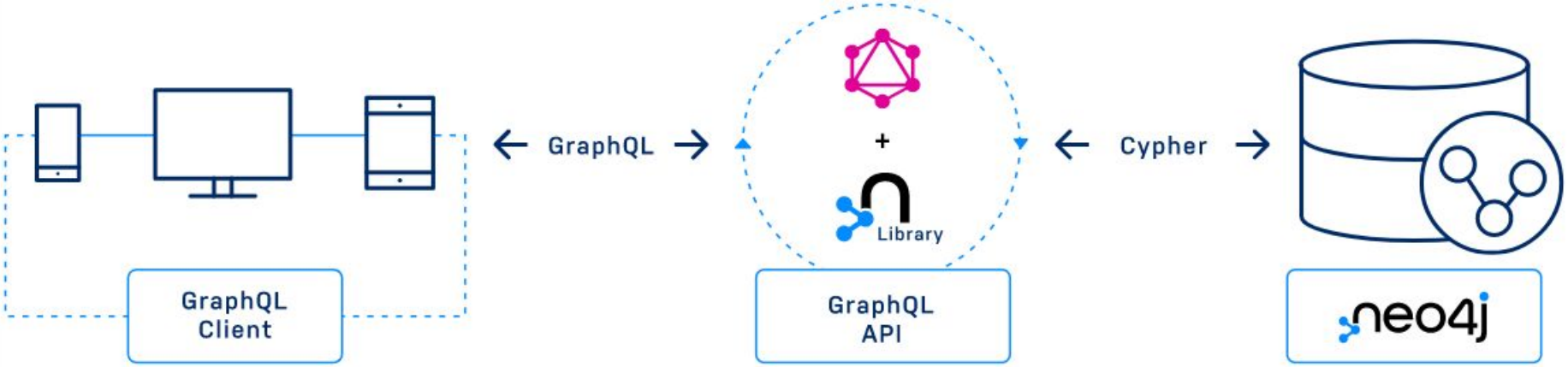
But it started moving in 2019, including an official renaming!

GQL is now an official ISO standard (in progress).



GQL ≠ GraphQL

Say the words and you'll hear the same thing twice …
but they couldn't be more different.



Play along!

Practice time!

- Clean your database
  MATCH (p) DETACH DELETE p

- Create a node representing yourself, with interests in at least 3 topics.
  No duplicate nodes allowed.
  …

- After doing it, you may want to check out the clause UNWIND, and adapt your query.

slido

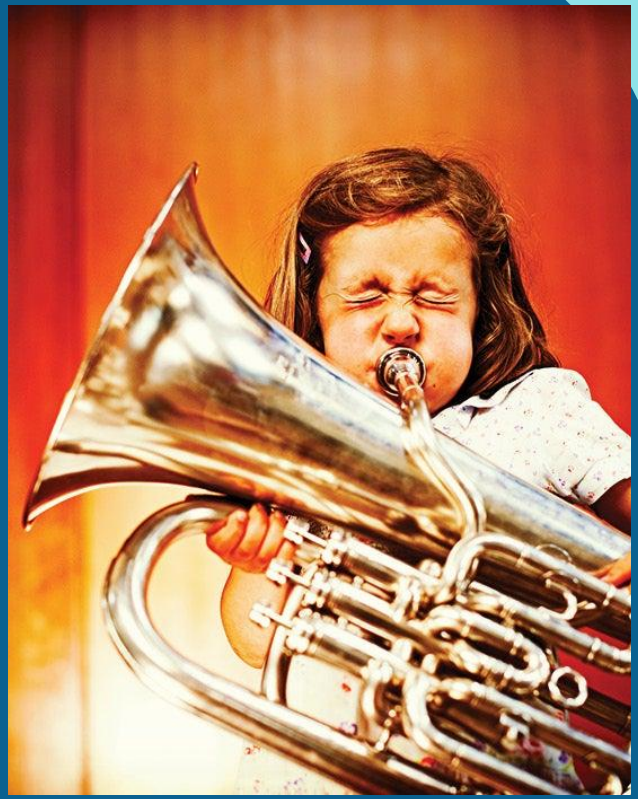
Now that you are old and wise, if you could say one single thing to your children, what would it be? What is your life lesson?

Cheaty solutions

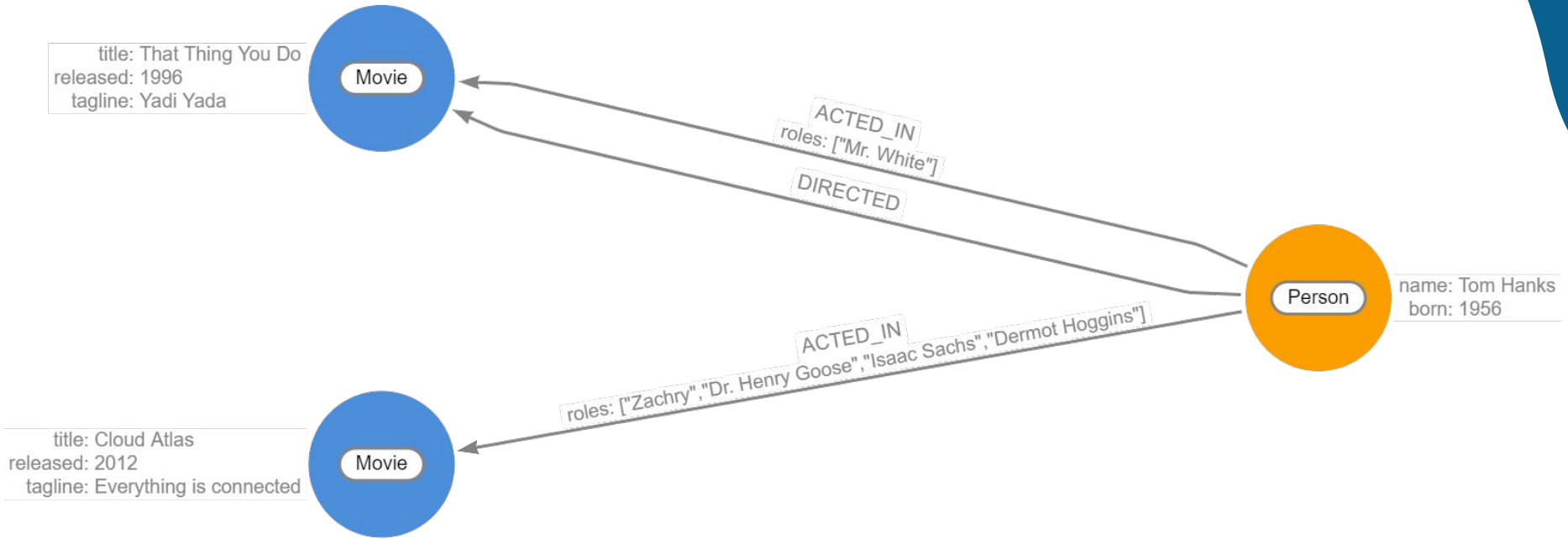
```
MERGE (me:Person {name:'Stefano'})
MERGE (t1:Topic {name:'Roller skating'})
MERGE (t2:Topic {name:'Acroyoga'})
MERGE (t3:Topic {name:'Rants'})
MERGE (me)-[:LIKES]->(t1)
MERGE (me)-[:LIKES]->(t2)
MERGE (me)-[:LIKES]->(t3)
RETURN me
```

```
MERGE (me:Person {name:'Stefano'})
WITH me AS me
UNWIND ['Roller skating', 'Acroyoga', 'Rants'] as interest
MERGE (t:Topic {name: interest})
MERGE (me)-[:LIKES]->(t)
RETURN me
```

More practice!



Movies model



MATCH Nodes

()

(:Person)

(:%)

(:!(Person)&(Actor|Actress))

(x WHERE x.name = "John Doe" AND x.age > 50)

(x:!(Person)&(Actor|Actress) WHERE x.name = "John Doe")

For an efficient pattern match, you should be as precise as possible!

Relationships – MATCH vs CREATE

- When CREATE-ing, you have to provide a direction…

  CREATE (me)-[:LIKES]->(topic)
  CREATE (me)<-[:LIKES]-(person)

- … but when MATCH-ing, you can ask for undirected!

  MATCH (me)-[:LIKES]->(topic)
  MATCH (me)-[:LIKES]-(person)

MATCH Relationships

()-[]-()

- No direction = either direction
- Can also be written as ()--()

Matches any single hop relationship between any two nodes.

MATCH Relationships

()-[]-()

()-[]->() (or ()<-[]-())

- Can also be written as ()-->()

Matches any single hop relationship between any two nodes.

MATCH Relationships

()-[]-()

()-[]->()

()-[:ACTED_IN]->()

Matches **any** single hop relationship between **any** two nodes filtering for a **specific** type of relationship.

MATCH Relationships

()-[]-()

()-[]->()

()-[:ACTED_IN]->()

()-[:ACTED_IN|DIRECTED]->()

Matches any single hop relationship between any two nodes filtering for several specific types of relationship.

MATCH Relationships

()-[]-()

()-[]->()

()-[:ACTED_IN]->()

()-[:ACTED_IN|DIRECTED]->()

()-[ai:ACTED_IN WHERE ai.roles = ["Herself"]]->()

Matches **any** single hop relationship between **any** two nodes filtering for a **specific type** of relationship and also filtering on the relationship property.

MATCH Relationships

There's more ...

```
MATCH sg=(b:Bank WHERE b.name =
"Fideuram")-[*1..3]-(c:Country WHERE c.name = "Panama")
RETURN sg;
```

Practice time!

1. Find 10 actor names.
2. Find how many movies have been released after 2005.
3. Create a WATCHED relationship between yourself and Cloud Atlas. Show this relationship.
4. Find who directed V for Vendetta.
5. Find people who have acted in movies released after 2005.
6. Find all people who have co-acted with Kevin Bacon (in any movie).
7. Find people 3 steps away from Tom Hanks.
8. Find (and show) the shortest path between Bill Paxton and Gary Sinise.

slido

If you had a magic wand, what is one thing you would like to make happen this saturday?

ⓘ Click **Present with Slido** or install our Chrome extension to activate this poll while presenting.

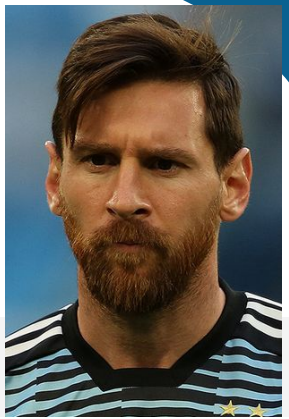
Cheaty solutions!

1. `MATCH (p:Person) RETURN p.name LIMIT 10`
2. `MATCH (m:Movie WHERE m.released > 2005) RETURN COUNT(*)`
   `MERGE (ste:Person {name: 'Stefano'})`
   `MERGE (film:Movie {title: 'Cloud Atlas'})`
   `MERGE sg=(ste)-[:WATCHED]->(film)`
   `RETURN sg`
3. `MATCH (p:Person)-[:DIRECTED]->(m:Movie {title:'V for Vendetta'}) RETURN p.name`
4. `MATCH (p:Person)-[:ACTED_IN]->(m:Movie WHERE m.released > 2005) RETURN p.name`
5. `MATCH sg=(p:Person)-[r:ACTED_IN]->(m:Movie)<-[:ACTED_IN]-(:Person {name: 'Kevin Bacon'}) RETURN sg`
6. `MATCH sg=(p:Person)-[*1..3]-(:Person {name: 'Tom Hanks'}) RETURN sg`
7. `MATCH sg=shortestPath((p:Person {name: 'Bill Paxton'})-[*]-(:Person {name: 'Gary Sinise'})) RETURN sg`

Interesting (?) queries...

This can get Messi ...

```
CREATE ()-[:DOESMORETHANMOSTTHINK]->();
```



Interesting (?) queries...

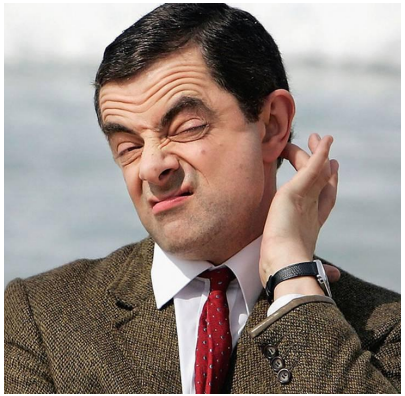
```
MERGE (p:Person
       {name: 'Stefano'})
MERGE (p:Topic
       {name: 'Rants'})
MERGE (p)-[:LIKES]->(t)
```

VS

```
MATCH  (p:Person
        {name: 'Fabio'})
MERGE (p:Topic
       {name: 'Acroyoga'})
MERGE (p)-[:LIKES]->(t)
```

No error! Cypher queries are *pipelines*!

SQL query – Interpreted as a whole.
Cypher query – If at any stage the number of results drops to zero, the remainder of the pipeline is *not* executed.



Interesting (?) queries...

What is being counted here?

```
MATCH (ste:Person WHERE tom.name = "Stefano
Hanks")-[ai:ACTED_IN]->(m:Movie)
RETURN count(ste);
```

Patterns. It counts matching patterns. Try with count(ai), count(m) or count(*), it makes absolutely no difference. It. Counts. Matching. Patterns.

What is this query doing?

```
MATCH (m:Member WHERE m.name = "Stefano")-[:MEMBER_OF]->
(g:Group)-[:HAS_TOPIC]->(t:Topic)
WHERE NOT EXISTS ((m)-[:INTERESTED_IN]->(t))
WITH m, t, count(*) AS weight
WHERE weight > 2
CREATE (m)-[:INTERESTED_IN]->(t);
```

It infers a relationship - i.e. gives a recommendation.

Magical Mr White

Magical Mr White

How many results does this query return?

```
MATCH ()-[x:ACTED_IN WHERE x.roles = ["Mr. White"]]-()
RETURN x;
```

*Hint: I did not lie, there is only **one** relationship containing ["Mr. White"] as the roles property in the database ...*

slido

How many results does the query `MATCH ()-[x:ACTED_IN WHERE x.roles = ["Mr. White"]]-() RETURN x;` return?

ⓘ Click **Present with Slido** or install our Chrome extension to activate this poll while presenting.

Questions?

I may have answers.

