

CC5212-1

PROCESAMIENTO MASIVO DE DATOS

OTOÑO 2021

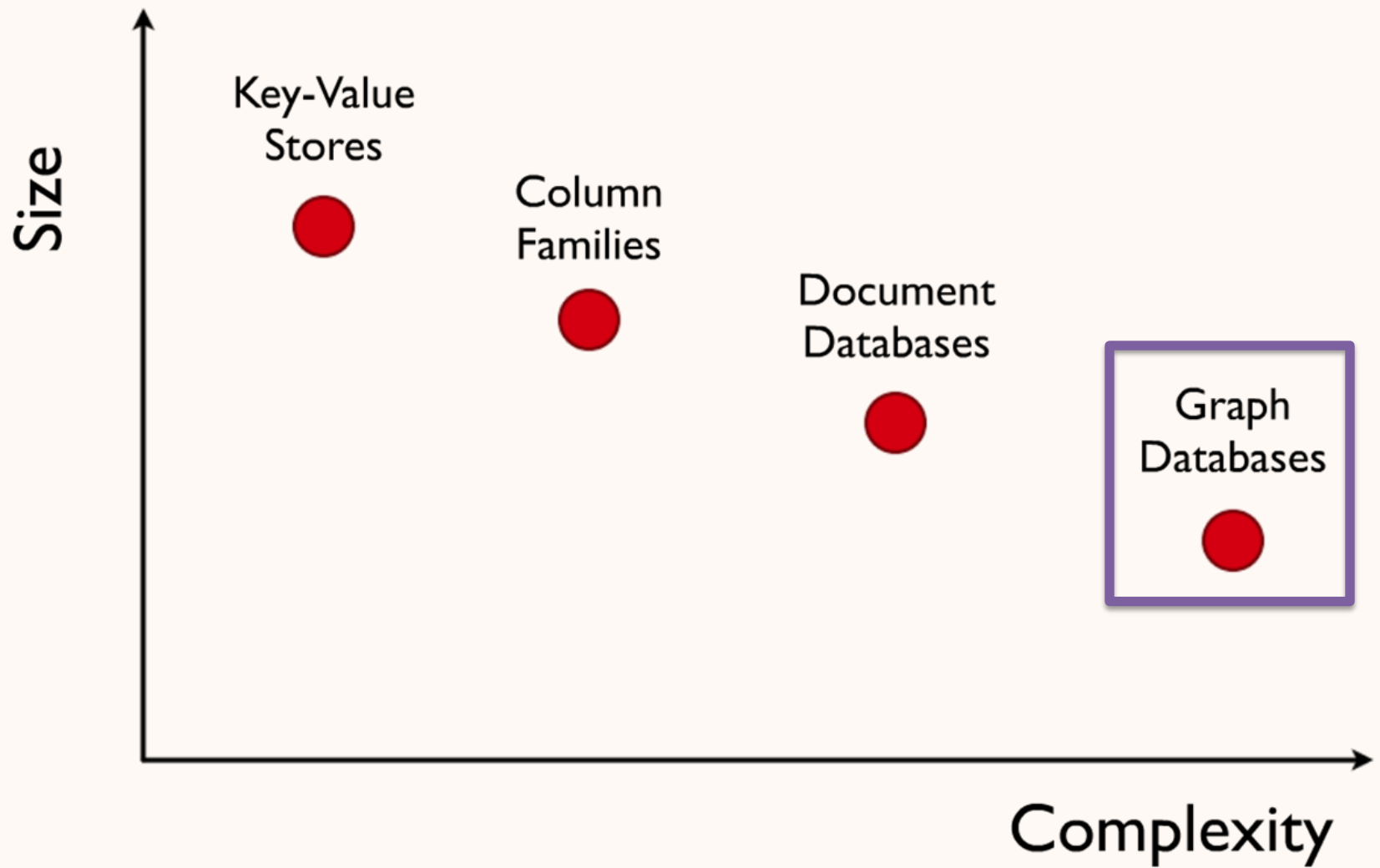
Lecture 11

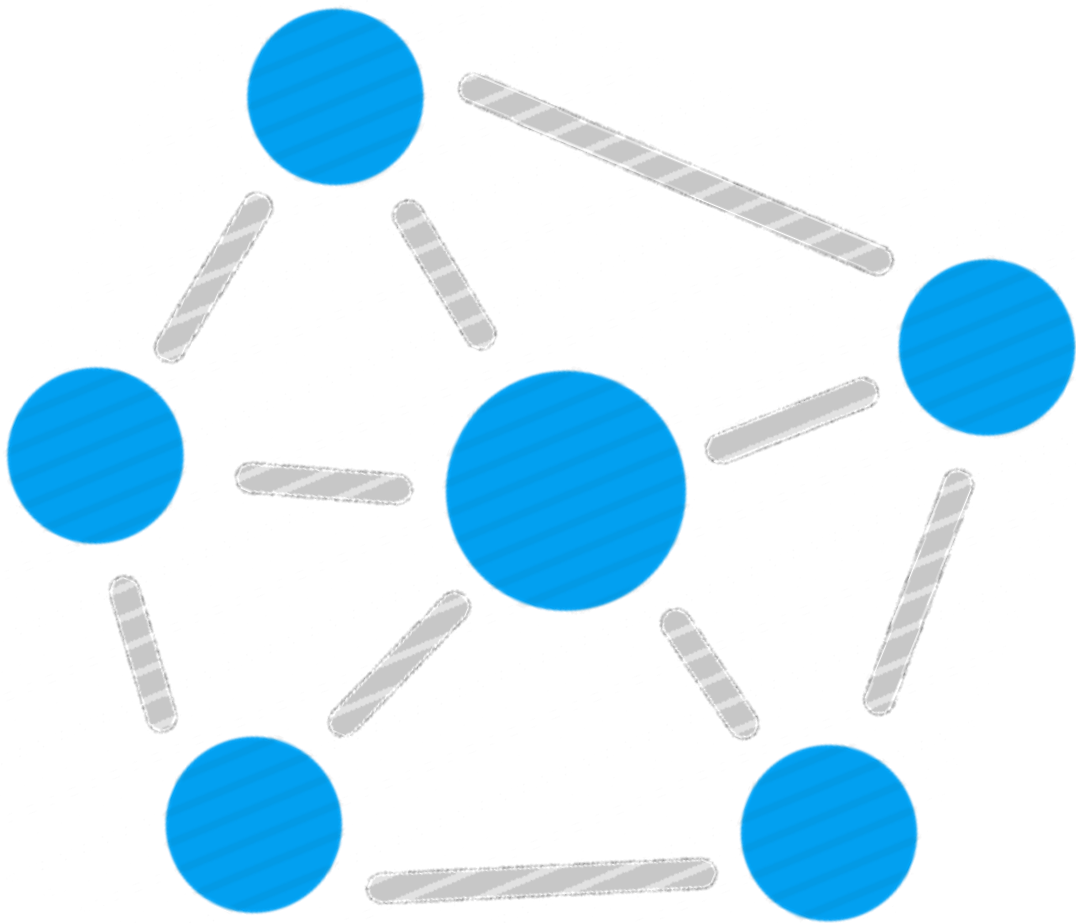
NoSQL: Neo4J

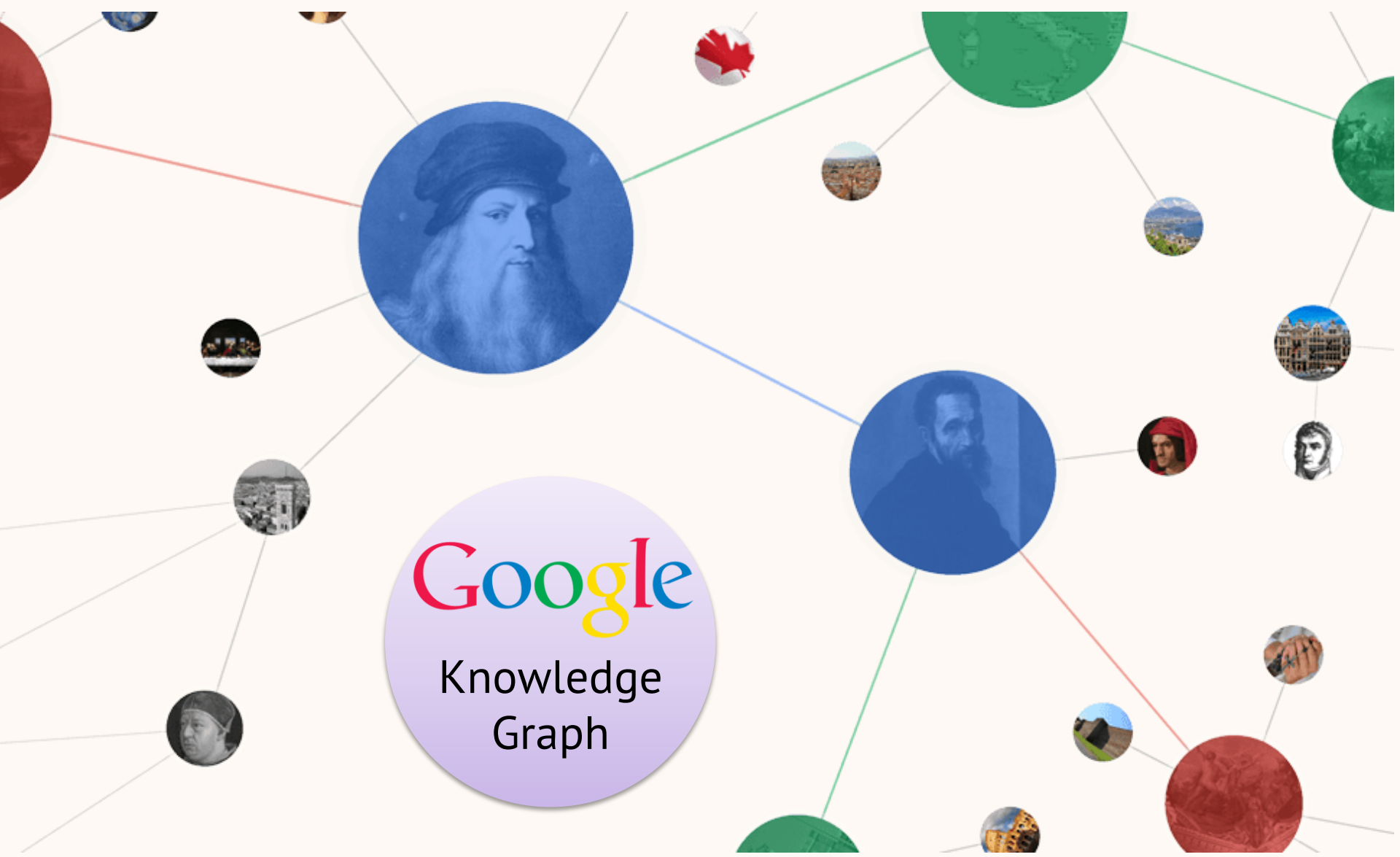
Aidan Hogan

aidhog@gmail.com

NoSQL



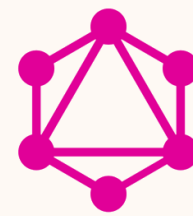






Facebook Open Graph

Thinking in Graphs



GraphQL

It's Graphs All the Way Down *

With GraphQL, you model your business domain as a graph

Graphs are powerful tools for modeling many real-world phenomena because they resemble our natural mental models and verbal descriptions of the underlying process. With GraphQL, you model your business domain as a graph by defining a schema; within your schema, you define different types of nodes and how they connect/relate to one another. On the client, this creates a pattern similar to Object-Oriented Programming: types that reference other types. On the server, since GraphQL only defines the interface, you have the freedom to use it with any backend (new or legacy!).

Shared Language

Naming things is a hard but important part of building intuitive APIs

Think of your GraphQL schema as an expressive shared language for your team and your users. To build a good schema, examine the everyday language you use to describe your business. For example, let's try to describe an email app in plain english:



Amazon Neptune

Fast, reliable graph database built for the cloud

Get started with Amazon Neptune

● Graph database
Topic

+ Compare

United States ▼

2004 - present ▼

All categories ▼

Web Search ▼

Interest over time ⓘ



● Graph database
Topic

● Relational database
Topic

+ Add comparison

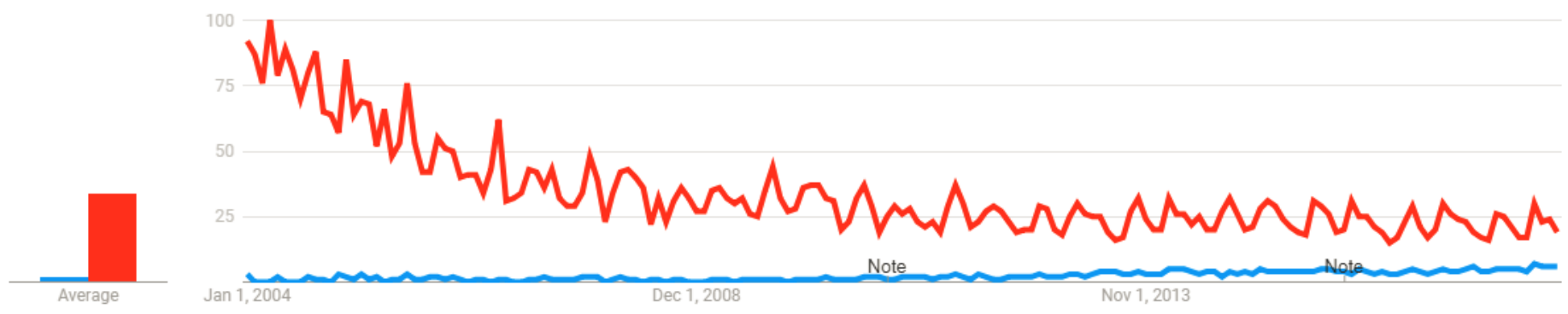
United States ▼

2004 - present ▼

All categories ▼

Web Search ▼

Interest over time ⓘ



● Graph database
Topic

+ Compare

United States ▼

2004 - present ▼

All categories ▼

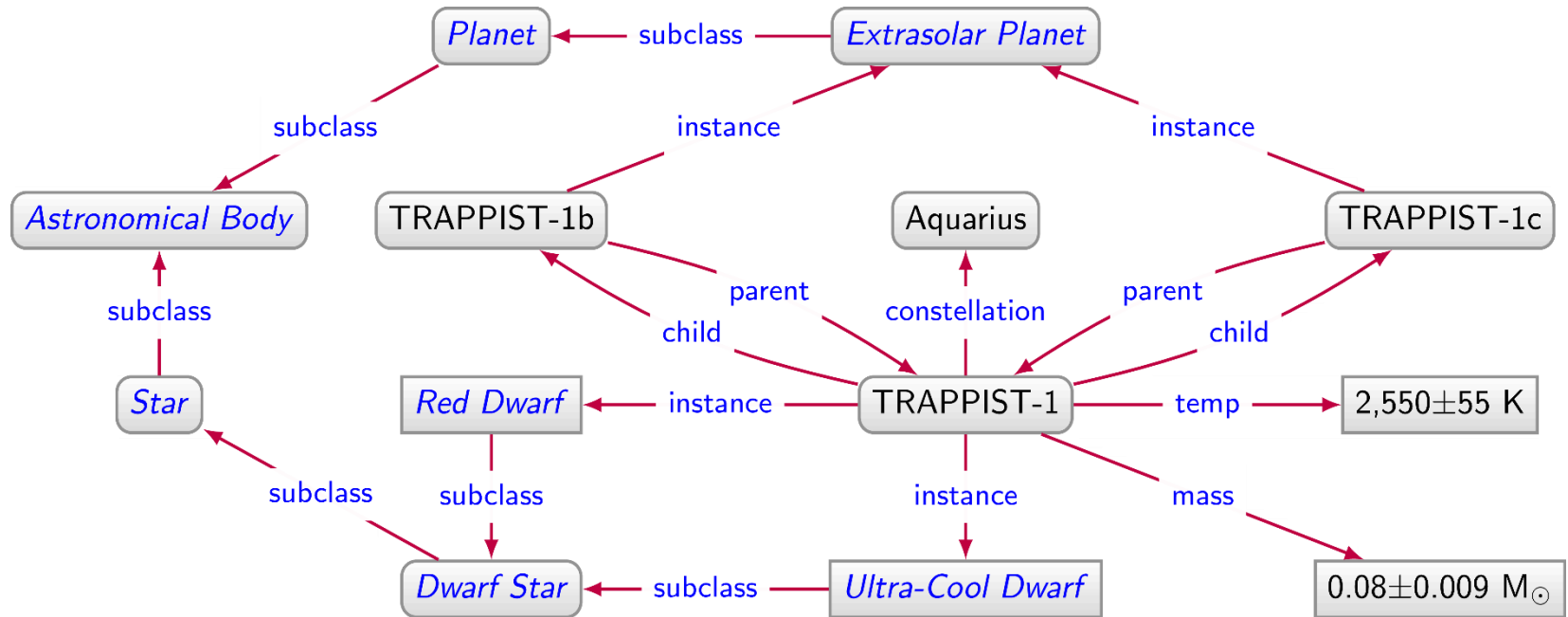
Web Search ▼

Interest over time ⓘ



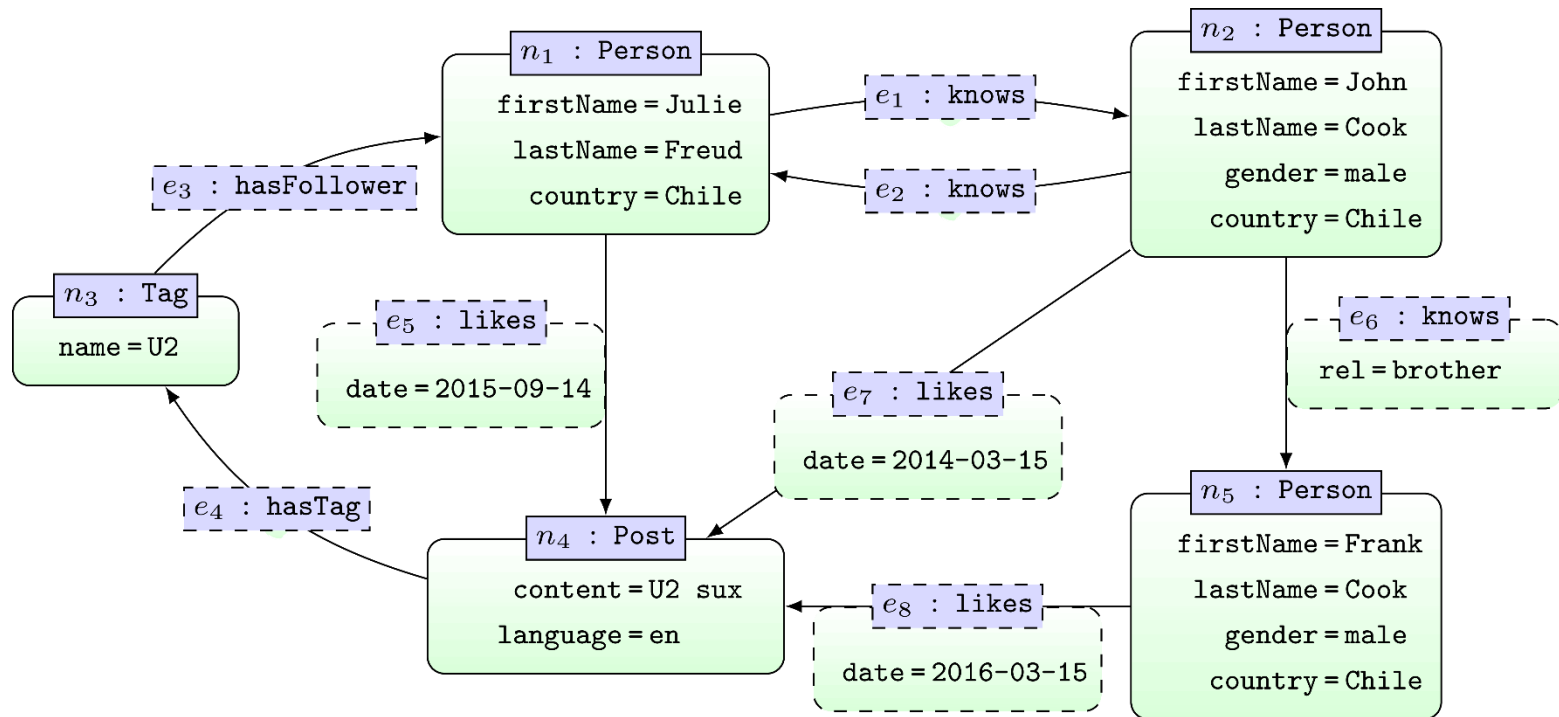
WHAT IS A GRAPH DATABASE?

Directed Edge-labelled Graph



```
SELECT ?const (COUNT(DISTINCT ?body) AS ?num)
WHERE {
  ?body :instance/:subclass* :AstronomicalBody .
  ?body :parent?/:constellation ?const .
}
GROUP BY ?const
ORDER BY DESC(?num)
```

Property Graph



```
MATCH (x1:Person {firstName:"Julie"})-[:knows*]->(x2:Person)
MATCH (x2)-[:likes]->()-[:hasTag]->()-[:hasFollower]->(x1)
RETURN x2.firstName
```

WHY DO WE NEED GRAPH DATABASES?

WHY DO WE NEED GRAPH DATABASES?

FLEXIBILITY

Relational Databases ...



Relational Databases ...

Debit

<u>account</u>	<u>comment</u>	<u>date</u>	<u>time</u>	<u>amount</u>	<u>total</u>	<u>id</u>
7873698669	Initial deposit	2020-21-01	20:02:02	300000	300000	TRCXGU8JSHD
7873698669	C0°0°L Designs	2020-02-06	09:15:33	50000	325000	TRCCIA2J8A0

Credit

<u>account</u>	<u>comment</u>	<u>date</u>	<u>time</u>	<u>amount</u>	<u>total</u>	<u>id</u>
7873698669	Electricity	2020-02-02	20:00:01	8200	291800	TRCJASJDA9A
7873698669	Heat	2020-02-02	20:00:02	600	291200	TRC81KAQWAS
7873698669	Moviestar	2020-02-02	20:00:03	16200	275000	TRCK8J7JA8D
7873698669	ATM	2020-02-08	16:05:02	100000	225000	TRCPM8A45AD

Account

<u>number</u>	<u>rut</u>	<u>type</u>	<u>total_clp</u>	<u>total_usd</u>
7873698669	32.000.273-K	Current	225000	344,94

Exchange

<u>c1</u>	<u>c2</u>	<u>value</u>
CLP	USD	0,0001533
USD	CLP	652,2750000

Client

<u>rut</u>	<u>name</u>	<u>phone</u>	<u>address</u>
32.000.273-K	Kelvin	+56976698463	Campo de Hielo Sur, Depto 273

Planets / Relational Database



Planets / Relational Database

Planet

name

Mercury

Venus

Earth

Mars

Jupiter

Saturn

Uranus

Neptune

Pluto

Planets / Relational Database

Planet	
<u>name</u>	<u>dist</u>
Mercury	
Venus	
Earth	1.00
Mars	
Jupiter	
Saturn	
Uranus	
Neptune	
Pluto	

Planets / Relational Database

Planet	
<u>name</u>	<u>dist</u>
Mercury	0.39
Venus	0.72
Earth	1.00
Mars	1.52
Jupiter	
Saturn	
Uranus	
Neptune	
Pluto	49.31

Planets / Relational Database

Planet		
<u>name</u>	dist	radius
Mercury	0.39	0.38
Venus	0.72	
Earth	1.00	1.00
Mars	1.52	0.53
Jupiter		10.97
Saturn	9.54	
Uranus	19.19	3.98
Neptune		
Pluto	49.31	

Planets / Relational Database

Planet							
<u>name</u>	dist	radius	grav	days	years	temp	ring
Mercury	0.39	0.38	2.8	58.646	0.241	440	false
Venus	0.72	0.95	8.9	-243.019	0.615	730	false
Earth	1.00	1.00	9.8	0.997	1.000	288	false
Mars	1.52	0.53	3.7	1.026	1.880	186	false
Jupiter	5.20	10.97	22.9	0.414	11.862	152	true
Saturn	9.54	9.14	9.1	0.444	29.447	134	true
Uranus	19.19	3.98	7.8	-0.719	84.017	76	true
Neptune	30.07	3.86	11.0	0.671	164.791	53	true
Pluto	49.31	0.19	0.063	6.39	248.000	44	false

Planets / Relational Database



Planet							
<u>name</u>	<u>dist</u>	<u>radius</u>	<u>grav</u>	<u>days</u>	<u>years</u>	<u>temp</u>	<u>ring</u>
Mercury	0.39	0.38	2.8	58.646	0.241	440	false
Venus	0.72	0.95	8.9	-243.019	0.615	730	false
Earth	1.00	1.00	9.8	0.997	1.000	288	false
Mars	1.52	0.53	3.7	1.026	1.880	186	false
Jupiter	5.20	10.97	22.9	0.414	11.862	152	true
Saturn	9.54	9.14	9.1	0.444	29.447	134	true
Uranus	19.19	3.98	7.8	-0.719	84.017	76	true
Neptune	30.07	3.86	11.0	0.671	164.791	53	true
Pluto	49.31	0.19	0.063	6.39	248.000	44	false

Planets / Relational Database



Planet								
<u>name</u>	dist	radius	grav	days	years	temp	ring	moon
Mercury	0.39	0.38	2.8	58.646	0.241	440	false	⊥
Venus	0.72	0.95	8.9	-243.019	0.615	730	false	⊥
Earth	1.00	1.00	9.8	0.997	1.000	288	false	Luna
Mars	1.52	0.53	3.7	1.026	1.880	186	false	Phobos, Deimos
Jupiter	5.20	10.97	22.9	0.414	11.862	152	true	Callisto, Ganymede, ...
Saturn	9.54	9.14	9.1	0.444	29.447	134	true	Titan, Rhea, ...
Uranus	19.19	3.98	7.8	-0.719	84.017	76	true	Oberon, Titania, ...
Neptune	30.07	3.86	11.0	0.671	164.791	53	true	Triton, ...
Pluto	49.31	0.19	0.063	6.39	248.000	44	false	Charon

Planets / Relational Database



Planet							
<u>name</u>	dist	radius	grav	days	years	temp	ring
Mercury	0.39	0.38	2.8	58.646	0.241	440	false
Venus	0.72	0.95	8.9	-243.019	0.615	730	false
Earth	1.00	1.00	9.8	0.997	1.000	288	false
Mars	1.52	0.53	3.7	1.026	1.880	186	false
Jupiter	5.20	10.97	22.9	0.414	11.862	152	true
Saturn	9.54	9.14	9.1	0.444	29.447	134	true
Uranus	19.19	3.98	7.8	-0.719	84.017	76	true
Neptune	30.07	3.86	11.0	0.671	164.791	53	true
Pluto	49.31	0.19	0.063	6.39	248.000	44	false

Moon	
<u>name</u>	planet
Ganimesdes	Jupiter
Calisto	Jupiter
Europa	Jupiter
Io	Jupiter
Titan	Saturn
Triton	Neptune
Luna	Terra
Oberon	Uranus
Charon	Pluto
...	...

Planets / Relational Database



Planet							
<u>name</u>	dist	radius	grav	days	years	temp	ring
Mercury	0.39	0.38	2.8	58.646	0.241	440	false
Venus	0.72	0.95	8.9	-243.019	0.615	730	false
Earth	1.00	1.00	9.8	0.997	1.000	288	false
Mars	1.52	0.53	3.7	1.026	1.880	186	false
Jupiter	5.20	10.97	22.9	0.414	11.862	152	true
Saturn	9.54	9.14	9.1	0.444	29.447	134	true
Uranus	19.19	3.98	7.8	-0.719	84.017	76	true
Neptune	30.07	3.86	11.0	0.671	164.791	53	true
Pluto	49.31	0.19	0.063	6.39	248.000	44	false

Moon			
<u>name</u>	planet	discoverer	year
Ganymedes	Jupiter	Galileo Galilei	1610
Calisto	Jupiter	Galileo Galilei	1610
Europa	Jupiter	Galileo Galilei	1610
Io	Jupiter	Galileo Galilei	1610
Titan	Saturn	Christiaan Huygens	1655
Triton	Neptune	William Lassell	1846
Luna	Terra	⊥	⊥
Oberon	Uranus	William Herschel	1787
Charon	Pluto	⊥	1978
...

Planets / Relational Database



Planet							
<u>name</u>	dist	radius	grav	days	years	temp	ring
Mercury	0.39	0.38	2.8	58.646	0.241	440	false
Venus	0.72	0.95	8.9	-243.019	0.615	730	false
Earth	1.00	1.00	9.8	0.997	1.000	288	false
Mars	1.52	0.53	3.7	1.026	1.880	186	false
Jupiter	5.20	10.97	22.9	0.414	11.862	152	true
Saturn	9.54	9.14	9.1	0.444	29.447	134	true
Uranus	19.19	3.98	7.8	-0.719	84.017	76	true
Neptune	30.07	3.86	11.0	0.671	164.791	53	true
Pluto	49.31	0.19	0.063	6.39	248.000	44	false

Moon	
<u>name</u>	<u>planet</u>
Ganimesdes	Jupiter
Calisto	Jupiter
Europa	Jupiter
Io	Jupiter
Titan	Saturn
Triton	Neptune
Luna	Terra
Oberon	Uranus
Charon	Pluto
...	...

MoonDiscoverer	
<u>name</u>	<u>discoverer</u>
Ganimesdes	Galileo Galilei
Calisto	Galileo Galilei
Europa	Galileo Galilei
Io	Galileo Galilei
Titan	Christiaan Huygens
Triton	William Lassell
Oberon	William Herschel
...	...

MoonDiscYear	
<u>name</u>	<u>year</u>
Ganimesdes	1610
Calisto	1610
Europa	1610
Io	1610
Titan	1655
Triton	1846
Oberon	1787
Charon	1978
...	...

Planets / Relational Database



Planet							
<u>name</u>	dist	radius	grav	days	years	temp	ring
Mercury	0.39	0.38	2.8	58.646	0.241	440	false
Venus	0.72	0.95	8.9	-243.019	0.615	730	false
Earth	1.00	1.00	9.8	0.997	1.000	288	false
Mars	1.52	0.53	3.7	1.026	1.880	186	false
Jupiter	5.20	10.97	22.9	0.414	11.862	152	true
Saturn	9.54	9.14	9.1	0.444	29.447	134	true
Uranus	19.19	3.98	7.8	-0.719	84.017	76	true
Neptune	30.07	3.86	11.0	0.671	164.791	53	true
Pluto	49.31	0.19	0.063	6.39	248.000	44	false

Moon	
<u>name</u>	planet
Ganimesdes	Jupiter
Calisto	Jupiter
Europa	Jupiter
Io	Jupiter
Titan	Saturn
Triton	Neptune
Luna	Terra
Oberon	Uranus
Charon	Pluto
...	...

MoonDiscoverer	
<u>name</u>	discoverer
Ganimesdes	Galileo Galilei
Calisto	Galileo Galilei
Europa	Galileo Galilei
Io	Galileo Galilei
Titan	Christiaan Huygens
Triton	William Lassell
Oberon	William Herschel
...	...

MoonDiscYear	
<u>name</u>	year
Ganimesdes	1610
Calisto	1610
Europa	1610
Io	1610
Titan	1655
Triton	1846
Oberon	1787
Charon	1978
...	...

Planets / Relational Database

Planet							
<u>name</u>	dist	radius	grav	days	years	temp	ring
Mercury	0.39	0.38	2.8	58.646	0.241	440	false
Venus	0.72	0.95	8.9	-243.019	0.615	730	false
Earth	1.00	1.00	9.8	0.997	1.000	288	false
Mars	1.52	0.53	3.7	1.026	1.880	186	false
Jupiter	5.20	10.97	22.9	0.414	11.862	152	true
Saturn	9.54	9.14	9.1	0.444	29.447	134	true
Uranus	19.19	3.98	7.8	-0.719	84.017	76	true
Neptune	30.07	3.86	11.0	0.671	164.791	53	true
Pluto	49.31	0.19	0.063	6.39	248.000	44	false

Moon	
<u>name</u>	P.name
Ganimesdes	Jupiter
Calisto	Jupiter
Europa	Jupiter
Io	Jupiter
Titan	Saturn
Triton	Neptune
Luna	Earth
Oberon	Uranus
Charon	Pluto
...	...

MoonDiscoverer	
<u>name</u>	discoverer
Ganimesdes	Galileo Galilei
Calisto	Galileo Galilei
Europa	Galileo Galilei
Io	Galileo Galilei
Titan	Christiaan Huygens
Triton	William Lassell
Oberon	William Herschel
...	...

MoonDiscYear	
<u>name</u>	year
Ganimesdes	1610
Calisto	1610
Europa	1610
Io	1610
Titan	1655
Triton	1846
Oberon	1787
Charon	1978
...	...

Planets / Relational Database



Planet							
<u>name</u>	<u>dist</u>	<u>radius</u>	<u>grav</u>	<u>days</u>	<u>years</u>	<u>temp</u>	<u>ring</u>
Mercury	0.39	0.38	2.8	58.646	0.241	440	false
Venus	0.72	0.95	8.9	-243.019	0.615	730	false
Earth	1.00	1.00	9.8	0.997	1.000	288	false
Mars	1.52	0.53	3.7	1.026	1.880	186	false
Jupiter	5.20	10.97	22.9	0.414	11.862	152	true
Saturn	9.54	9.14	9.1	0.444	29.447	134	true
Uranus	19.19	3.98	7.8	-0.719	84.017	76	true
Neptune	30.07	3.86	11.0	0.671	164.791	53	true
Pluto	49.31	0.19	0.063	6.39	248.000	44	false

Moon	
<u>name</u>	<u>P.name</u>
Ganimesdes	Jupiter
Calisto	Jupiter
Europa	Jupiter
Io	Jupiter
Titan	Saturn
Triton	Neptune
Luna	Earth
Oberon	Uranus
Charon	Pluto
...	...

MoonDiscoverer	
<u>name</u>	<u>discoverer</u>
Ganimesdes	Galileo Galilei
Calisto	Galileo Galilei
Europa	Galileo Galilei
Io	Galileo Galilei
Titan	Christiaan Huygens
Triton	William Lassell
Oberon	William Herschel
...	...

MoonDiscYear	
<u>name</u>	<u>year</u>
Ganimesdes	1610
Calisto	1610
Europa	1610
Io	1610
Titan	1655
Triton	1846
Oberon	1787
Charon	1978
...	...

Planets / Relational Database



Planet							
<u>name</u>	dist	radius	grav	days	years	temp	ring
Mercury	0.39	0.38	2.8	58.646	0.241	440	false
Venus	0.72	0.95	8.9	-243.019	0.615	730	false
Earth	1.00	1.00	9.8	0.997	1.000	288	false
Mars	1.52	0.53	3.7	1.026	1.880	186	false
Jupiter	5.20	10.97	22.9	0.414	11.862	152	true
Saturn	9.54	9.14	9.1	0.444	29.447	134	true
Uranus	19.19	3.98	7.8	-0.719	84.017	76	true
Neptune	30.07	3.86	11.0	0.671	164.791	53	true

DwarfPlanet							
<u>name</u>	dist	radius	grav	days	years	temp	ring
Pluto	49.31	0.19	0.063	6.39	248.000	44	false

Moon	
<u>name</u>	P.name
Ganimesdes	Jupiter
Calisto	Jupiter
Europa	Jupiter
Io	Jupiter
Titan	Saturn
Triton	Neptune
Luna	Earth
Oberon	Uranus
Charon	Pluto
...	...

MoonDiscoverer	
<u>name</u>	discoverer
Ganimesdes	Galileo Galilei
Calisto	Galileo Galilei
Europa	Galileo Galilei
Io	Galileo Galilei
Titan	Christiaan Huygens
Triton	William Lassell
Oberon	William Herschel
...	...

MoonDiscYear	
<u>name</u>	year
Ganimesdes	1610
Calisto	1610
Europa	1610
Io	1610
Titan	1655
Triton	1846
Oberon	1787
Charon	1978
...	...

Planets / Relational Database



Planet							
<u>name</u>	dist	radius	grav	days	years	temp	ring
Mercury	0.39	0.38	2.8	58.646	0.241	440	false
Venus	0.72	0.95	8.9	-243.019	0.615	730	false
Earth	1.00	1.00	9.8	0.997	1.000	288	false
Mars	1.52	0.53	3.7	1.026	1.880	186	false
Jupiter	5.20	10.97	22.9	0.414	11.862	152	true
Saturn	9.54	9.14	9.1	0.444	29.447	134	true
Uranus	19.19	3.98	7.8	-0.719	84.017	76	true
Neptune	30.07	3.86	11.0	0.671	164.791	53	true

DwarfPlanet							
<u>name</u>	dist	radius	grav	days	years	temp	ring
Pluto	49.31	0.19	0.063	6.39	248.000	44	false

Moon	
<u>name</u>	P.name
Ganimesdes	Jupiter
Calisto	Jupiter
Europa	Jupiter
Io	Jupiter
Titan	Saturn
Triton	Neptune
Luna	Earth
Oberon	Uranus
Charon	Pluto
...	...

MoonDiscoverer	
<u>name</u>	discoverer
Ganimesdes	Galileo Galilei
Calisto	Galileo Galilei
Europa	Galileo Galilei
Io	Galileo Galilei
Titan	Christiaan Huygens
Triton	William Lassell
Oberon	William Herschel
...	...

MoonDiscYear	
<u>name</u>	year
Ganimesdes	1610
Calisto	1610
Europa	1610
Io	1610
Titan	1655
Triton	1846
Oberon	1787
Charon	1978
...	...

Planets / Relational Database



Planet							
<u>name</u>	dist	radius	grav	days	years	temp	ring
Mercury	0.39	0.38	2.8	58.646	0.241	440	false
Venus	0.72	0.95	8.9	-243.019	0.615	730	false
Earth	1.00	1.00	9.8	0.997	1.000	288	false
Mars	1.52	0.53	3.7	1.026	1.880	186	false
Jupiter	5.20	10.97	22.9	0.414	11.862	152	true
Saturn	9.54	9.14	9.1	0.444	29.447	134	true
Uranus	19.19	3.98	7.8	-0.719	84.017	76	true
Neptune	30.07	3.86	11.0	0.671	164.791	53	true

DwarfPlanet							
<u>name</u>	dist	radius	grav	days	years	temp	ring
Pluto	49.31	0.19	0.063	6.39	248.000	44	false

Moon	
<u>name</u>	parent
Ganimesdes	Jupiter
Calisto	Jupiter
Europa	Jupiter
Io	Jupiter
Titan	Saturn
Triton	Neptune
Luna	Earth
Oberon	Uranus
Charon	Pluto
...	...

MoonDiscoverer	
<u>name</u>	discoverer
Ganimesdes	Galileo Galilei
Calisto	Galileo Galilei
Europa	Galileo Galilei
Io	Galileo Galilei
Titan	Christiaan Huygens
Triton	William Lassell
Oberon	William Herschel
...	...

MoonDiscYear	
<u>name</u>	year
Ganimesdes	1610
Calisto	1610
Europa	1610
Io	1610
Titan	1655
Triton	1846
Oberon	1787
Charon	1978
...	...

Planets / Relational Database



Planet

<u>name</u>	dist	radius	grav	days	years	temp	ring
Mercury	0.39	0.38	2.8	58.646	0.241	440	false
Venus	0.72	0.95	8.9	-243.019	0.615	730	false
Earth	1.00	1.00	9.8	0.997	1.000	288	false
Mars	1.52	0.53	3.7	1.026	1.880	186	false
Jupiter	5.20	10.97	22.9	0.414	11.862	152	true
Saturn	9.54	9.14	9.1	0.444	29.447	134	true
Uranus	19.19	3.98	7.8	-0.719	84.017	76	true
Neptune	30.07	3.86	11.0	0.671	164.791	53	true

DwarfPlanet

<u>name</u>	dist	radius	grav	days	years	temp	ring
Pluto	49.31	0.19	0.063	6.39	248.000	44	false

Moon

<u>name</u>	parent
Ganimesdes	Jupiter
Calisto	Jupiter
Europa	Jupiter
Io	Jupiter
Titan	Saturn
Triton	Neptune
Luna	Earth
Oberon	Uranus
Charon	Pluto
...	...

MoonDiscoverer

<u>name</u>	discoverer
Ganimesdes	Galileo Galilei
Calisto	Galileo Galilei
Europa	Galileo Galilei
Io	Galileo Galilei
Titan	Christiaan Huygens
Triton	William Lassell
Oberon	William Herschel
...	...

MoonDiscYear

<u>name</u>	year
Ganimesdes	1610
Calisto	1610
Europa	1610
Io	1610
Titan	1655
Triton	1846
Oberon	1787
Charon	1978
...	...

Planets / Relational Database



Planet							
<u>name</u>	dist	radius	grav	days	years	temp	ring
Mercury	0.39	0.38	2.8	58.646	0.241	440	false
Venus	0.72	0.95	8.9	-243.019	0.615	730	false
Earth	1.00	1.00	9.8	0.997	1.000	288	false
Mars	1.52	0.53	3.7	1.026	1.880	186	false
Jupiter	5.20	10.97	22.9	0.414	11.862	152	true
Saturn	9.54	9.14	9.1	0.444	29.447	134	true
Uranus	19.19	3.98	7.8	-0.719	84.017	76	true
Neptune	30.07	3.86	11.0	0.671	164.791	53	true

DwarfPlanet							
<u>name</u>	dist	radius	grav	days	years	temp	ring
Pluto	49.31	0.19	0.005	0.39	248.000	44	false

Moon	
<u>name</u>	parent
Ganimesdes	Jupiter
Calisto	Jupiter
Europa	Jupiter
Io	Jupiter
Titan	Saturn
Triton	Neptune
Luna	Earth
Oberon	Uranus
Charon	Pluto
...	...

MoonDiscoverer	
<u>name</u>	discoverer
Ganimesdes	Galileo Galilei
Calisto	Galileo Galilei
Europa	Galileo Galilei
Io	Galileo Galilei
Titan	Christiaan Huygens
Triton	William Lassell
Oberon	William Herschel
...	...

MoonDiscYear	
<u>name</u>	year
Ganimesdes	1610
Calisto	1610
Europa	1610
Io	1610
Titan	1655
Triton	1846
Oberon	1787
Charon	1978
...	...

Planets / **Graph** Database



Planets / Graph Database



Planets / Graph Database

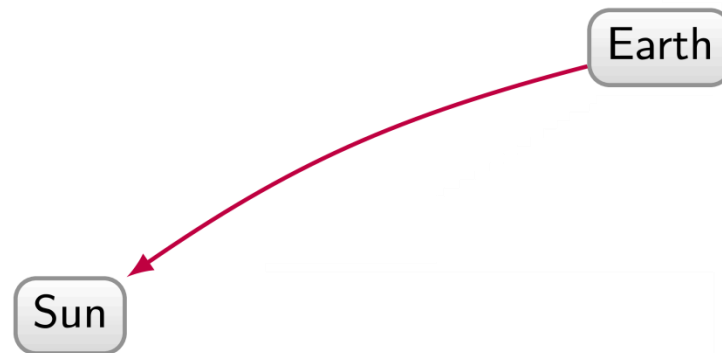
Earth

The image shows a graph database visualization. A single node, represented by a rounded rectangle with a light gray background and a dark gray border, is labeled 'Earth'. The node is positioned on the right side of the frame. The rest of the graph is empty, with no other nodes or edges visible.

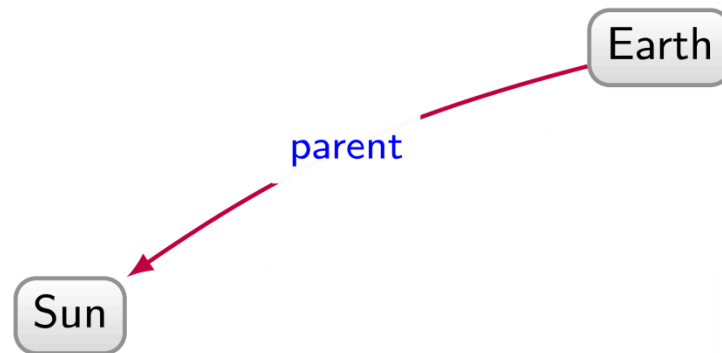
Planets / Graph Database



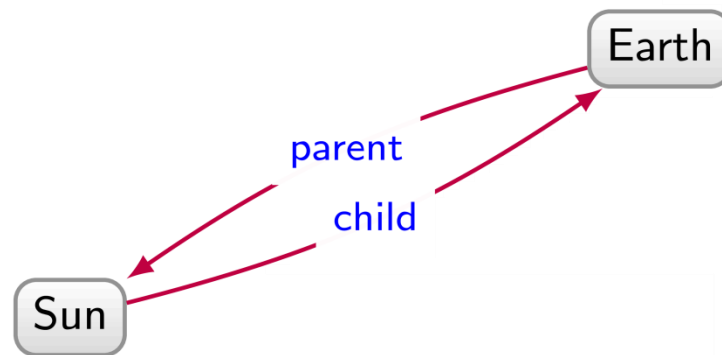
Planets / Graph Database



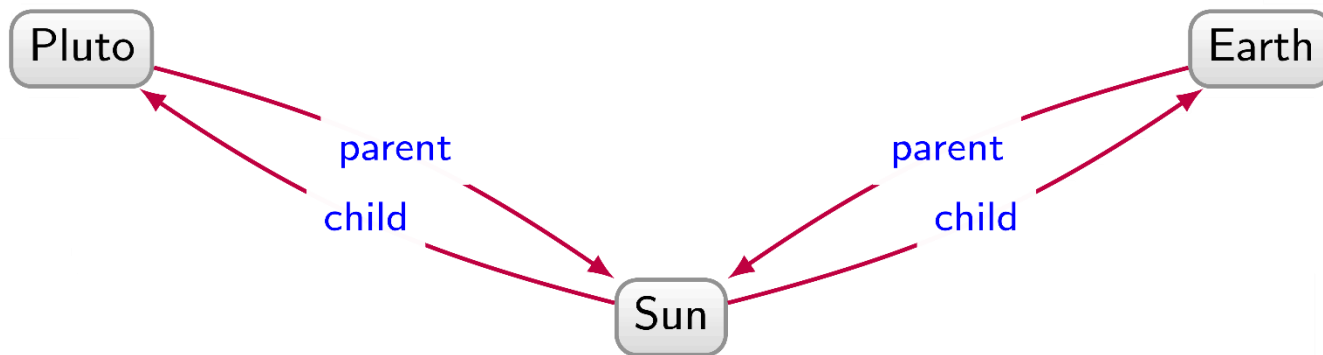
Planets / Graph Database



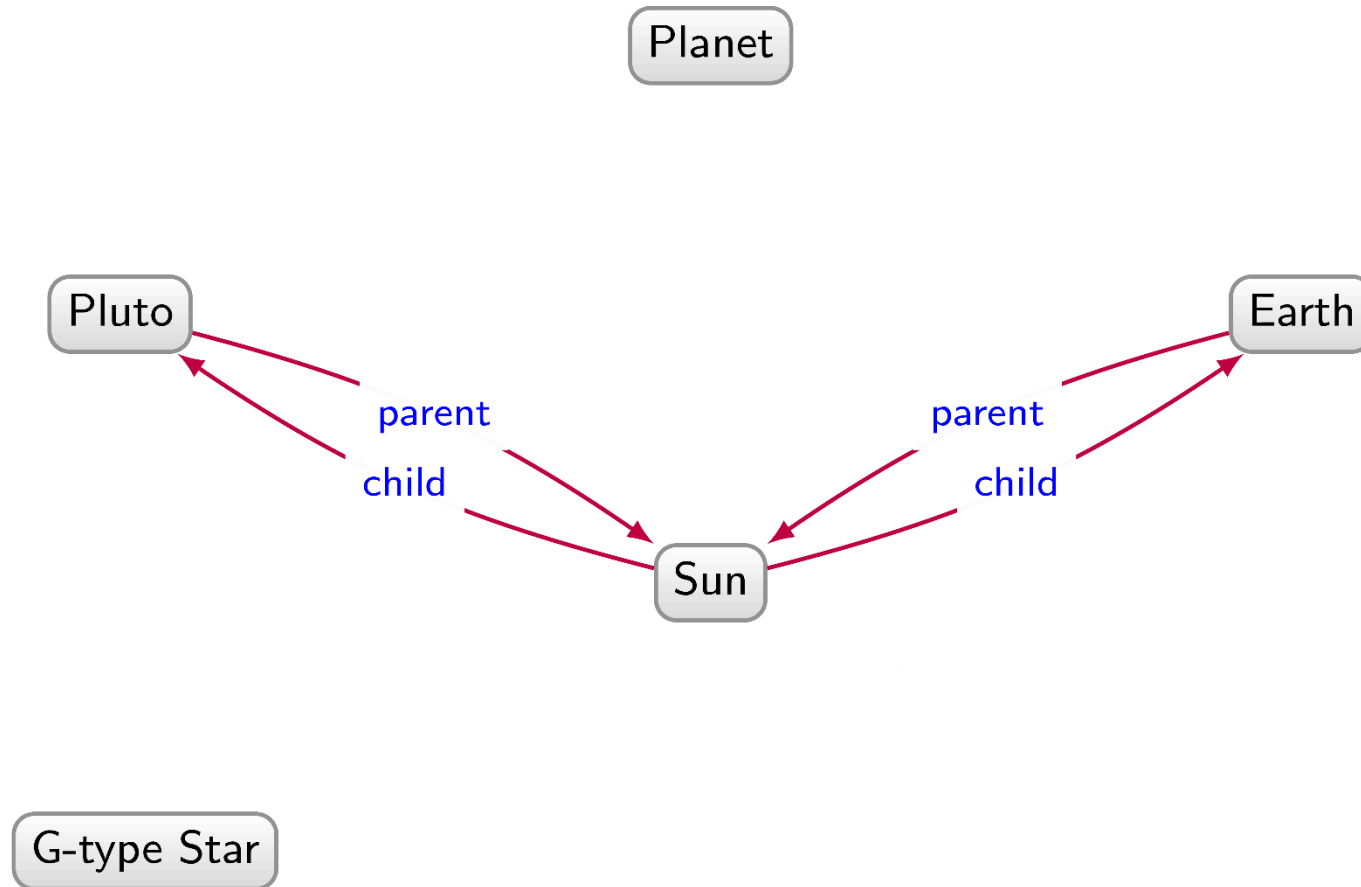
Planets / Graph Database



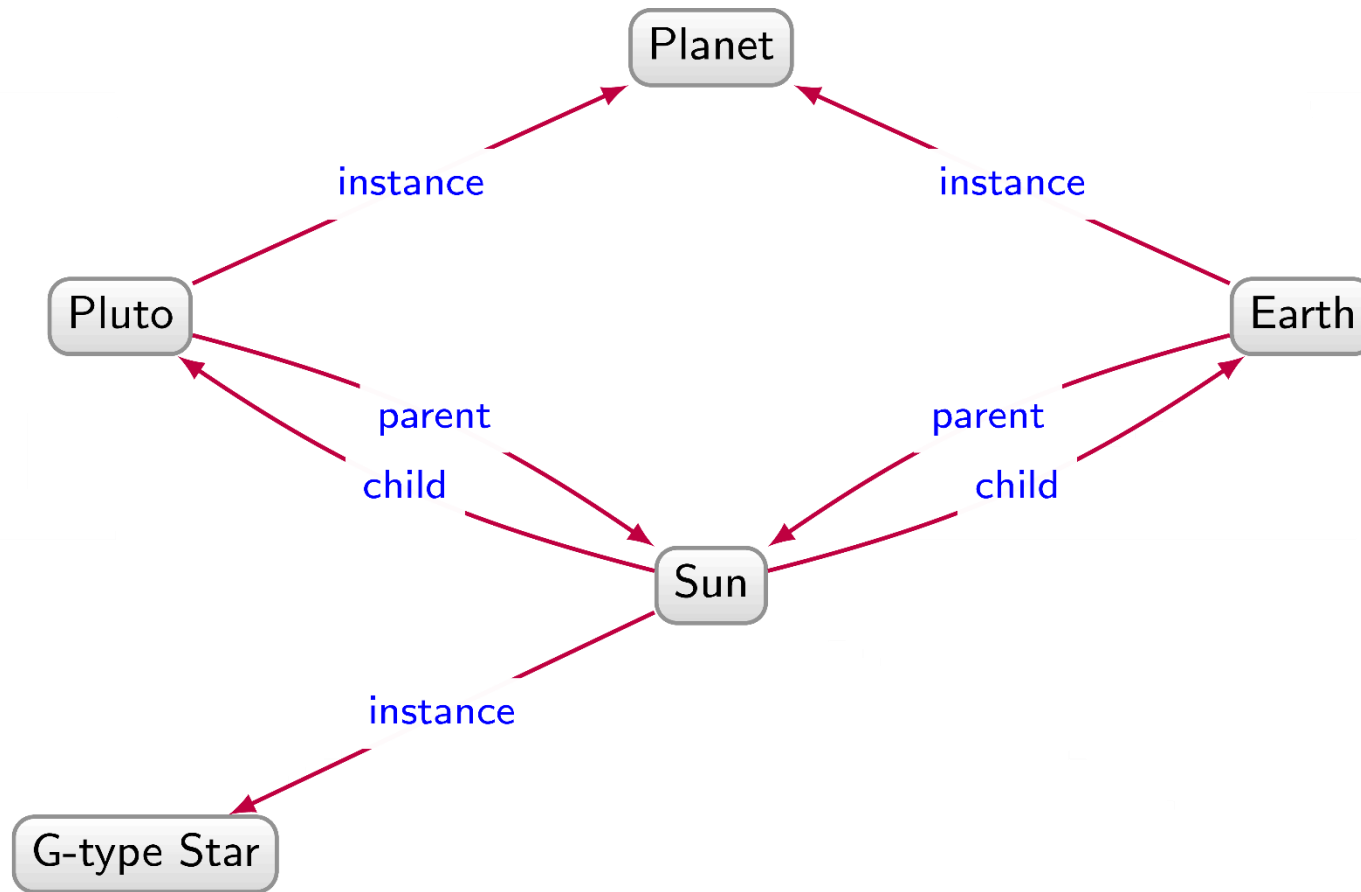
Planets / Graph Database



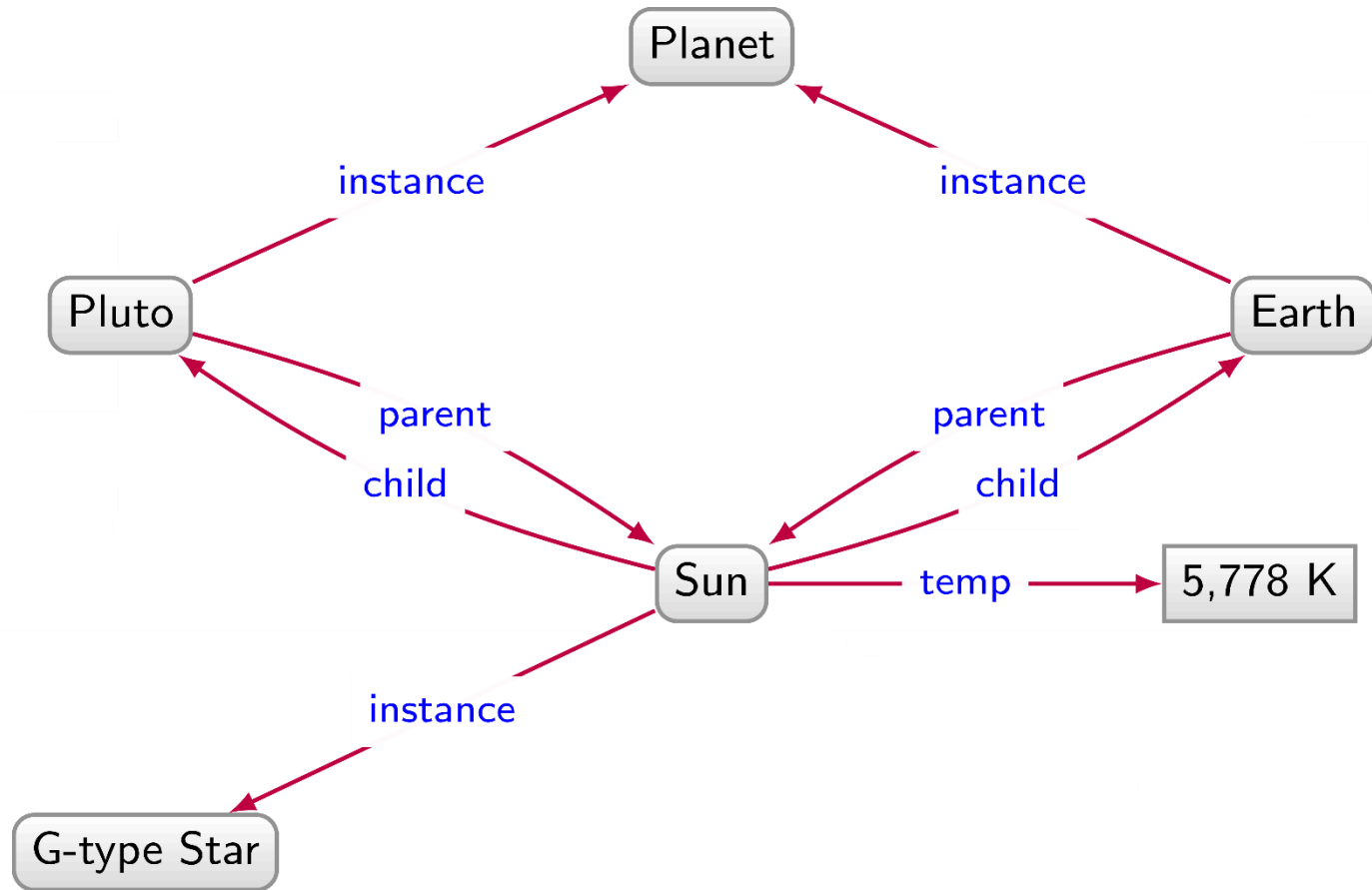
Planets / Graph Database



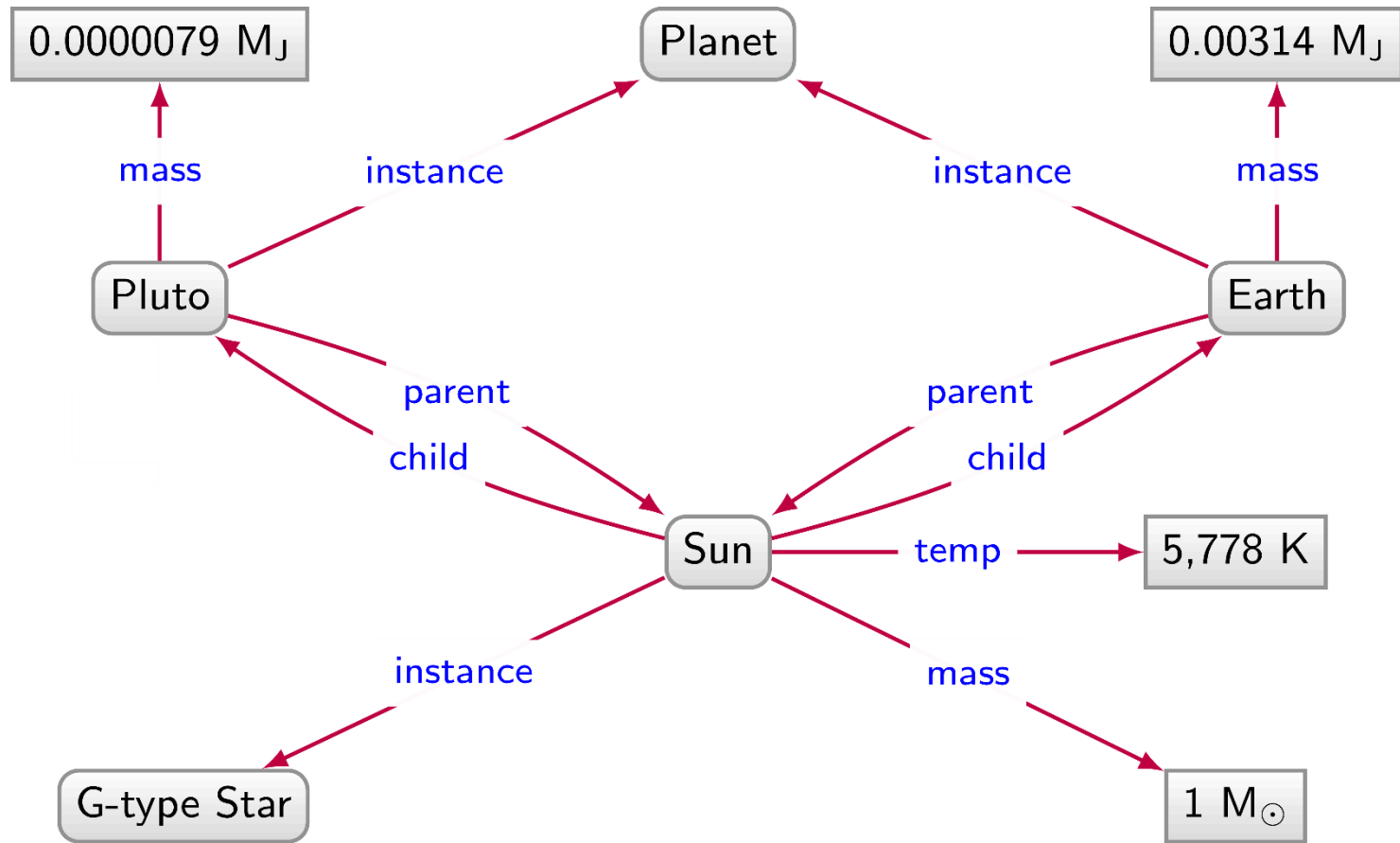
Planets / Graph Database



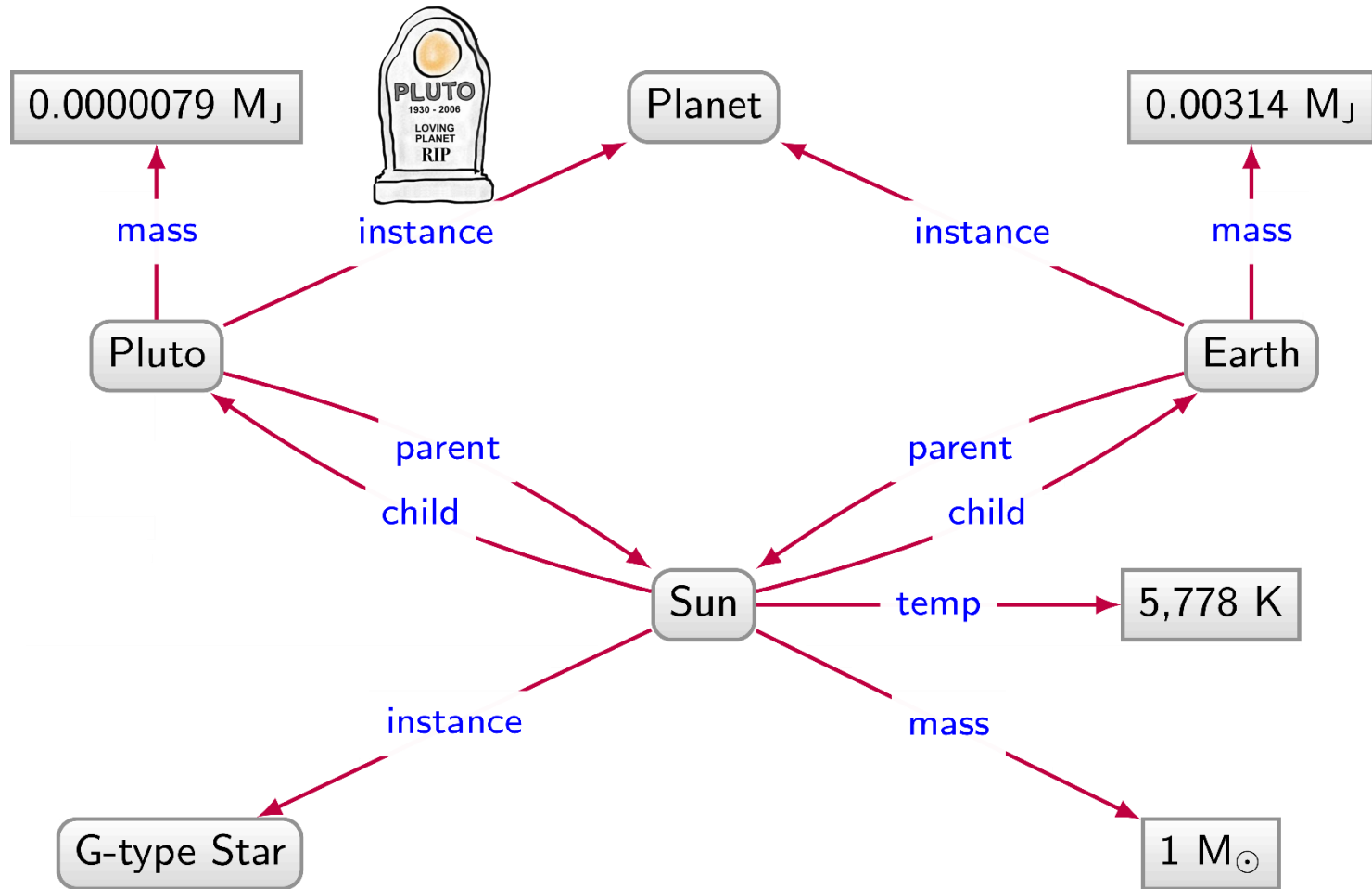
Planets / Graph Database



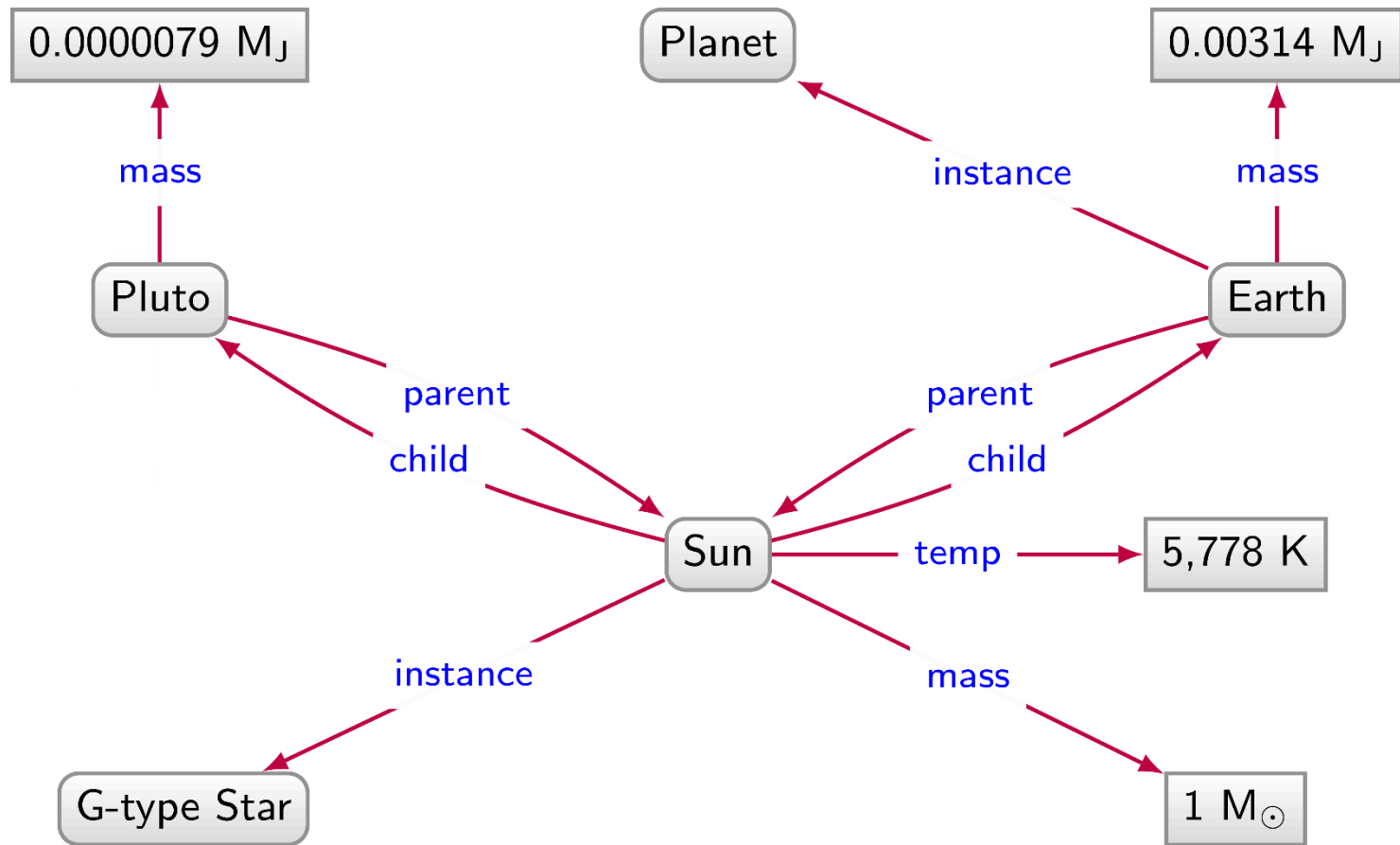
Planets / Graph Database



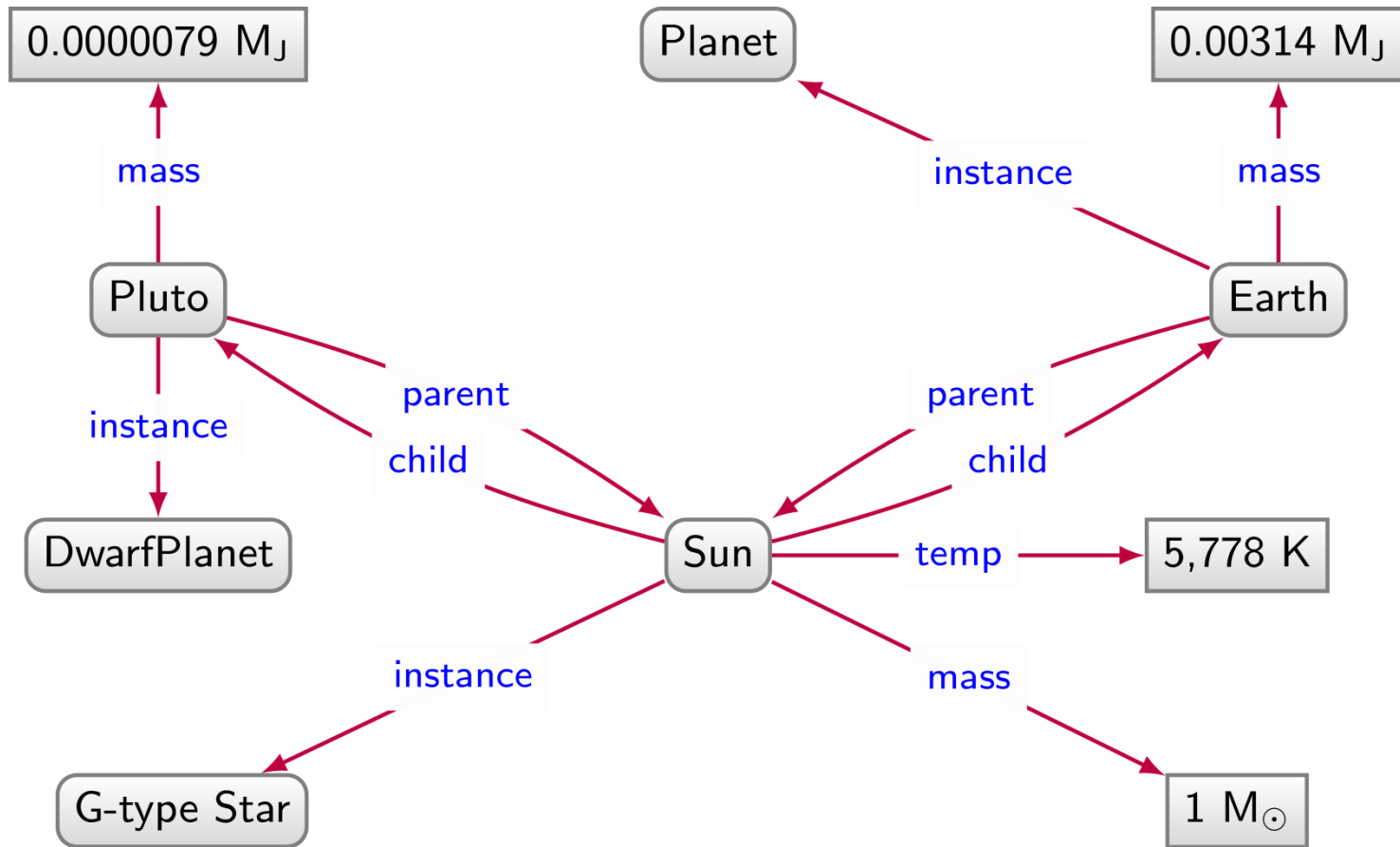
Planets / Graph Database



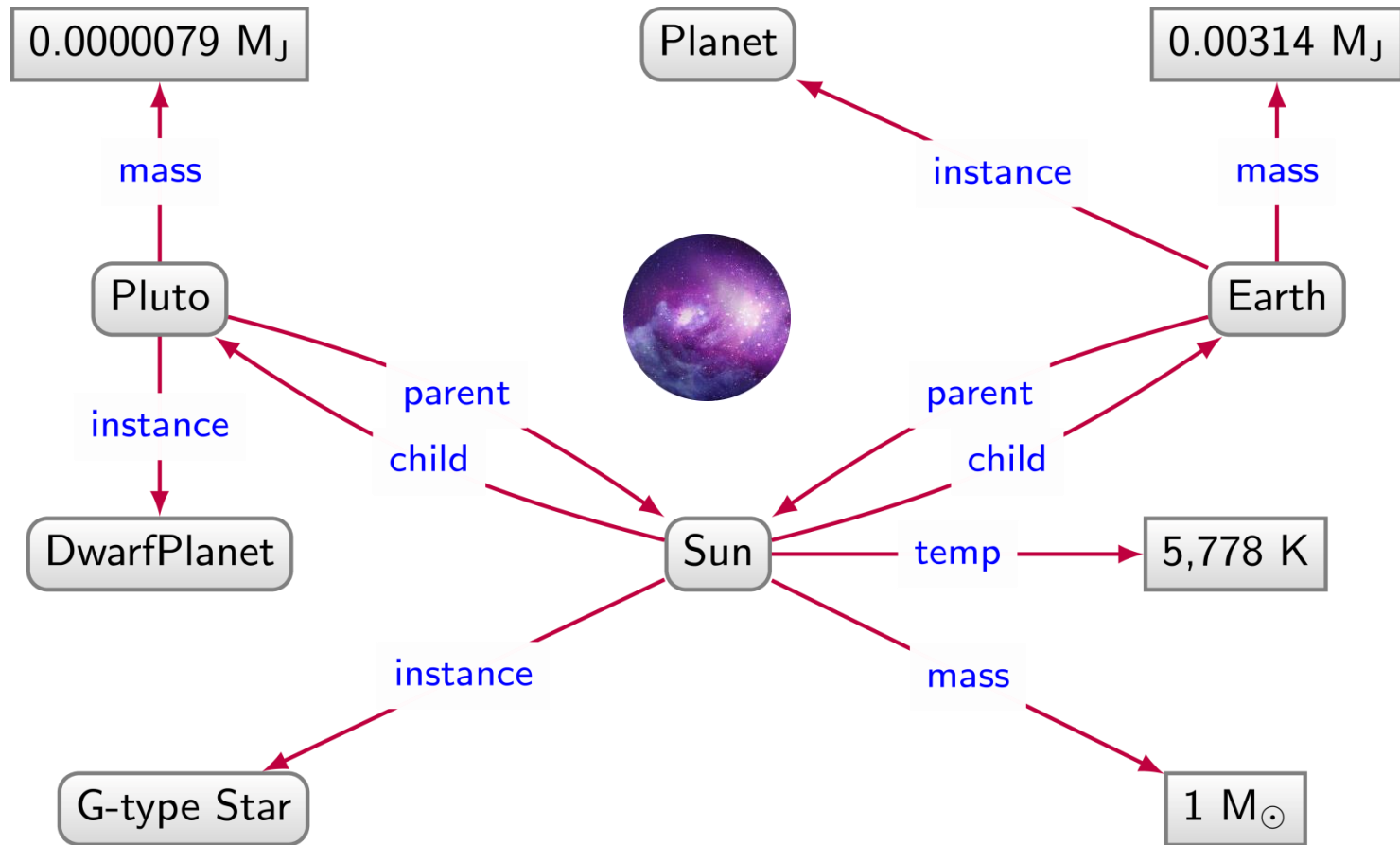
Planets / Graph Database



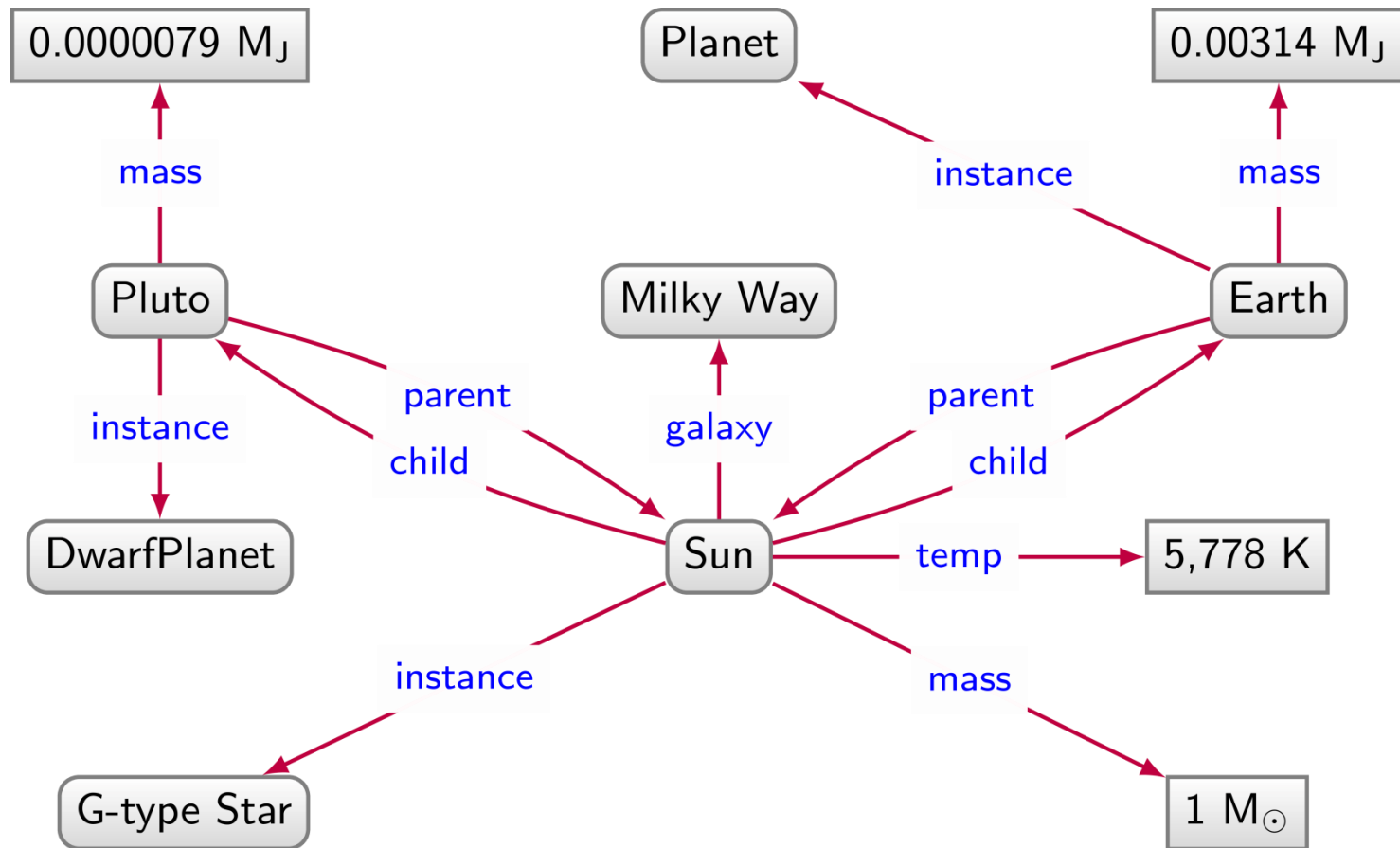
Planets / Graph Database



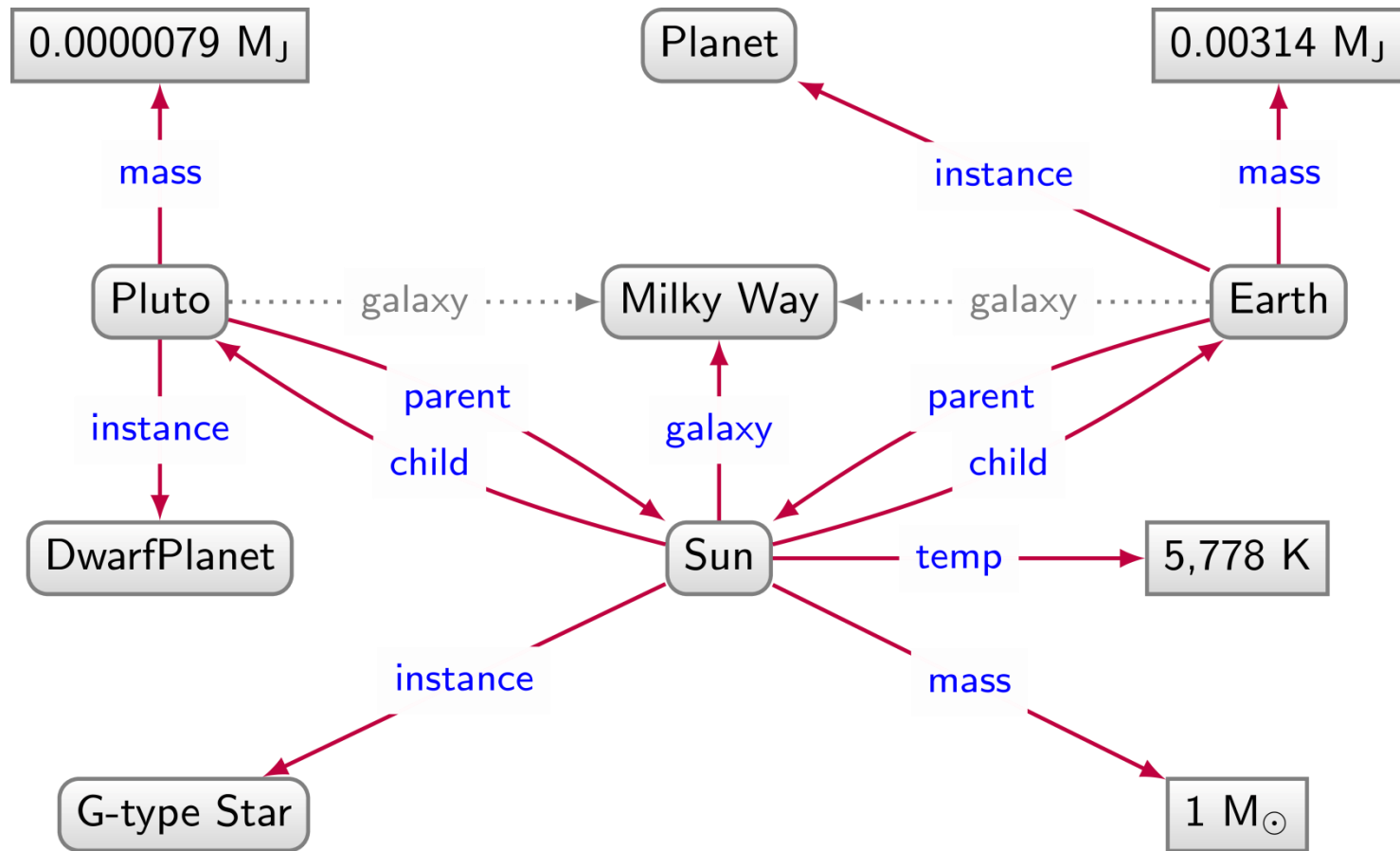
Planets / Graph Database



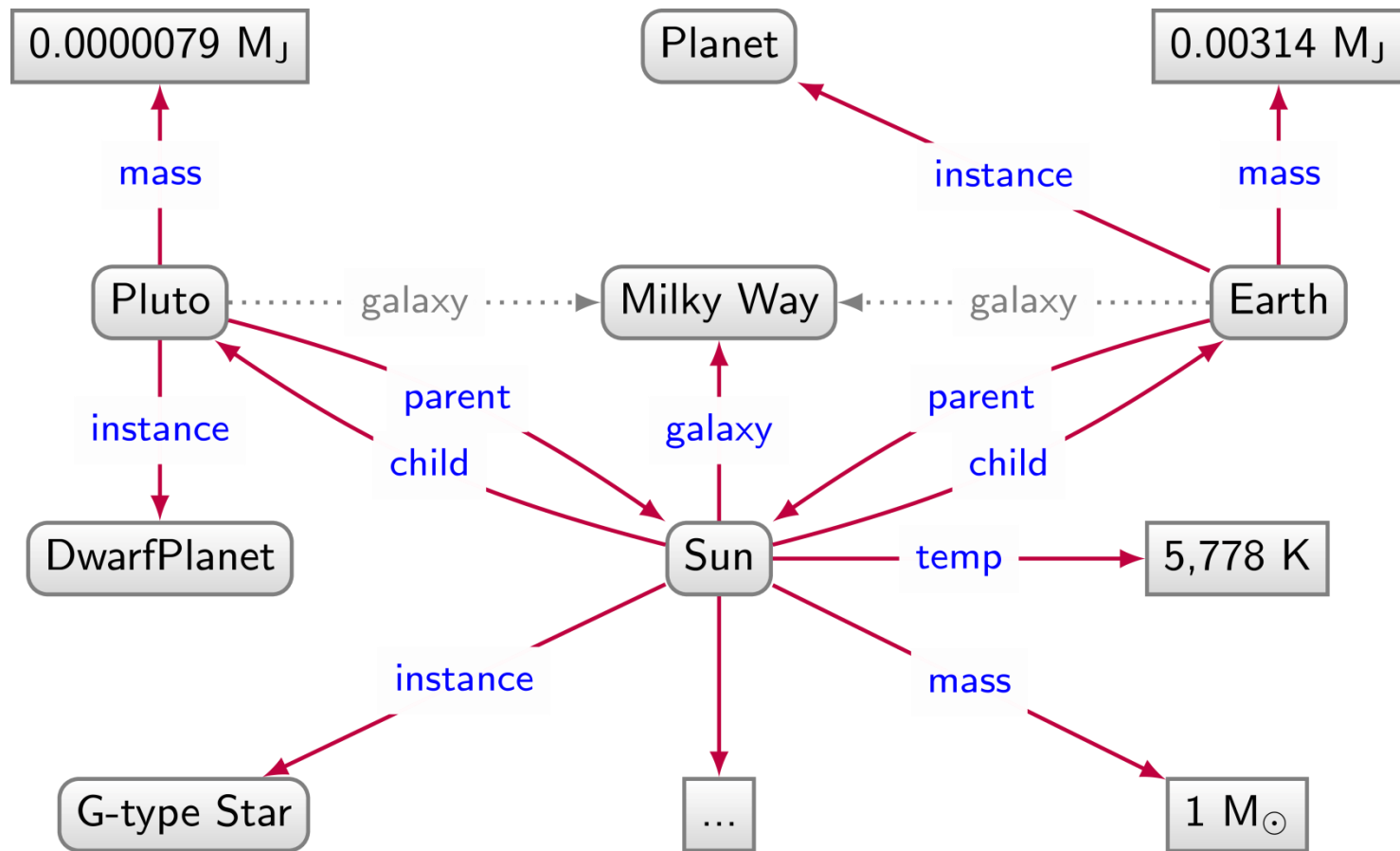
Planets / Graph Database



Planets / Graph Database



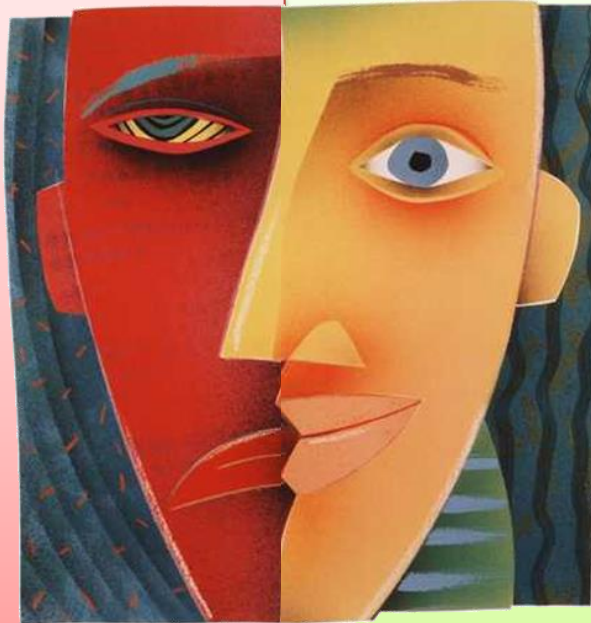
Planets / Graph Database



Relational databases: Pros and Cons

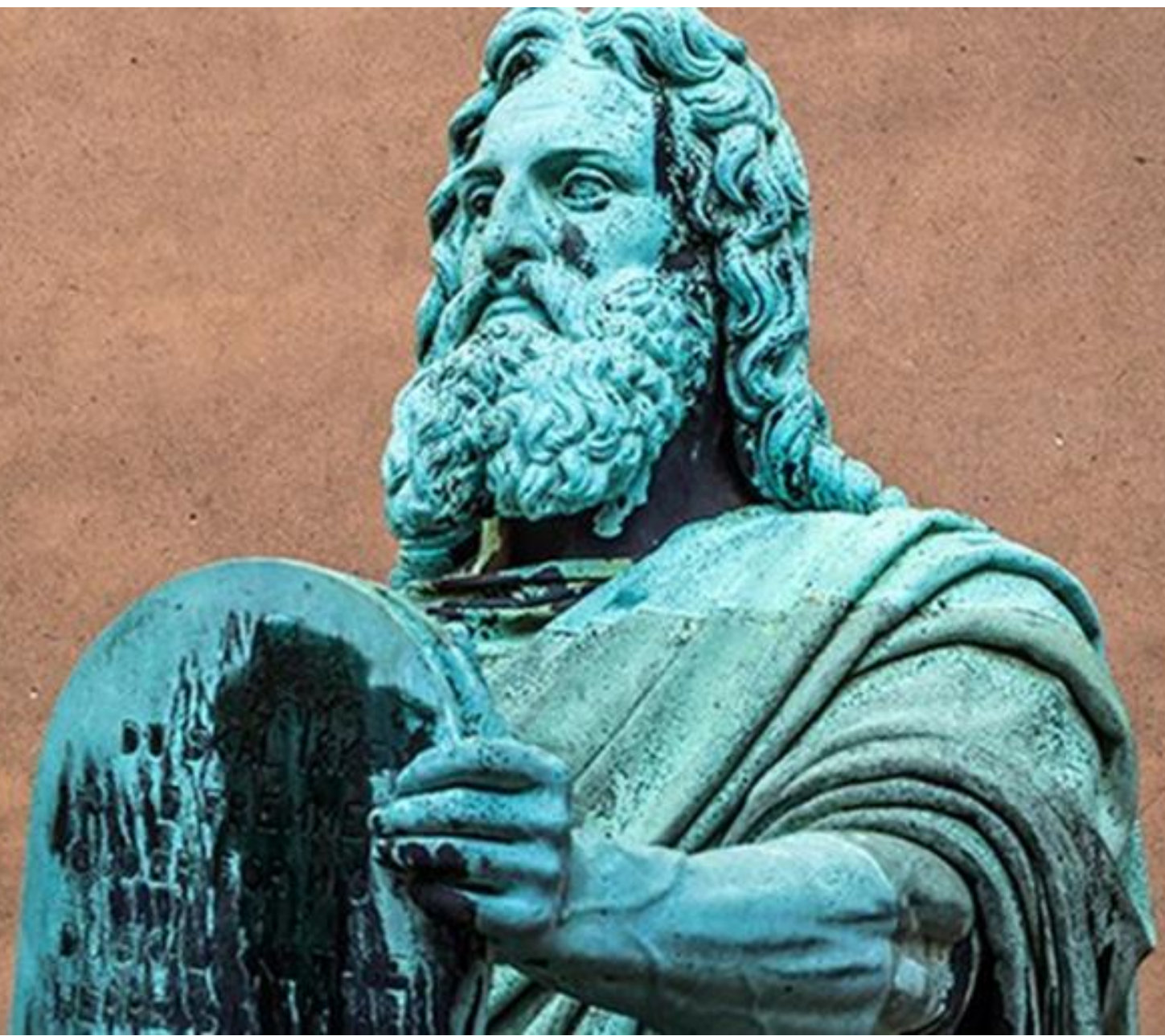
Planet								
<u>name</u>	dist	radius	grav	days	years	temp	ring	
Mercury	0.39	0.38	2.8	58.646	0.241	440	false	
Venus	0.72	0.95	8.9	-243.019	0.615	730	false	

We have to impose a structure (schema) from the start

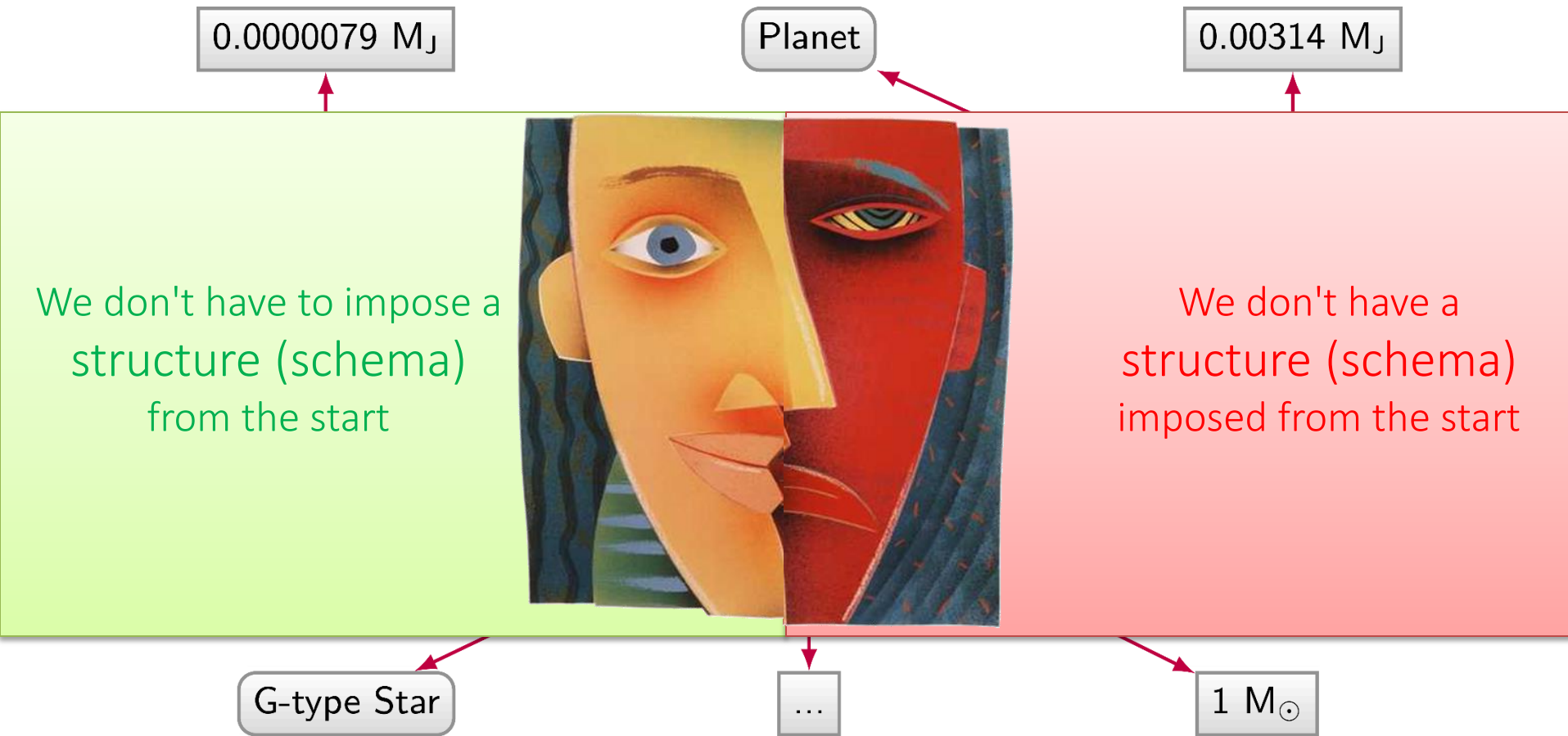


We have a structure (schema) imposed from the start

Europa	Jupiter	Europa	Galileo Galilei	Europa	1610
Io	Jupiter	Io	Galileo Galilei	Io	1610
Titan	Saturn	Titan	Christiaan Huygens	Titan	1655
Triton	Neptune	Triton	William Lassell	Triton	1846
Luna	Earth	Oberon	William Herschel	Oberon	1787
Oberon	Uranus	Charon	1978
Charon	Pluto		
...	...				



Graph Databases: Pros and Cons

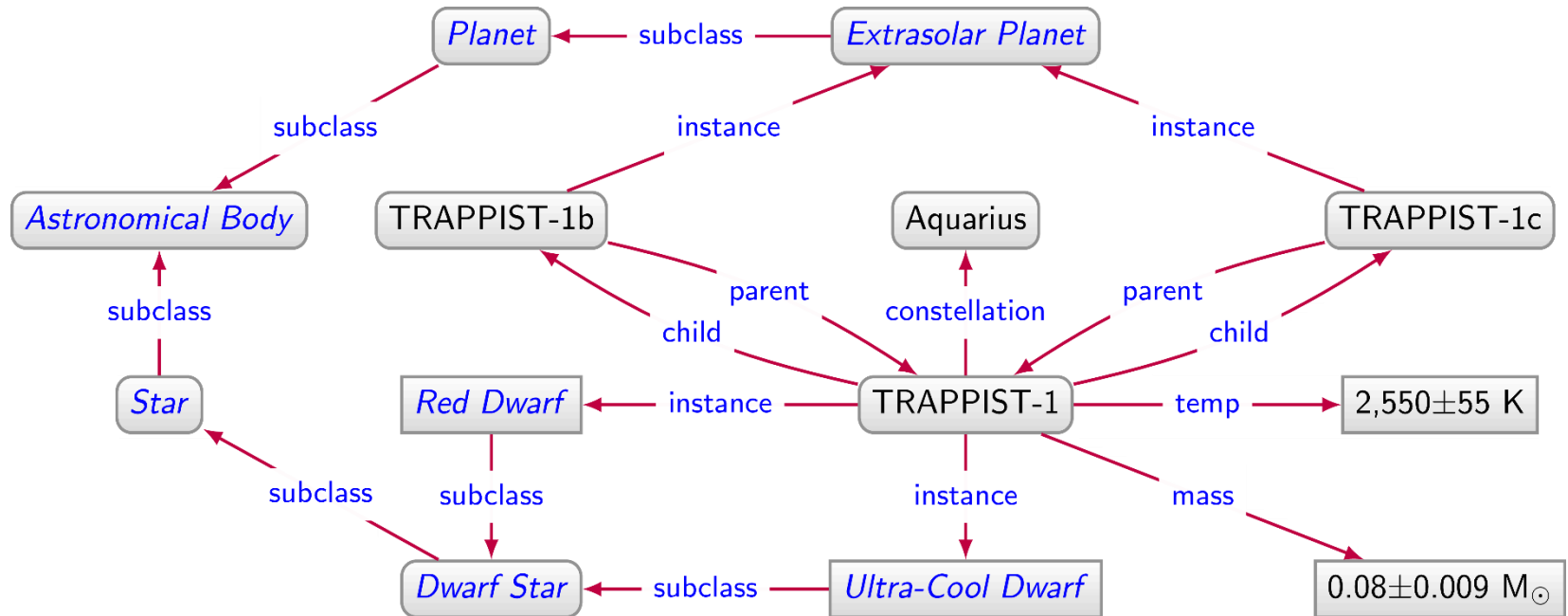




WHY DO WE NEED GRAPH DATABASES?

PATH QUERIES

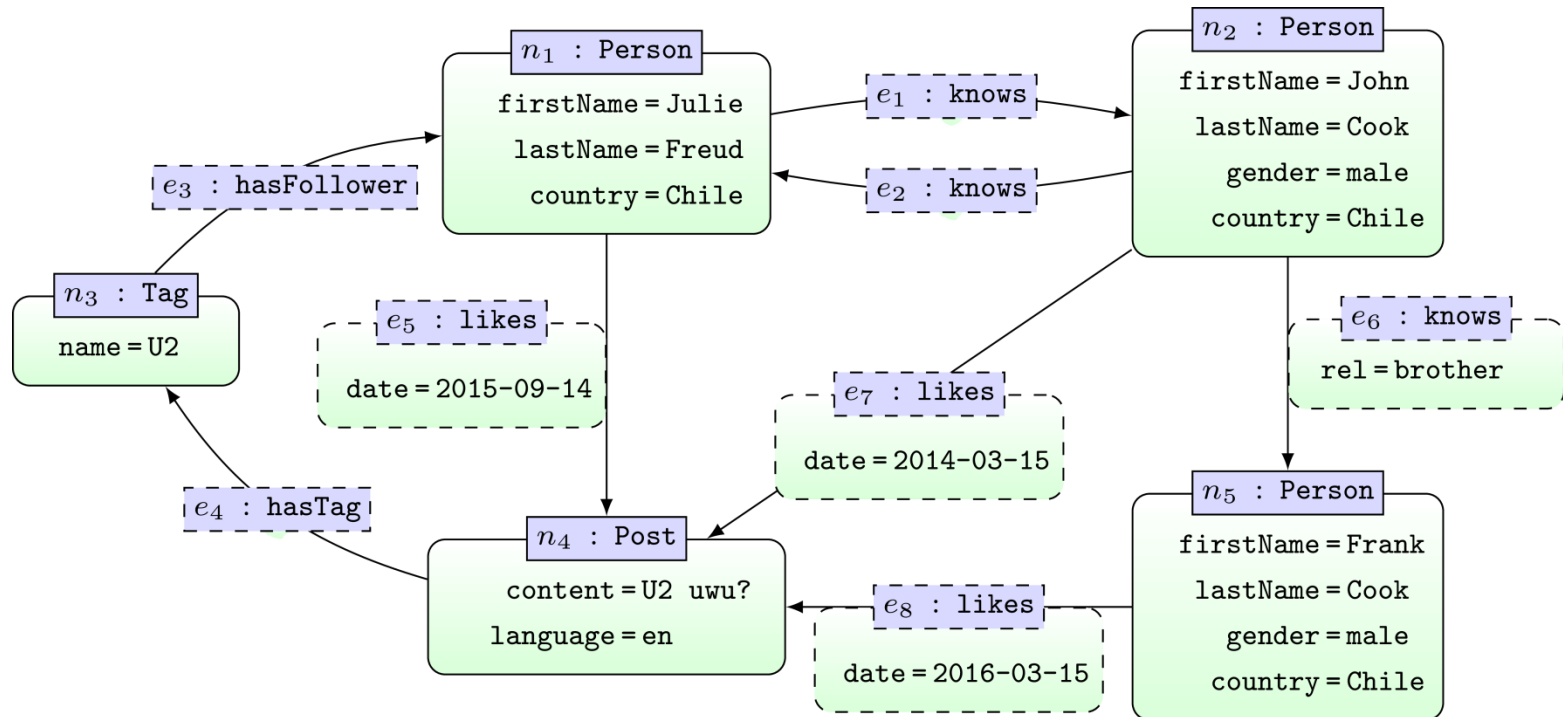
Directed Edge-labelled Graph



```
SELECT ?const (COUNT(DISTINCT ?body) AS ?num)
WHERE {
  ?body :instance/:subclass* :AstronomicalBody .
  ?body :parent?/:constellation ?const .
}
GROUP BY ?const
ORDER BY DESC(?num)
```

?const	?num
:Aquarius	3

Property Graph



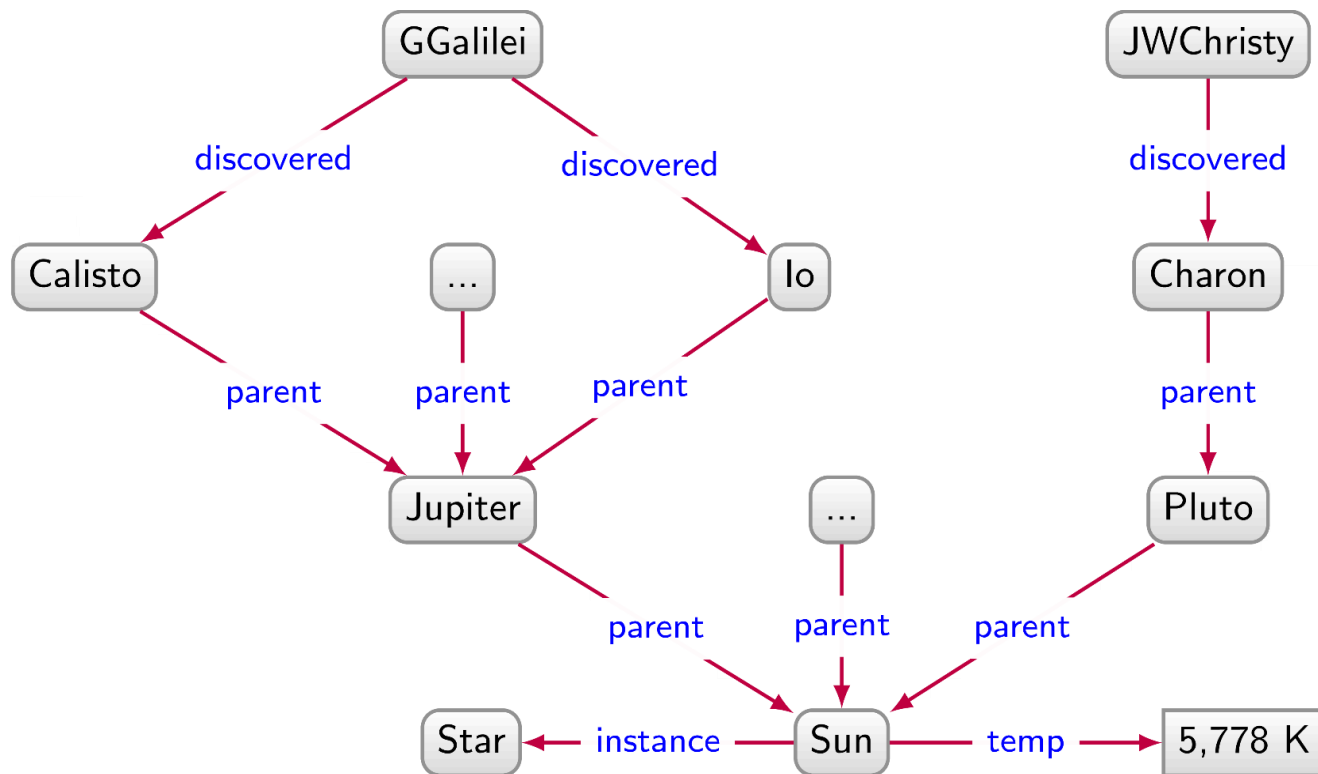
```
MATCH (x1:Person {firstName:"Julie"})-[:knows*]->(x2:Person)
MATCH (x2)-[:likes]->()-[:hasTag]->()-[:hasFollower]->(x1)
RETURN x2.firstName
```

???

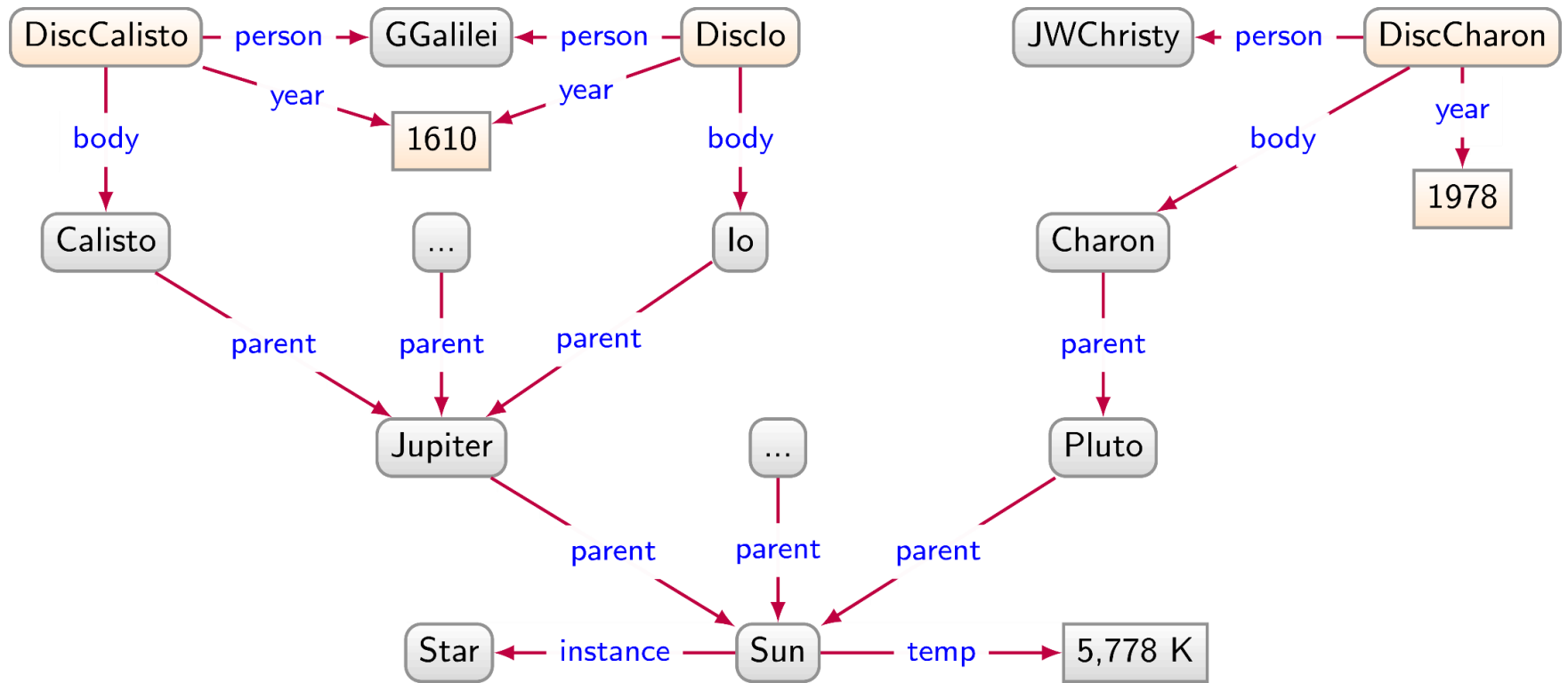
WHY DO WE NEED PROPERTY GRAPHS?

Directed Labelled Graph

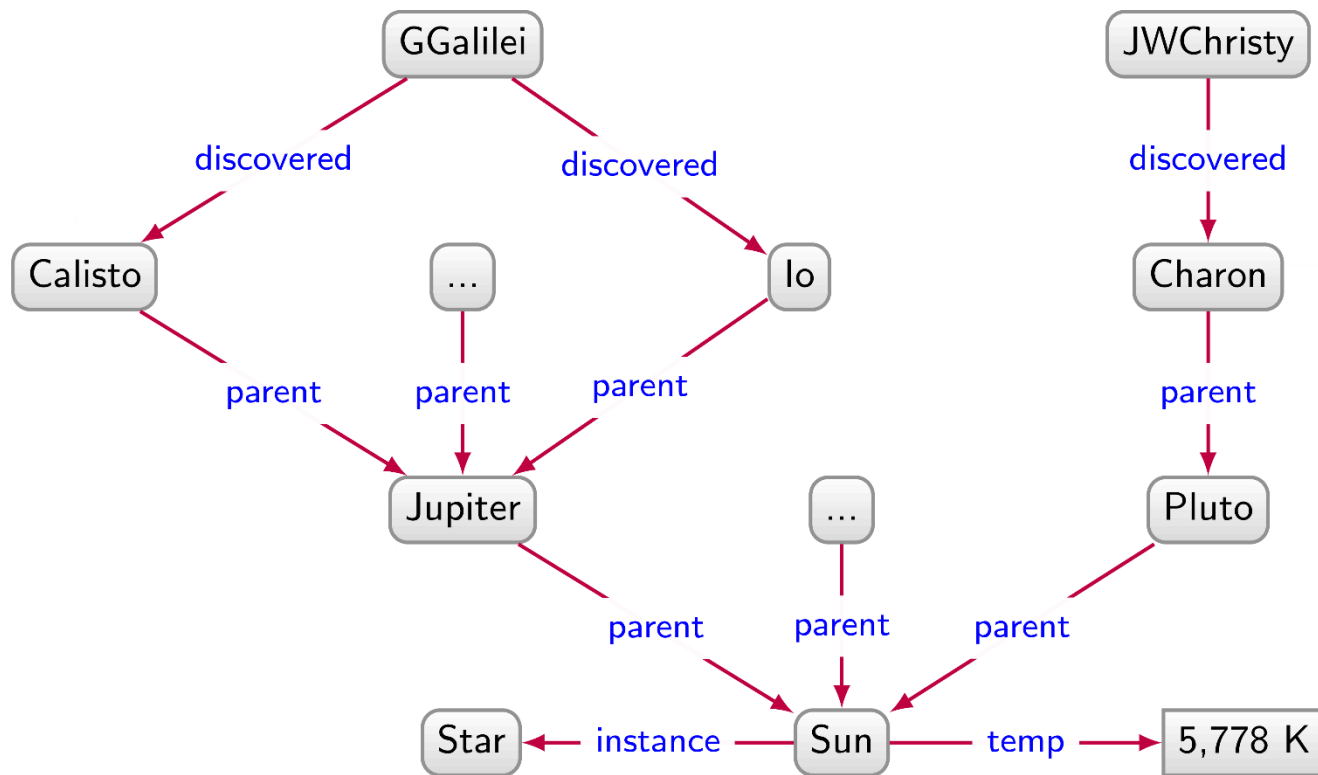
How can we say that Galileo Galilei discovered Calisto and Io in 1610 while James W. Christy discovered Charon in 1978?



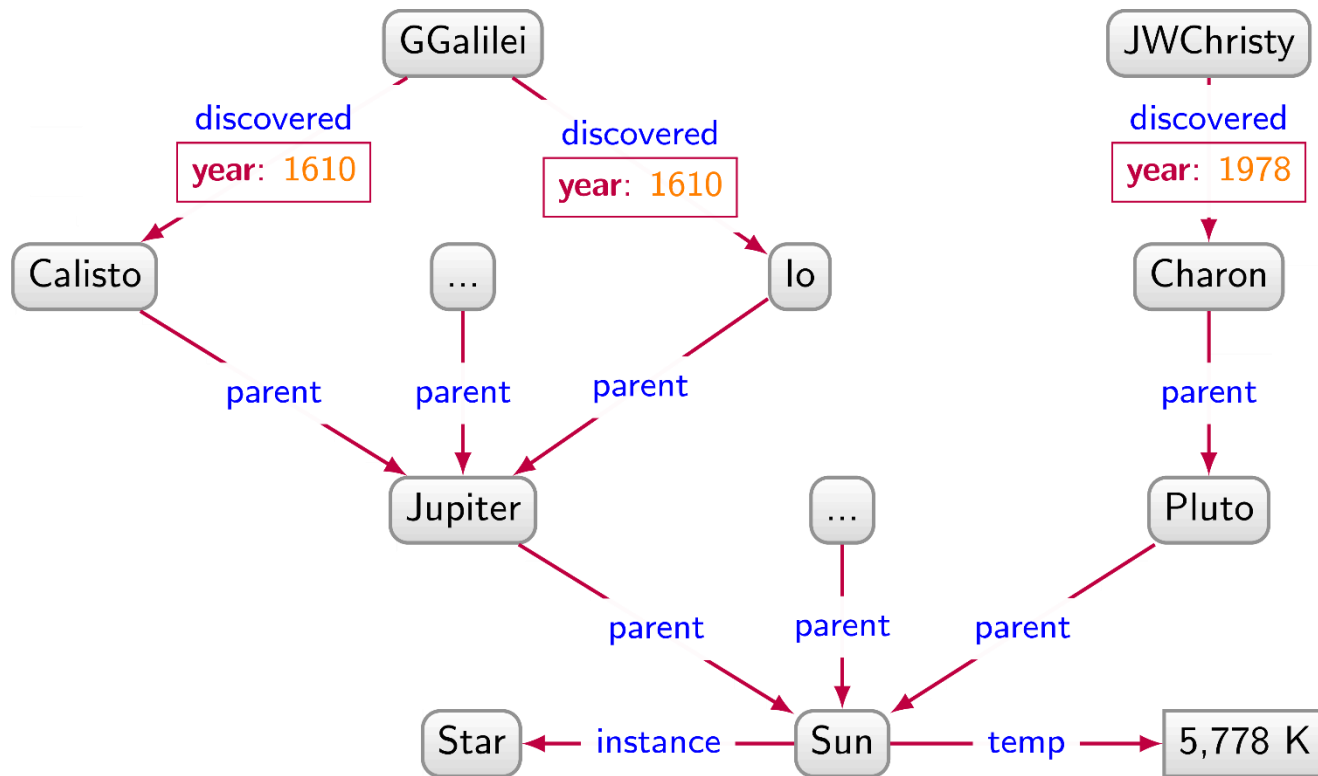
Directed Labelled Graph



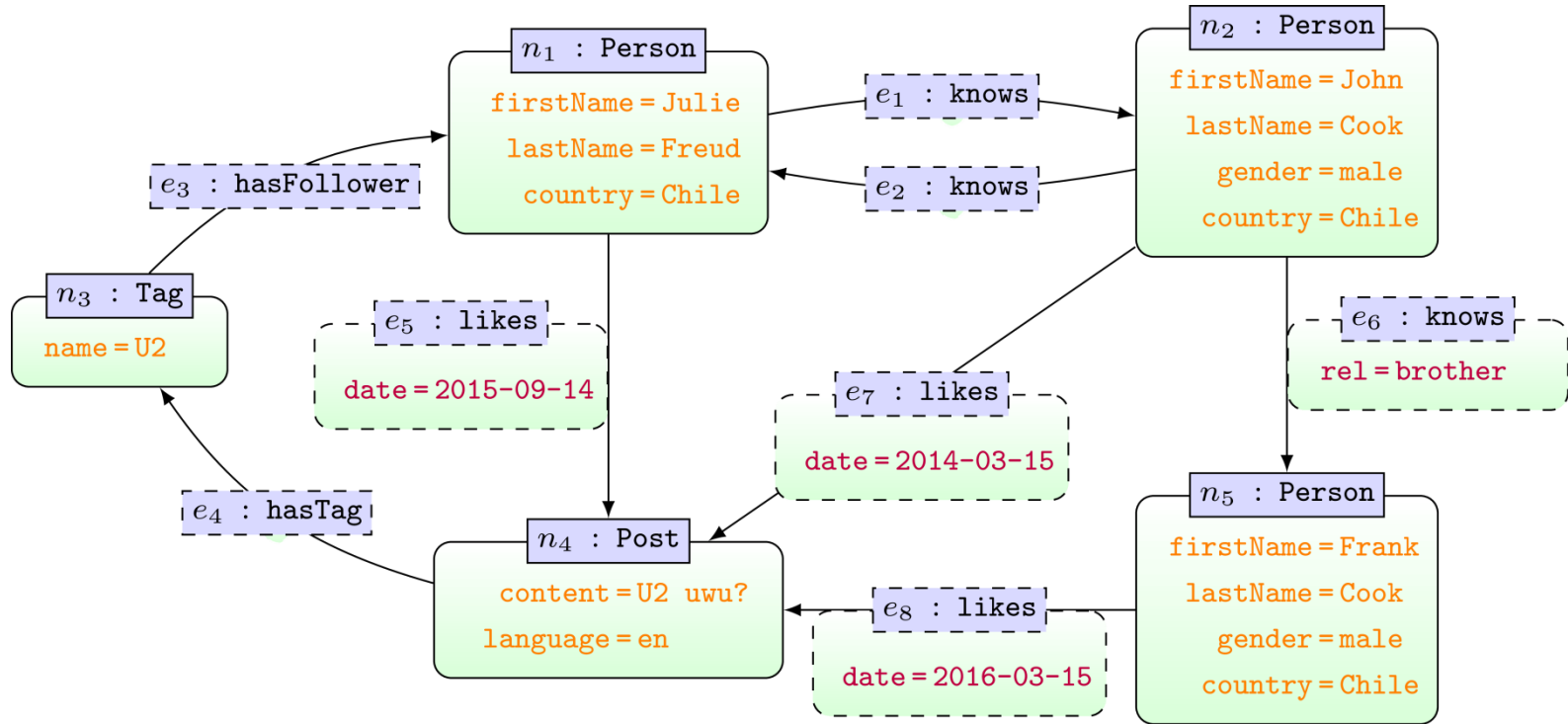
Wouldn't it have been nice to simply ...



Wouldn't it have been nice to simply ...

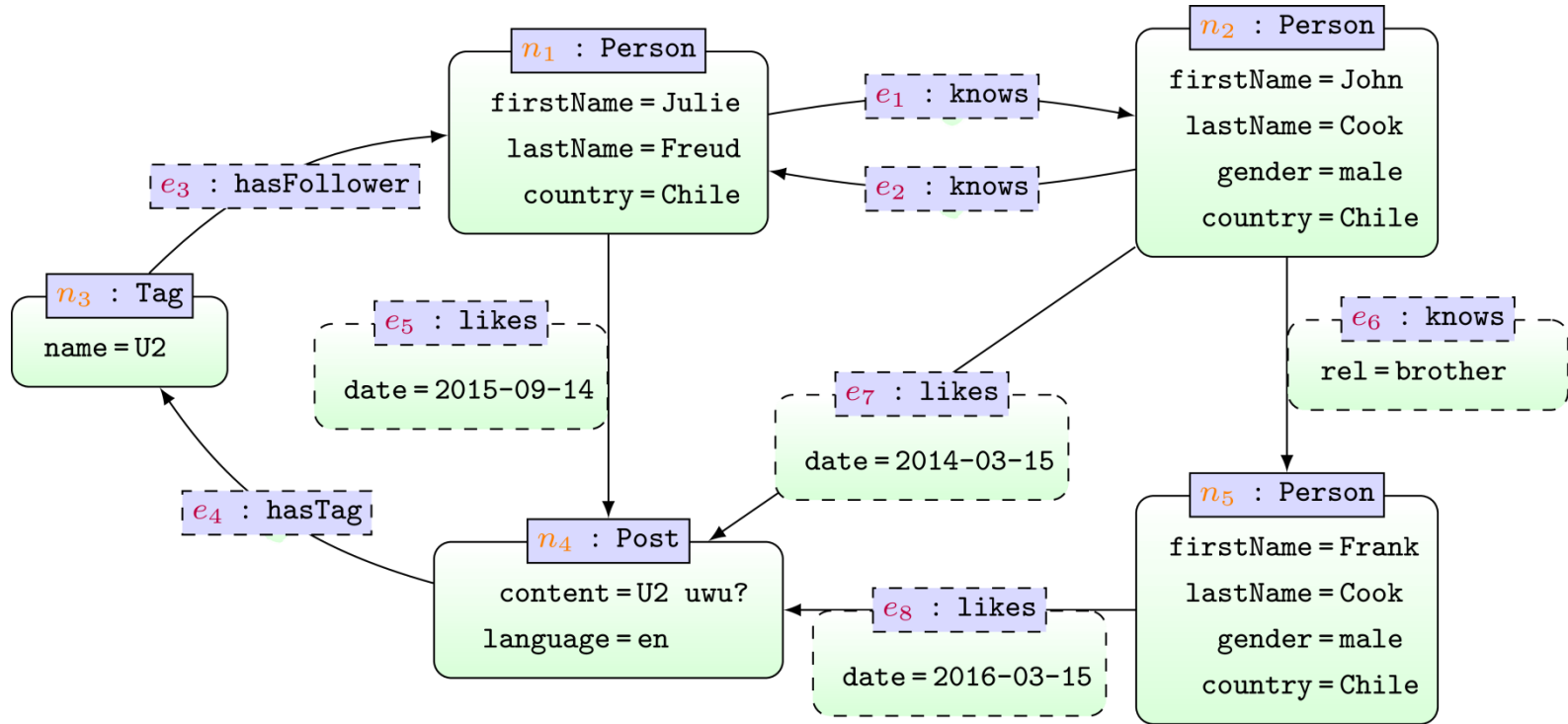


Property Graphs ...



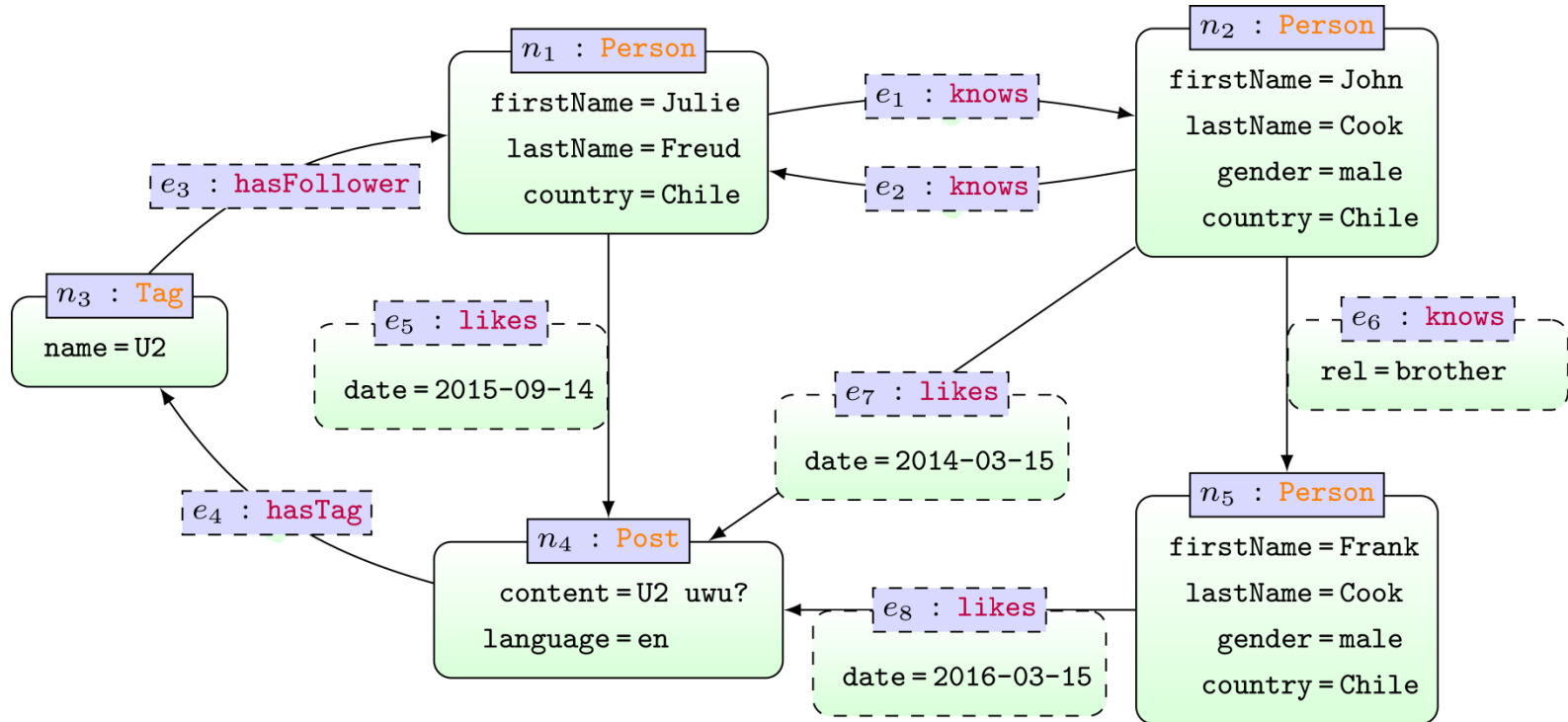
... attributes on **nodes** and **edges**

Property Graphs



... attributes on nodes and edges
... IDs on **nodes** and **edges**

Property Graphs



- ... attributes on nodes and edges
- ... IDs on nodes and edges
- ... labels on nodes and edges

POPULAR GRAPH DATABASES

Rank			DBMS	Database Model	Score		
Jun 2021	May 2021	Jun 2020			Jun 2021	May 2021	Jun 2020
1.	1.	1.	Oracle	Relational, Multi-model	1270.94	+1.00	-72.65
2.	2.	2.	MySQL	Relational, Multi-model	1227.86	-8.52	-50.03
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	991.07	-1.59	-76.24
4.	4.	4.	PostgreSQL	Relational, Multi-model	568.51	+9.26	+45.53
5.	5.	5.	MongoDB	Document, Multi-model	488.22	+7.20	+51.14
6.	6.	6.	IBM Db2	Relational, Multi-model	167.03	+0.37	+5.23
7.	7.	8.	Redis	Key-value, Multi-model	165.25	+3.08	+19.61
8.	8.	7.	Elasticsearch	Search engine, Multi-model	154.71	-0.65	+5.02
9.	9.	9.	SQLite	Relational	130.54	+3.84	+5.72
10.	10.	11.	Microsoft Access	Relational	114.94	-0.46	-2.24
11.	11.	10.	Cassandra	Wide column	114.11	+3.18	-4.90
12.	12.	12.	MariaDB	Relational, Multi-model	96.79	+0.10	+7.00
13.	13.	13.	Splunk	Search engine	90.27	-1.84	+2.19
14.	14.	14.	Hive	Relational	79.69	+3.51	+1.04
15.	15.	23.	Microsoft Azure SQL Database	Relational, Multi-model	74.79	+4.33	+27.01
16.	16.	16.	Amazon DynamoDB	Multi-model	73.76	+3.69	+8.90
17.	17.	15.	Teradata	Relational, Multi-model	69.34	-0.64	-3.95
18.	19.	22.	Neo4j	Graph	55.75	+3.52	+7.48
19.	18.	19.	SAP HANA	Relational, Multi-model	54.11	+1.36	+3.29
20.	20.	18.	Solr	Search engine, Multi-model	52.10	+0.91	+0.84
21.	22.	20.	FileMaker	Relational	49.13	+2.40	-1.02
22.	21.	17.	SAP Adaptive Server	Relational, Multi-model	48.99	-0.98	-4.11
23.	23.	21.	HBase	Wide column	43.52	+0.27	-5.22
24.	24.	26.	Google BigQuery	Relational	39.63	+2.00	+11.33
25.	25.	24.	Microsoft Azure Cosmos DB	Multi-model	36.47	+1.76	+5.67

include secondary database models

32 systems in ranking, June 2021

Rank			DBMS	Database Model	Score		
Jun 2021	May 2021	Jun 2020			Jun 2021	May 2021	Jun 2020
1.	1.	1.	Neo4j	Graph	55.75	+3.52	+7.48
2.	2.	2.	Microsoft Azure Cosmos DB	Multi-model	36.47	+1.76	+5.67
3.	3.	3.	ArangoDB	Multi-model	4.92	+0.53	-0.47
4.	4.	4.	OrientDB	Multi-model	4.45	+0.26	-0.37
5.	5.	5.	Virtuoso	Multi-model	3.69	+0.25	+1.41
6.	7.	7.	JanusGraph	Graph	2.49	+0.32	+0.48
7.	6.	9.	GraphDB	Multi-model	2.31	+0.06	+1.06
8.	8.	6.	Amazon Neptune	Multi-model	2.20	+0.26	+0.03
9.	9.	13.	TigerGraph	Graph	1.88	+0.14	+0.99
10.	10.	11.	Stardog	Multi-model	1.73	+0.02	+0.57
11.	12.	8.	Dgraph	Graph	1.71	+0.35	+0.31
12.	11.	10.	Fauna	Multi-model	1.67	+0.18	+0.48
13.	13.	14.	AllegroGraph	Multi-model	1.34	+0.01	+0.45
14.	14.	12.	Giraph	Graph	1.20	+0.07	+0.23
15.	15.	21.	Nebula Graph	Graph	1.04	+0.01	+0.79
16.	17.	15.	Blazegraph	Multi-model	0.82	+0.01	+0.15
17.	16.	16.	TypeDB	Multi-model	0.81	-0.02	+0.21
18.	18.	17.	Graph Engine	Multi-model	0.74	+0.04	+0.16
19.	19.	18.	InfiniteGraph	Graph	0.51	+0.00	+0.12
20.	21.	20.	FlockDB	Graph	0.35	+0.02	+0.08
21.	20.	19.	Fluree	Graph	0.35	+0.00	+0.06
22.	23.	27.	AnzoGraph DB	Multi-model	0.34	+0.05	+0.25
23.	22.	22.	HyperGraphDB	Graph	0.29	0.00	+0.07
24.	24.	25.	Memgraph	Graph	0.26	+0.03	+0.17
25.	27.	23.	TinkerGraph	Graph	0.18	+0.02	+0.03

NEO4J

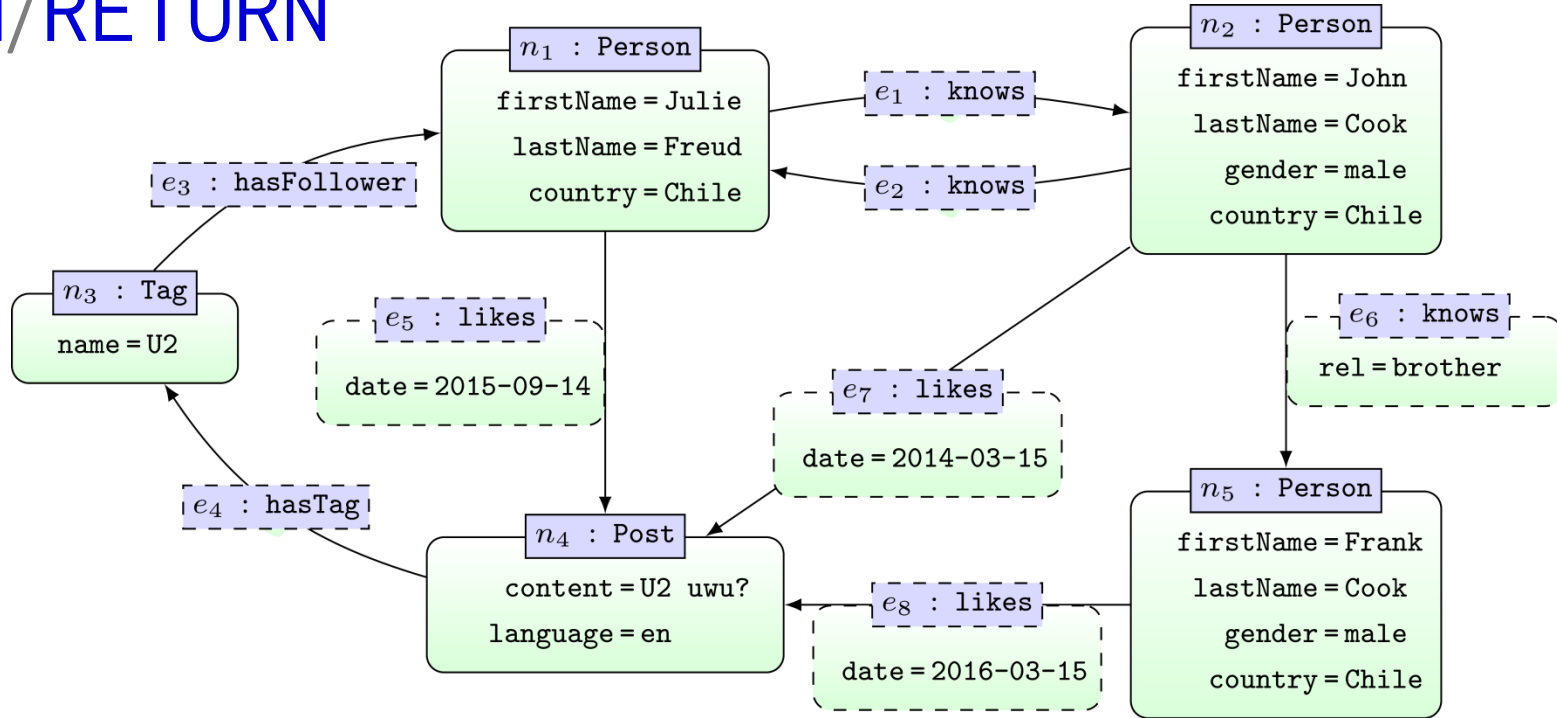
Neo4j Graph Database

- Data Model: [Property Graphs](#)
- Query Language: [Cypher](#)
- Scripting Language: [Gremlin](#)
- Licence: [Open Source \(Single Machine\)](#)
[Commercial \(Cluster Edition\)](#)



CYPHER: MATCH/RETURN

MATCH/RETURN

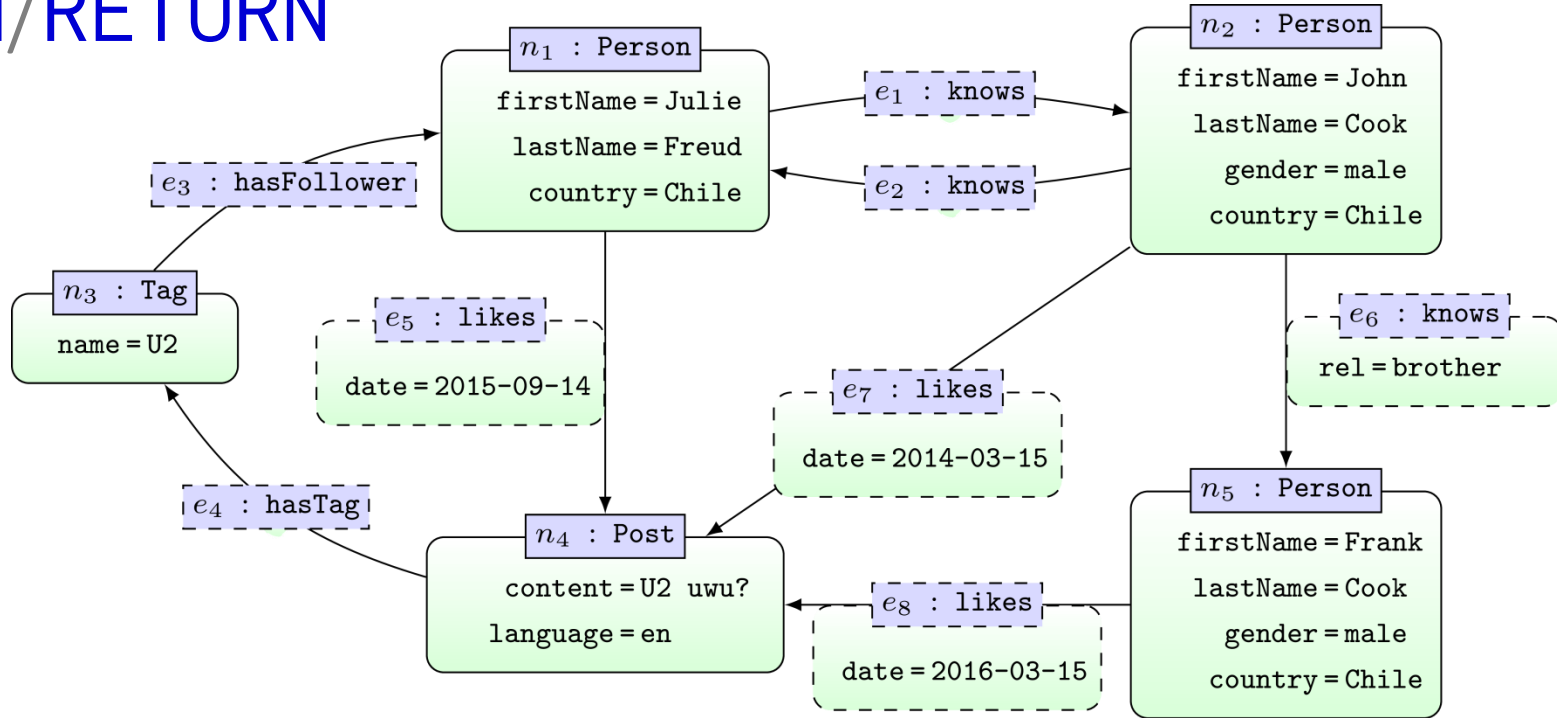


```
MATCH (x:Post)
RETURN x
```

x

```
(:Post {content: "U2 uwu?", language: "en"})
```

MATCH/RETURN

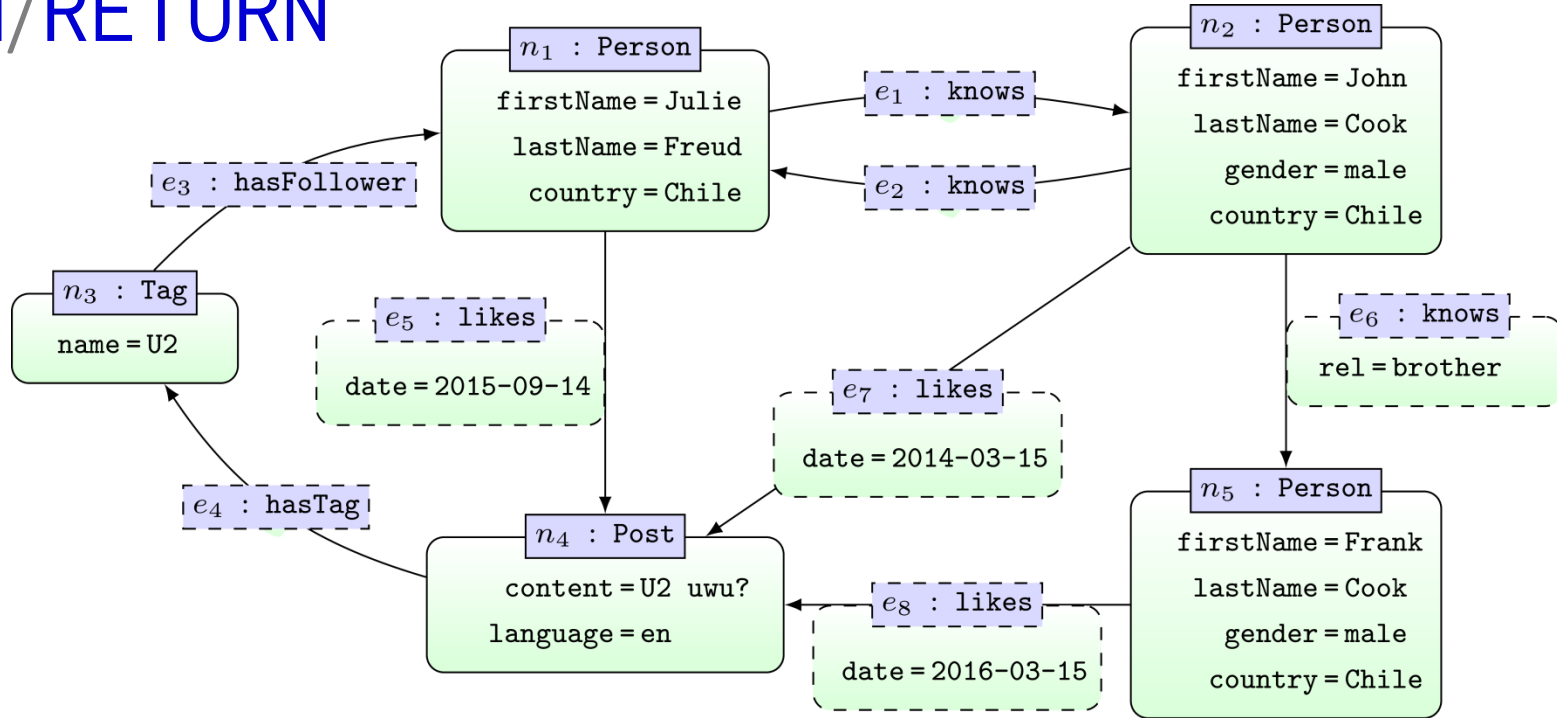


```
MATCH (x:Person)
RETURN x.firstName
```

x.firstName

Julie
John
Frank

MATCH/RETURN

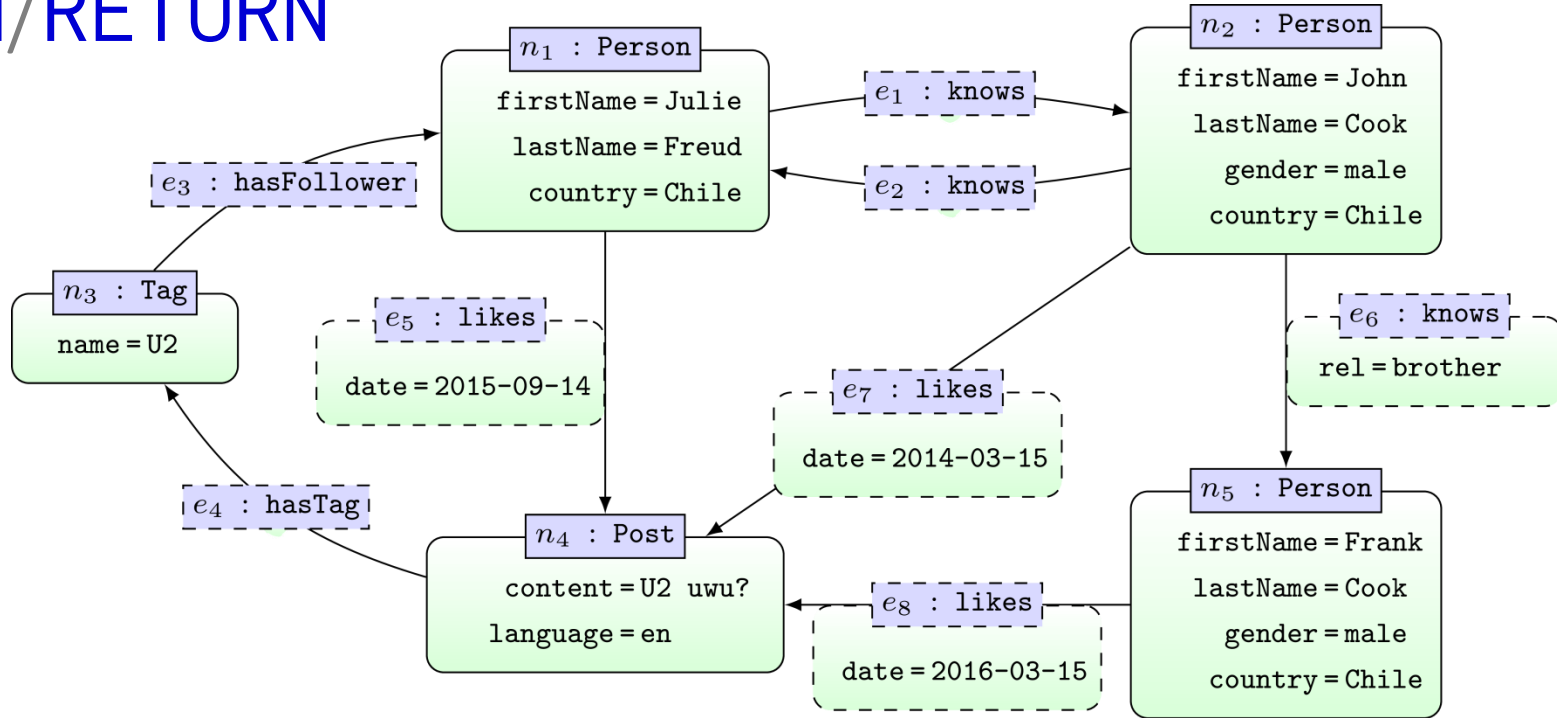


```
MATCH (x:Person {gender: "male", lastName: "Cook"})  
RETURN x.firstName
```

x.firstName

John
Frank

MATCH/RETURN



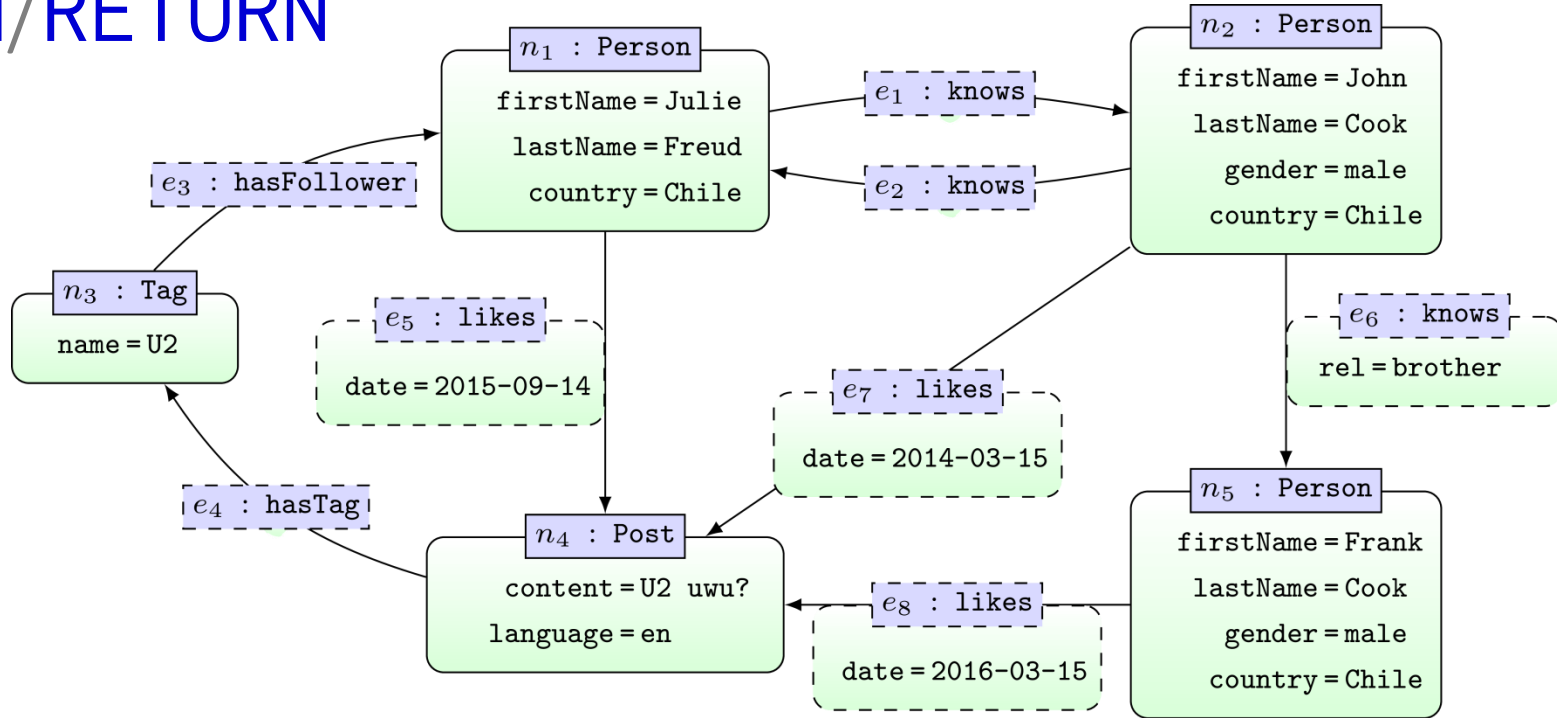
```

MATCH (x:Person)
RETURN x.firstName, x.gender
  
```

x.firstName	x.gender
Julie	
John	male
Frank	male

... matching nodes returned with blank attributes

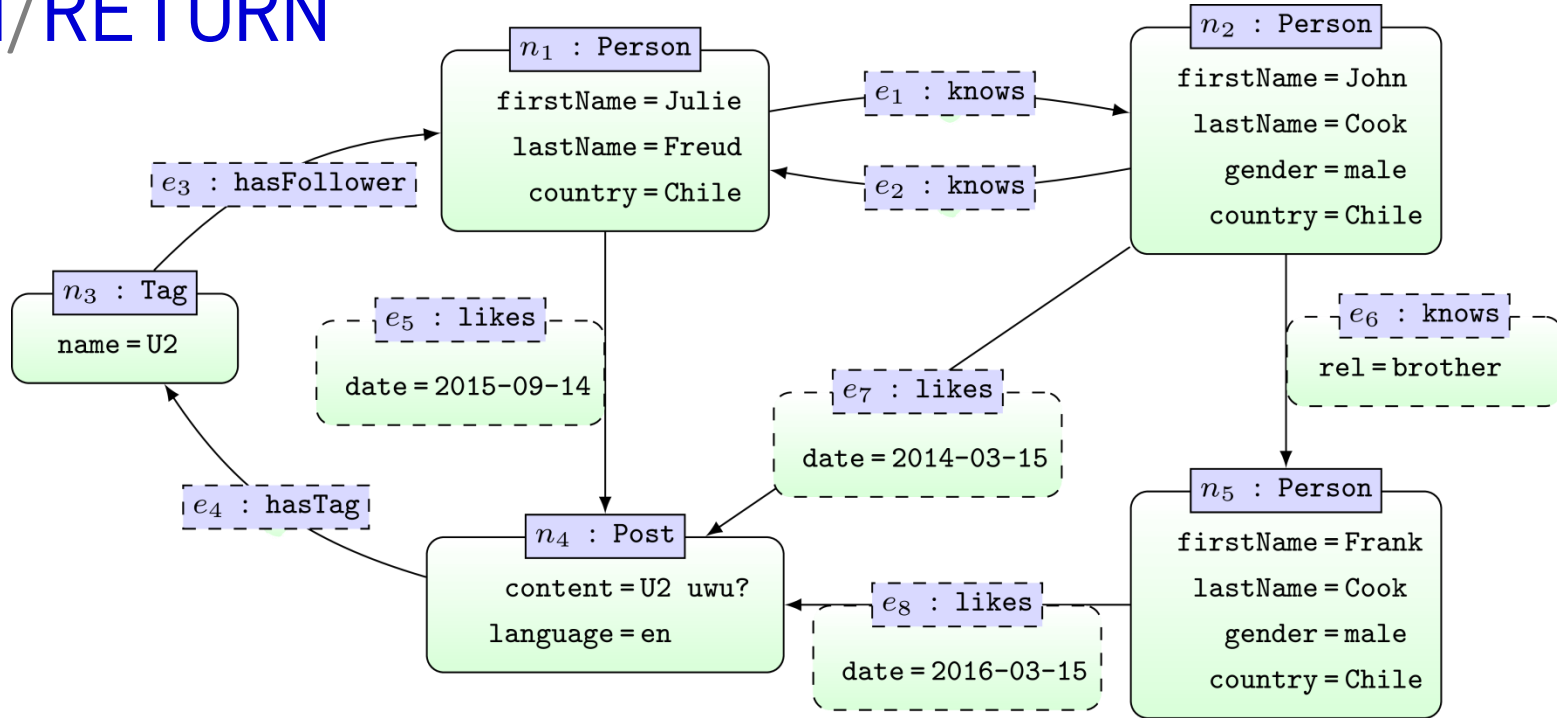
MATCH/RETURN



```
MATCH (x)
RETURN x.firstName, x.gender
```

x.firstName	x.gender
Julie	
John	male
Frank	male

MATCH/RETURN

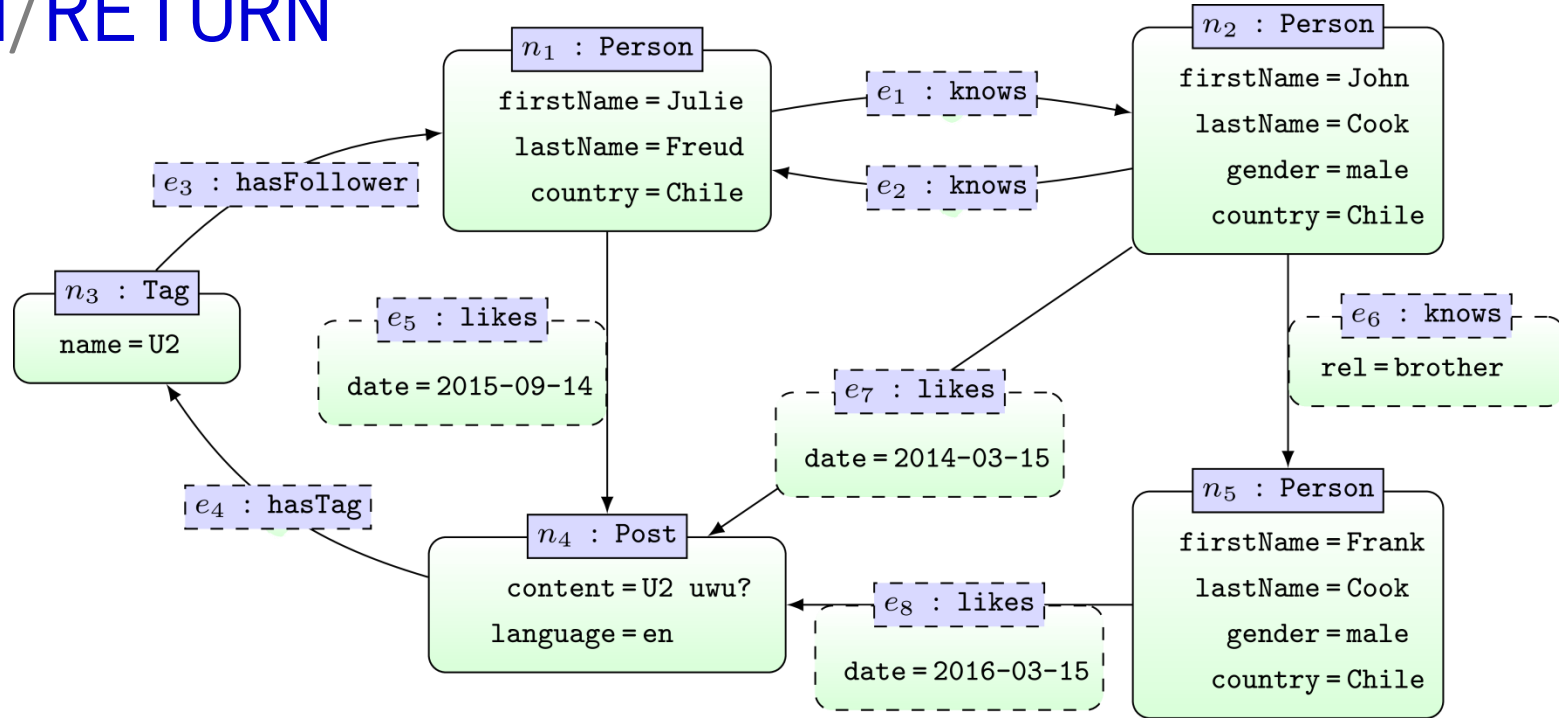


```
MATCH (:Person)-->(x:Person)
RETURN x.firstName
```

x.firstName

Julie
John
Frank

MATCH/RETURN

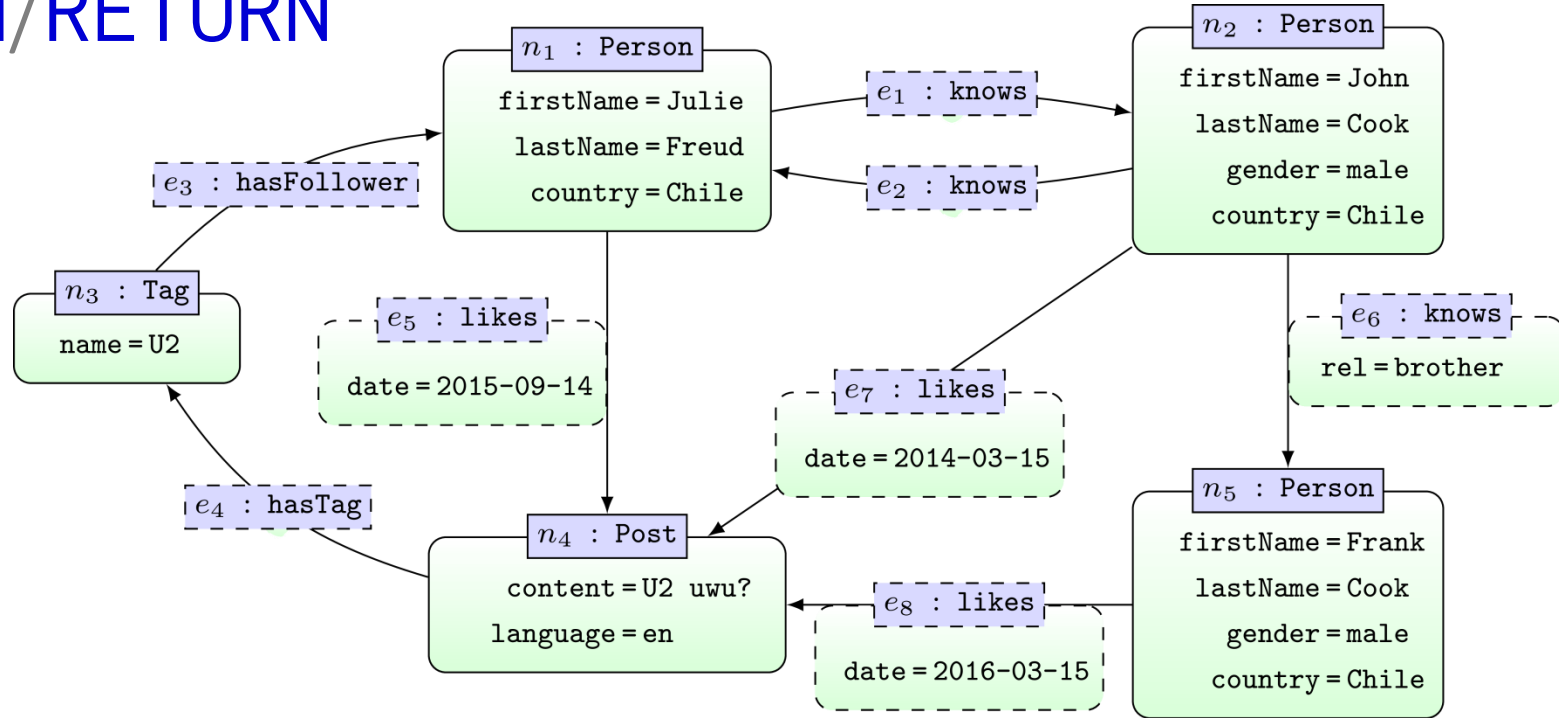


```
MATCH (x:Person)-->(y:Person)
RETURN x.firstName
```

```
-----
x.firstName
-----
Julie
John
John
-----
```

... multiplicity of results corresponds to number of matches

MATCH/RETURN



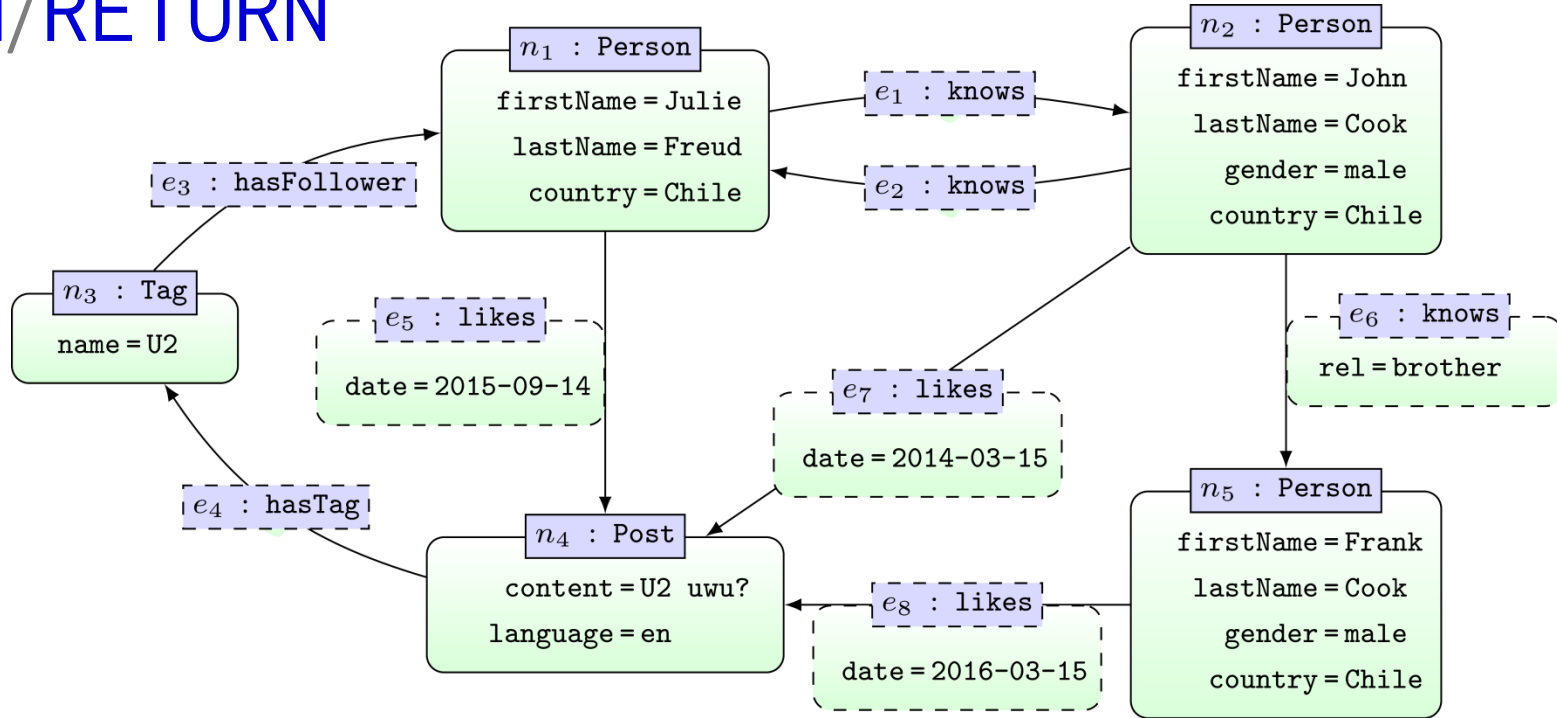
```
MATCH (x:Person)-->()  
RETURN x.firstName
```

x.firstName

Julie
Julie
John
John
John
Frank

... multiplicity of results corresponds to number of matches

MATCH/RETURN



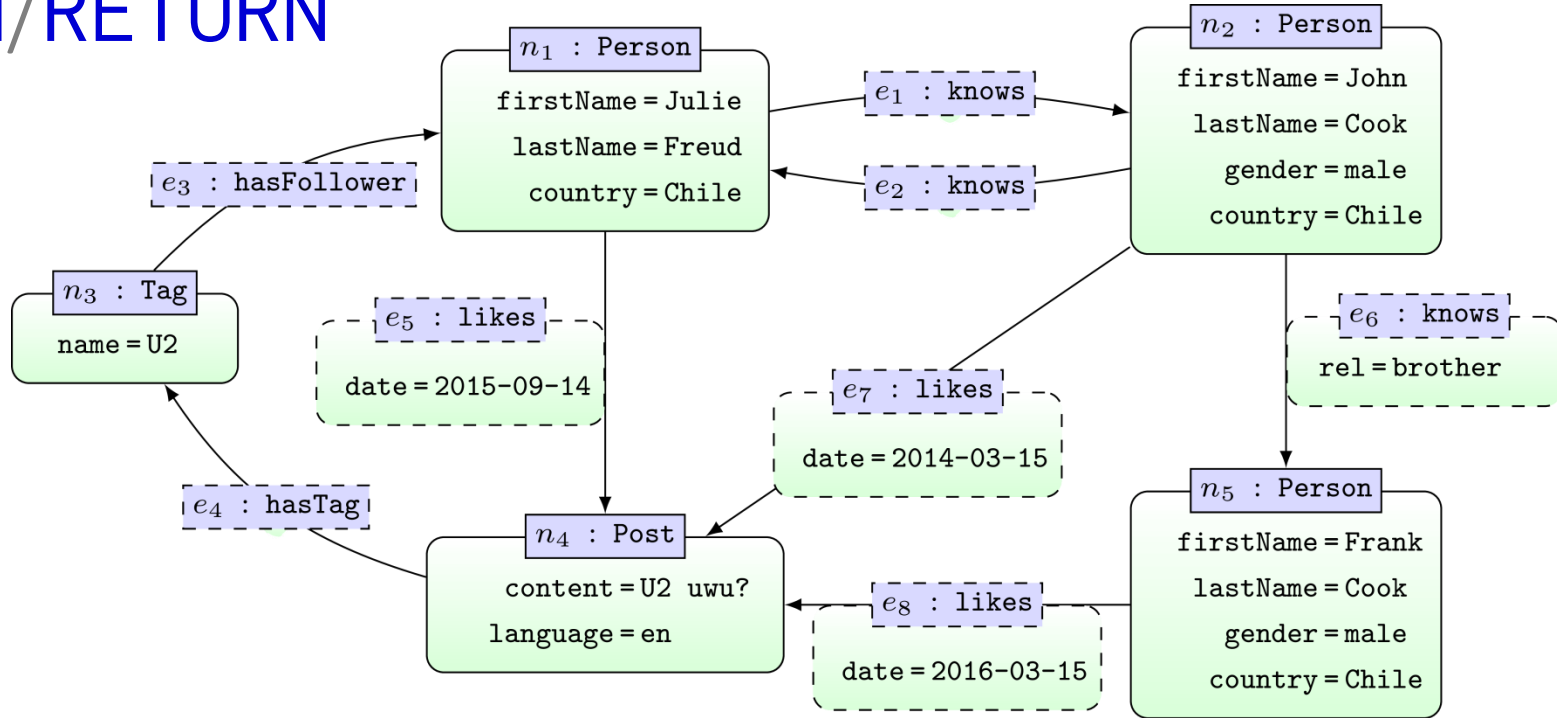
```
MATCH (x:Person)-->()  
RETURN DISTINCT x.firstName
```

```
x.firstName
```

```
Julie  
John  
Frank
```

... RETURN DISTINCT removes duplicates

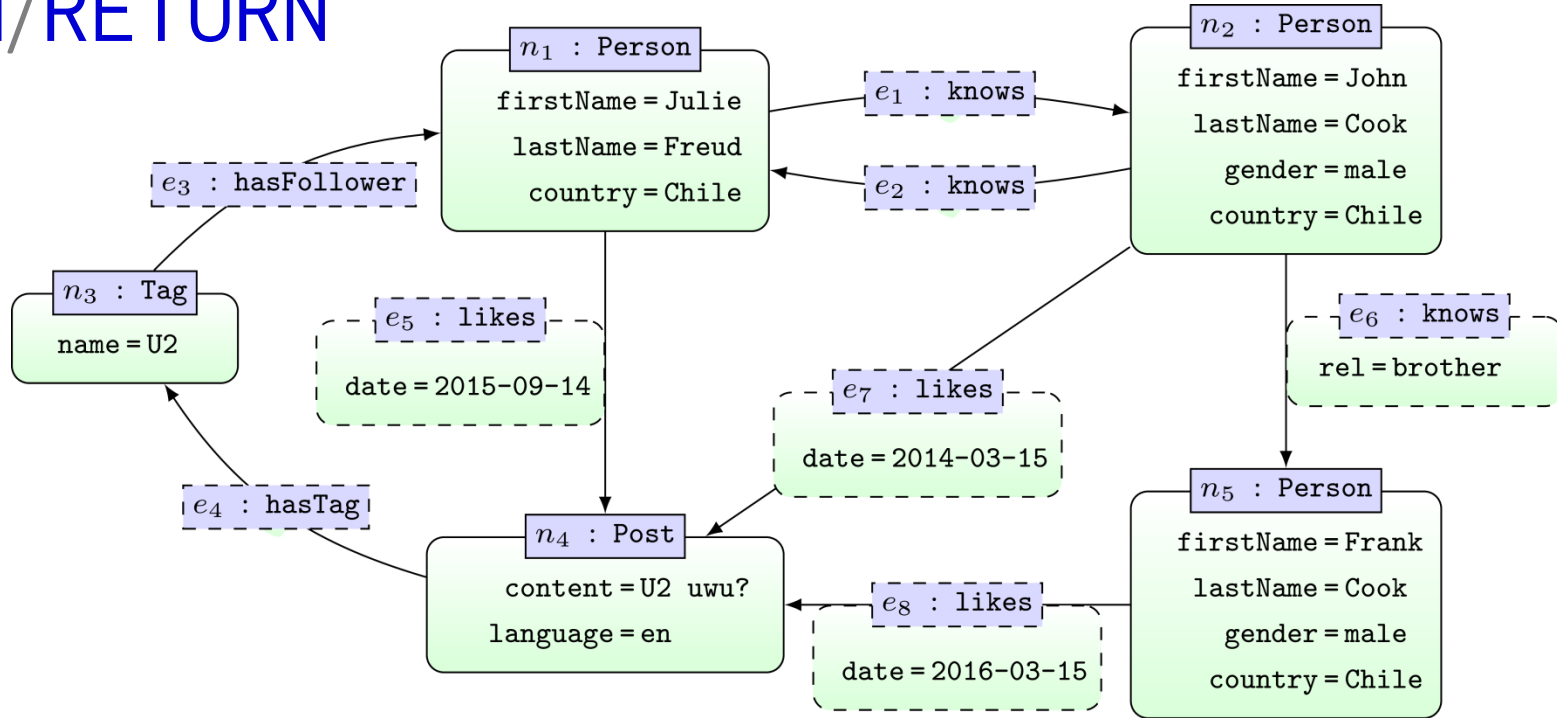
MATCH/RETURN



```
MATCH (x1:Person)-->(x2:Person)
RETURN x1.firstName,x2.firstname
```

x1.firstName	x2.firstName
Julie	John
John	Julie
John	Frank

MATCH/RETURN

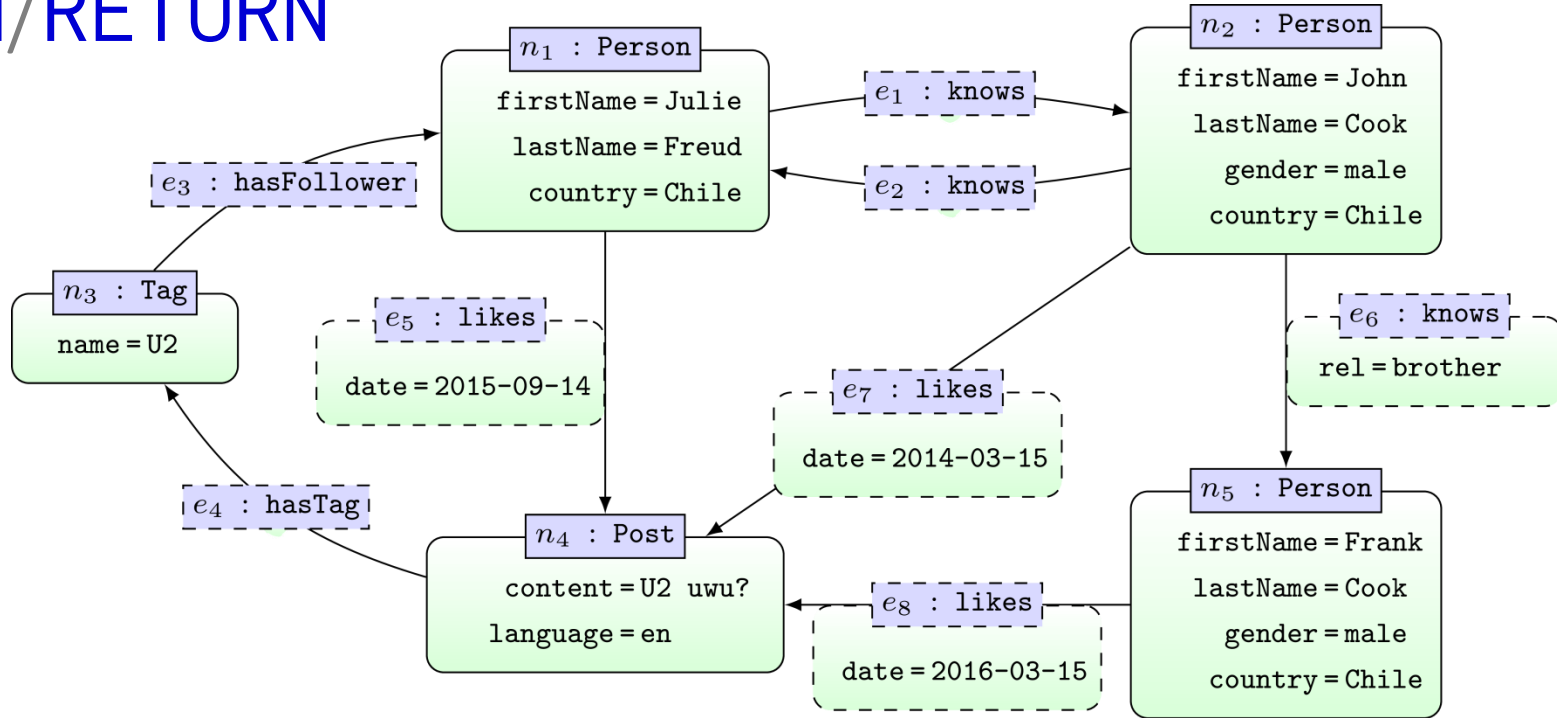


```

MATCH (x1:Person)-[r]->(x2:Person)
RETURN x1.firstName,x2.firstName,r.rel
  
```

x1.firstName	x2.firstName	r.rel
Julie	John	
John	Julie	
John	Frank	brother

MATCH/RETURN

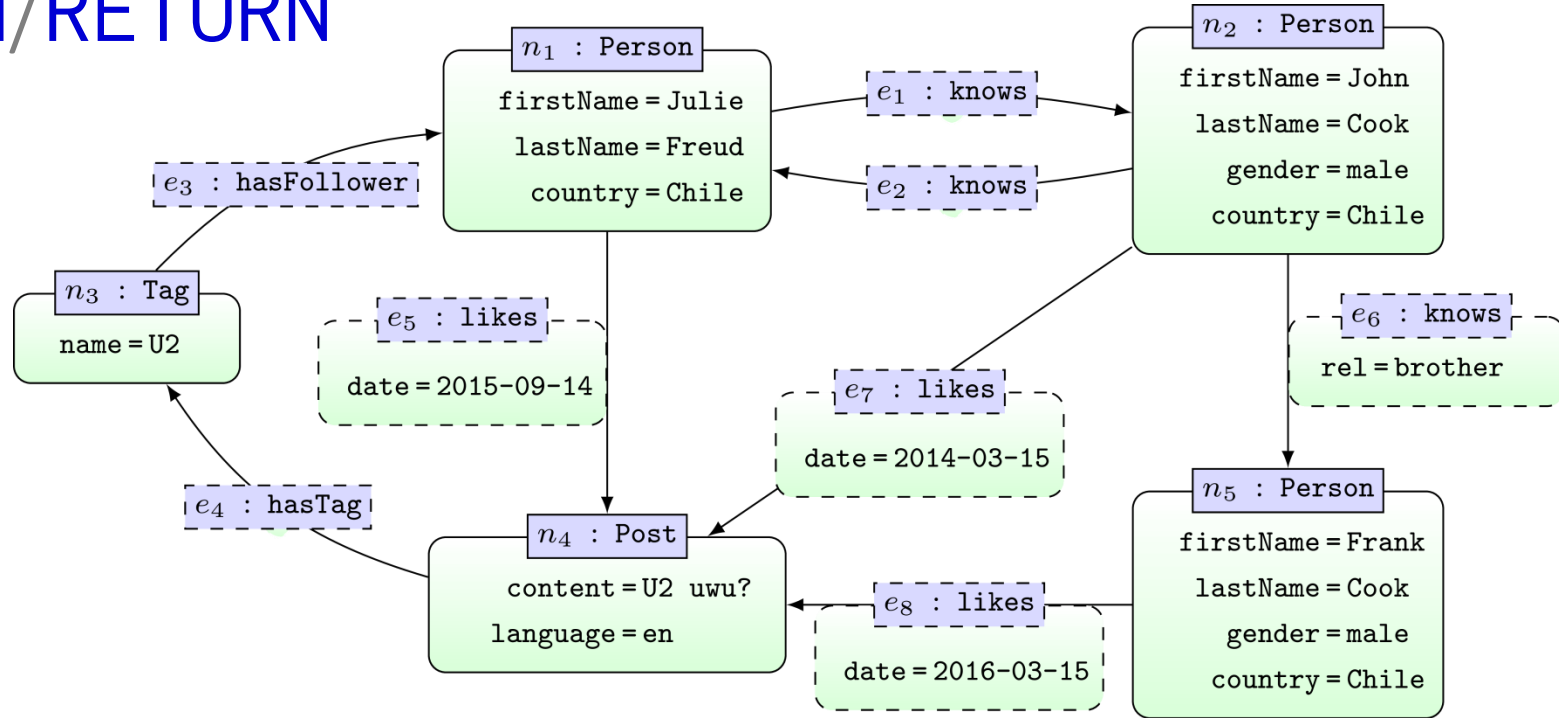


```
MATCH (x1:Person)-[r]->(x2:Person)
RETURN r
```

```
r
```

```
[:knows]
[:knows]
[:knows {rel: "brother"}]
```

MATCH/RETURN



```
MATCH ()<-[:knows]-(y)-[:knows]->()  
RETURN y.firstName
```

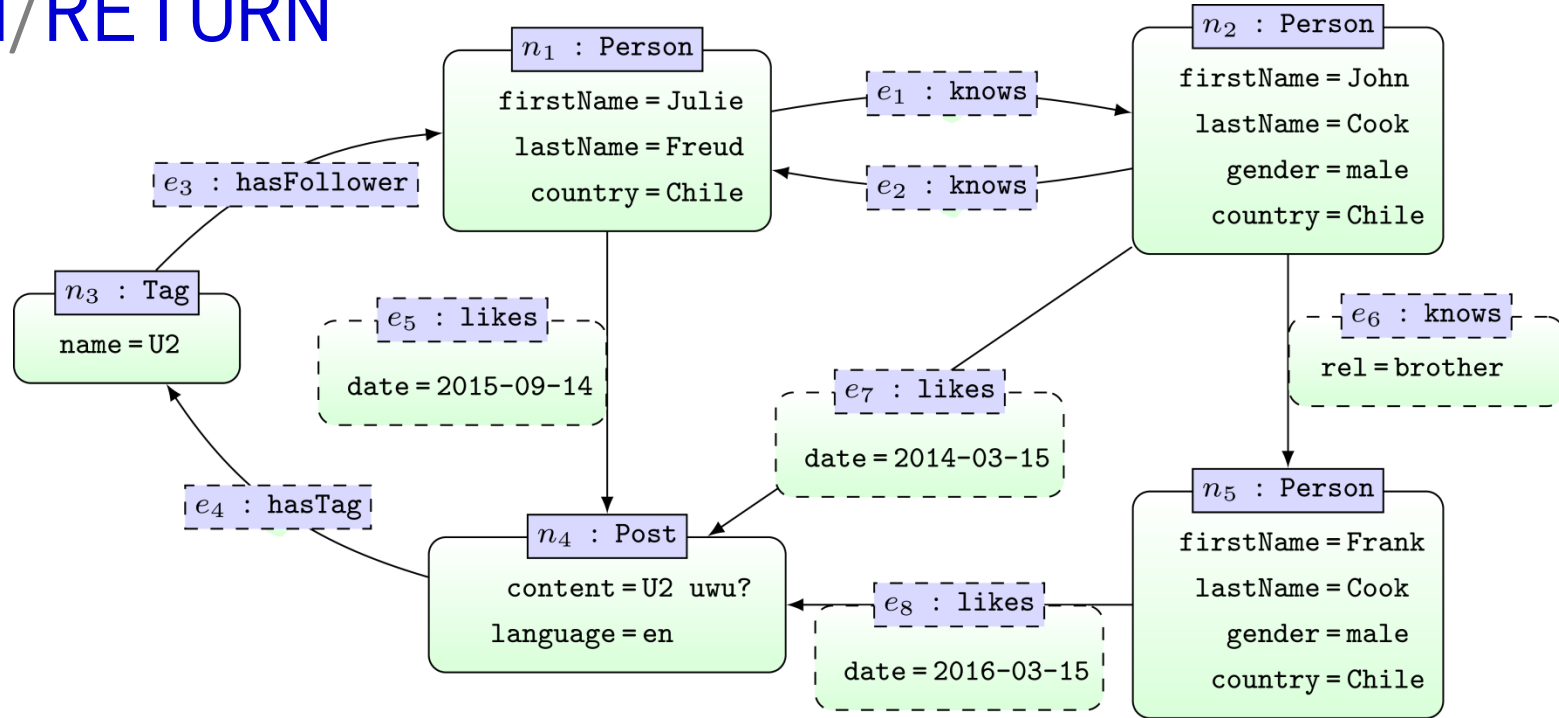
y.firstName

John

John

... MATCH will not match the same edge twice

MATCH/RETURN



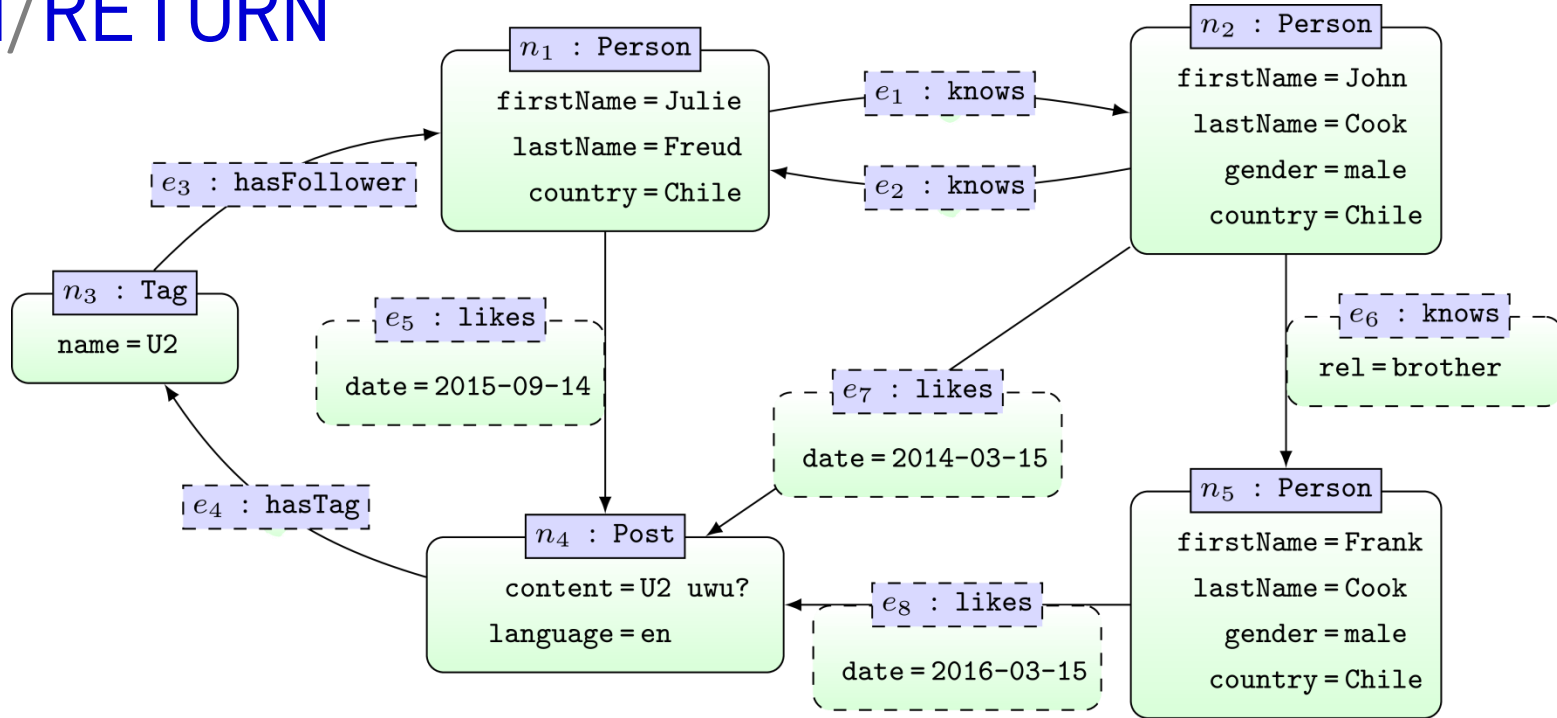
```
MATCH ()-[:knows]->(y)-[:knows]->()  
RETURN y.firstName
```

y.firstName

Julie
John
John

... MATCH will match same node twice

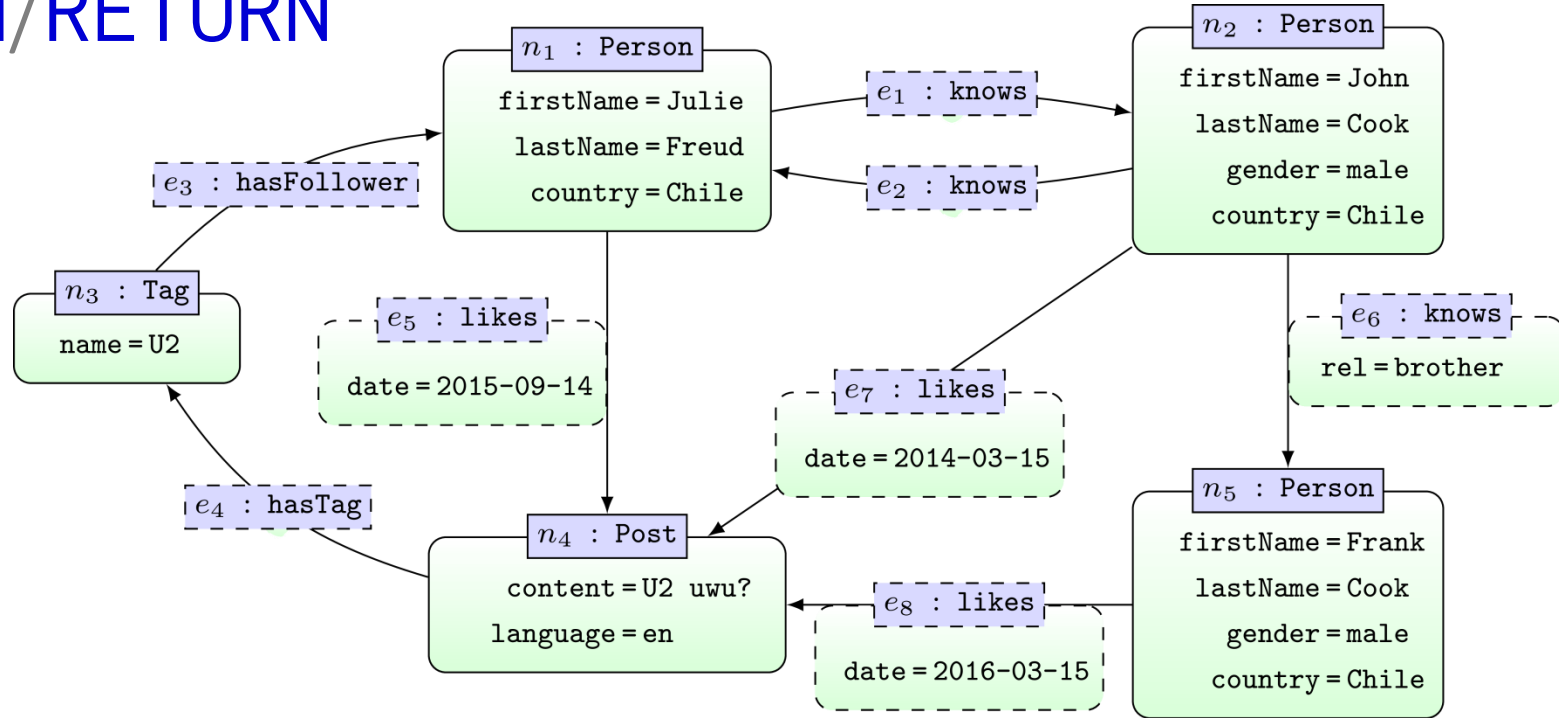
MATCH/RETURN



```
MATCH (x:Person)-->()->()->(x)
RETURN x.firstName
```

```
-----
x.firstName
-----
Julie
-----
```

MATCH/RETURN

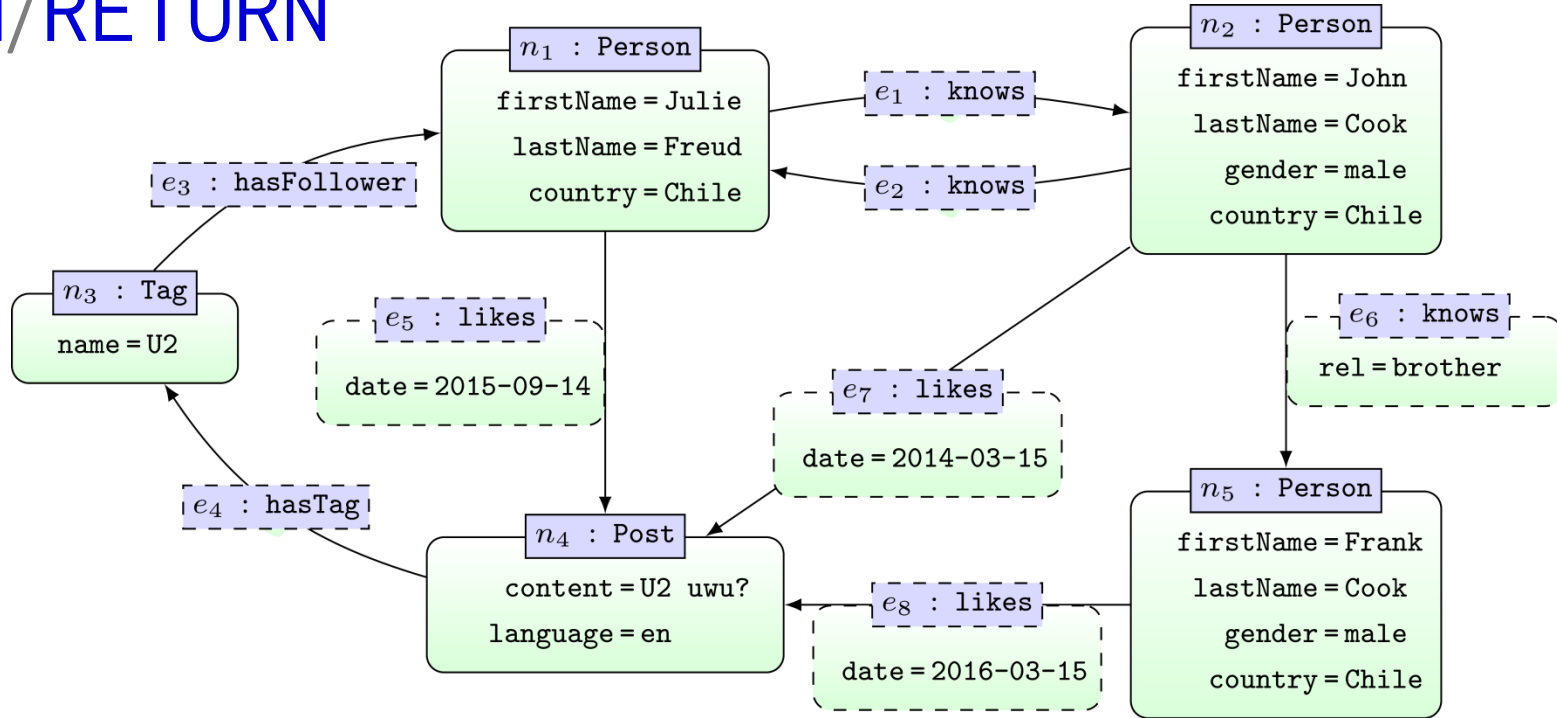


```
MATCH (x1)-[:likes]->(y)<-[:likes]-(x2)
RETURN x1.firstName AS n1, x2.firstName AS n2
```

n1	n2
Julie	John
John	Julie
John	Frank
Frank	John
Frank	Julie
Julie	Frank

... AS renames columns in results

MATCH/RETURN



```

MATCH (x1)-[:likes]->(y)
MATCH (y)<-[:likes]-(x2)
RETURN x1.firstName AS n1, x2.firstName AS n2
    
```

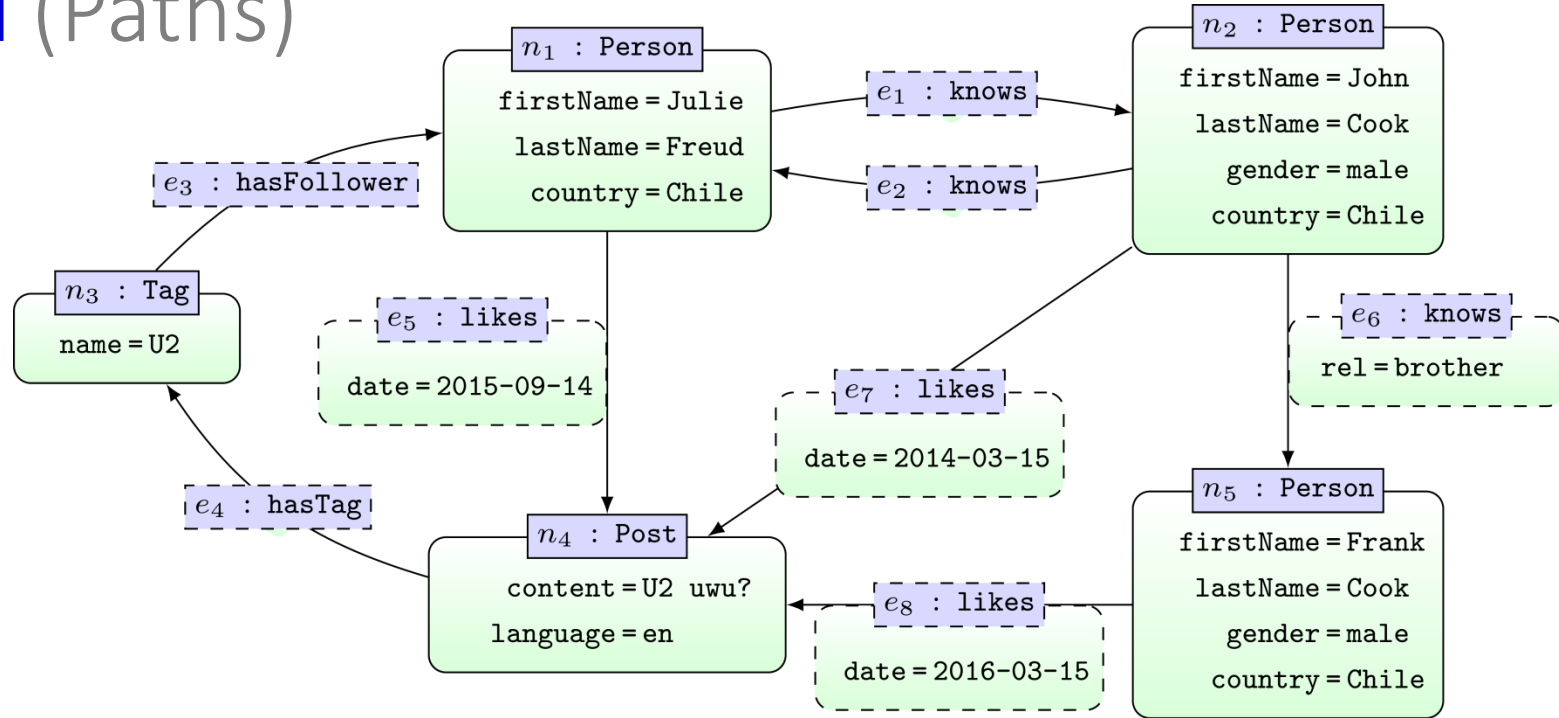
n1	n2
Julie	John
John	Julie
Julie	Julie
John	Frank
...	...

... use multiple **MATCH** to match same edge multiple times

CYPHER:

MATCH (PATHS)

MATCH (Paths)

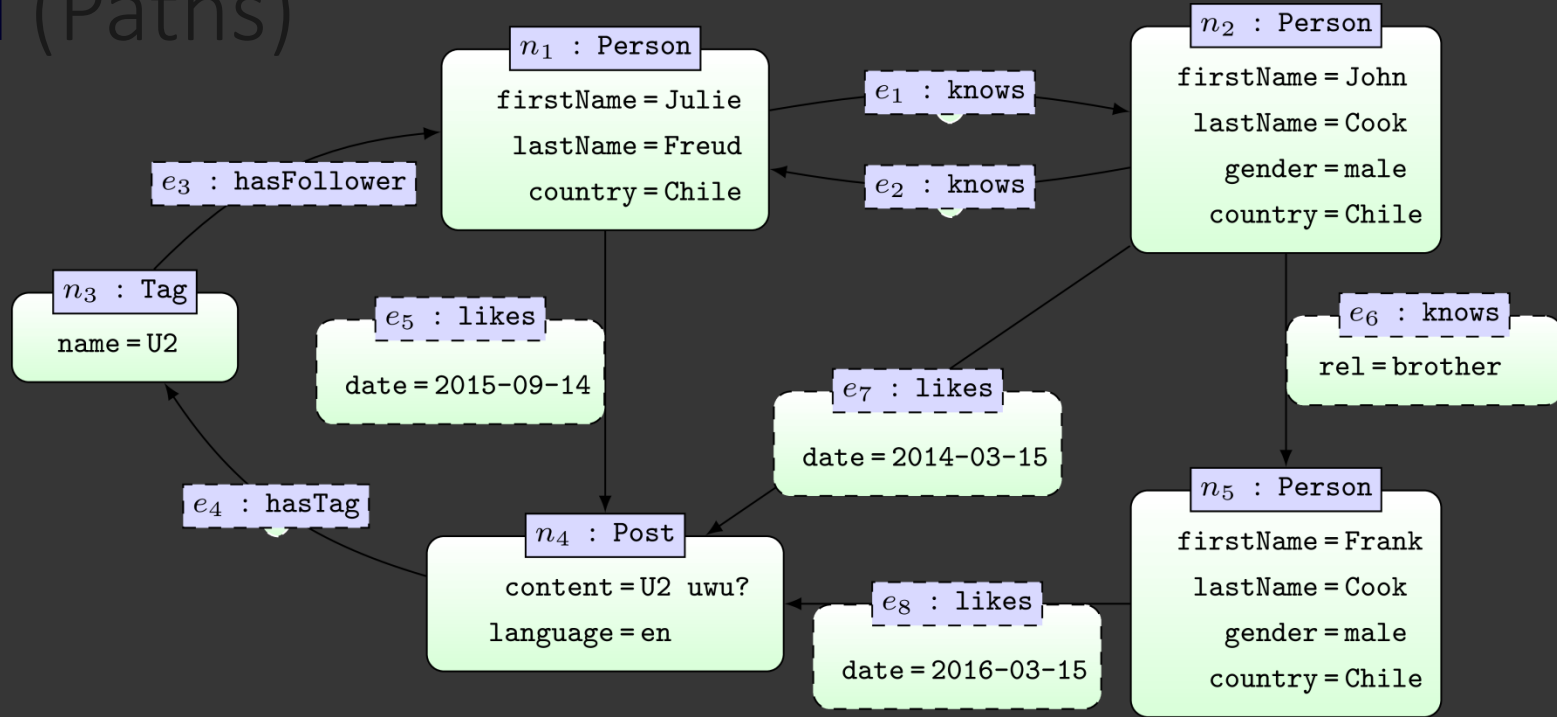


```
MATCH (x1)-[:knows*]->(x2)
RETURN x1.firstName, x2.firstName
```

x1.firstName	x2.firstName
Julie	John
Julie	Frank
Julie	Julie
John	Julie
John	John
John	Frank
John	Frank

... paths of length one-or-more

MATCH (Paths)



```

MATCH (x1)-[:knows*]->(x2)
RETURN x1.firstName, x2.firstName
  
```

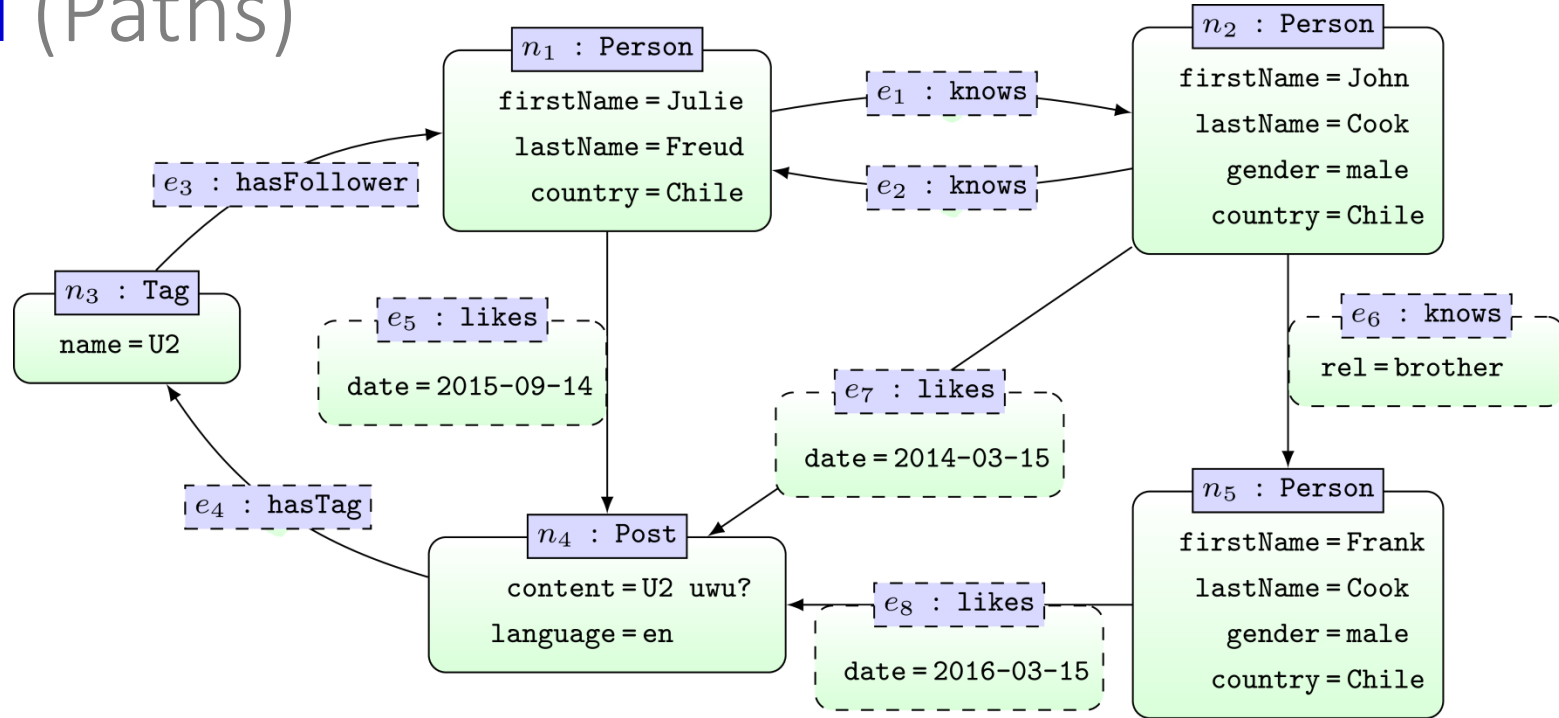
x1.firstName	x2.firstName
Julie	John
Julie	Frank
Julie	Julie
John	Julie
John	John
John	Frank
John	Frank

Otherwise ...

... we could have infinite paths!

... paths visit each edge at most once!

MATCH (Paths)

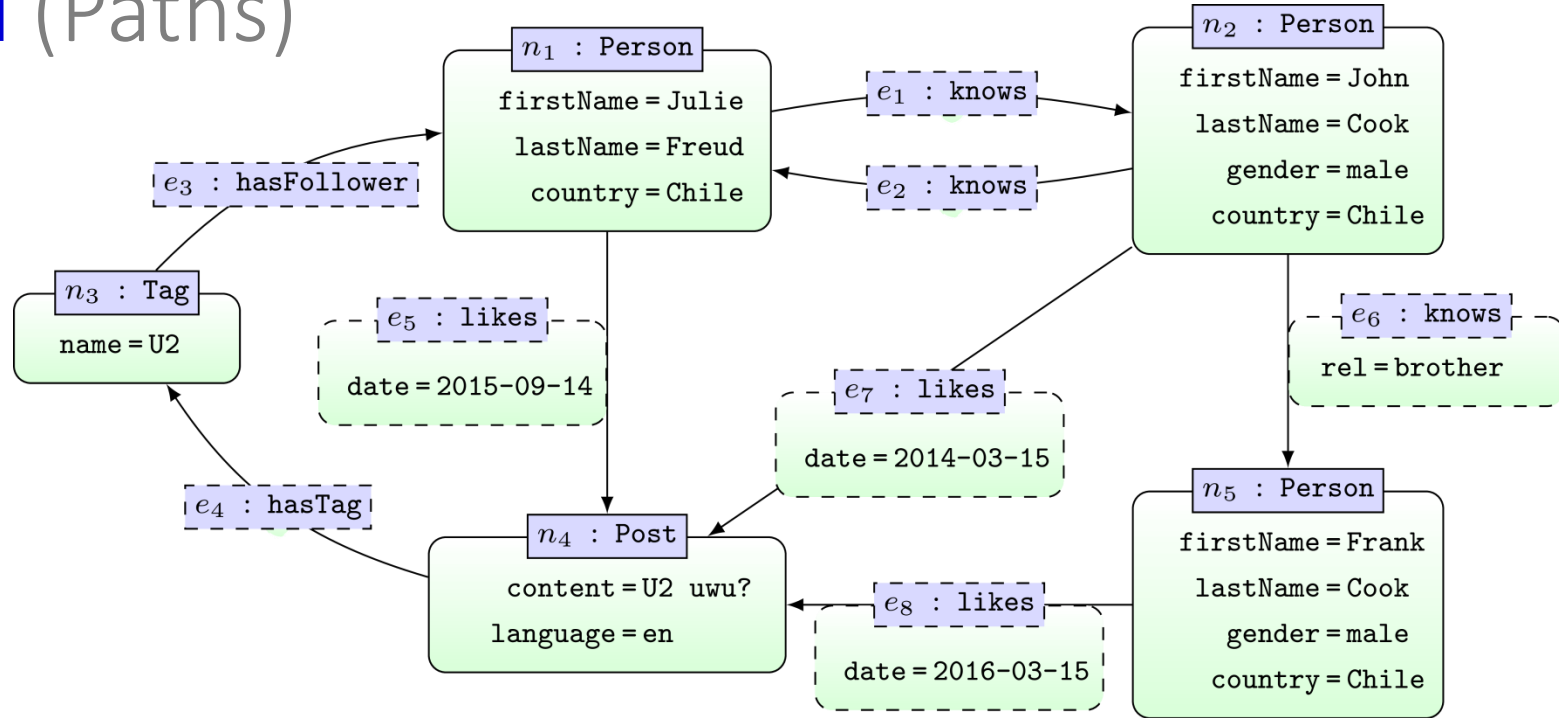


```
MATCH (x1)-[:knows*3]->(x2)
RETURN x1.firstName, x2.firstName
```

x1.firstName	x2.firstName
John	Frank

... can set minimum path length (no. of nodes visited)

MATCH (Paths)

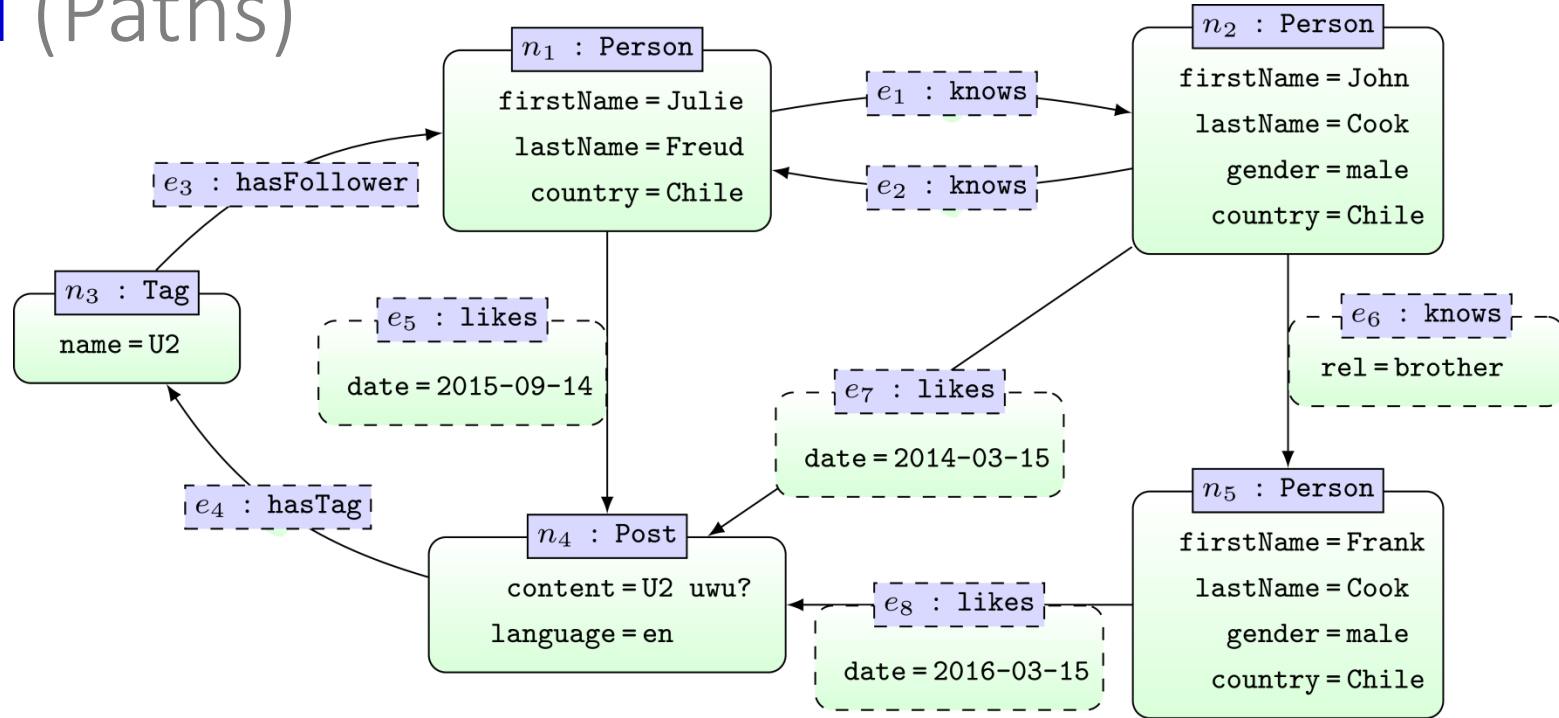


```
MATCH (x1)-[:knows*2..3]->(x2)
RETURN x1.firstName, x2.firstName
```

x1.firstName	x2.firstName
Julie	Frank
Julie	Julie
John	Frank
John	John

... or range of path length

MATCH (Paths)

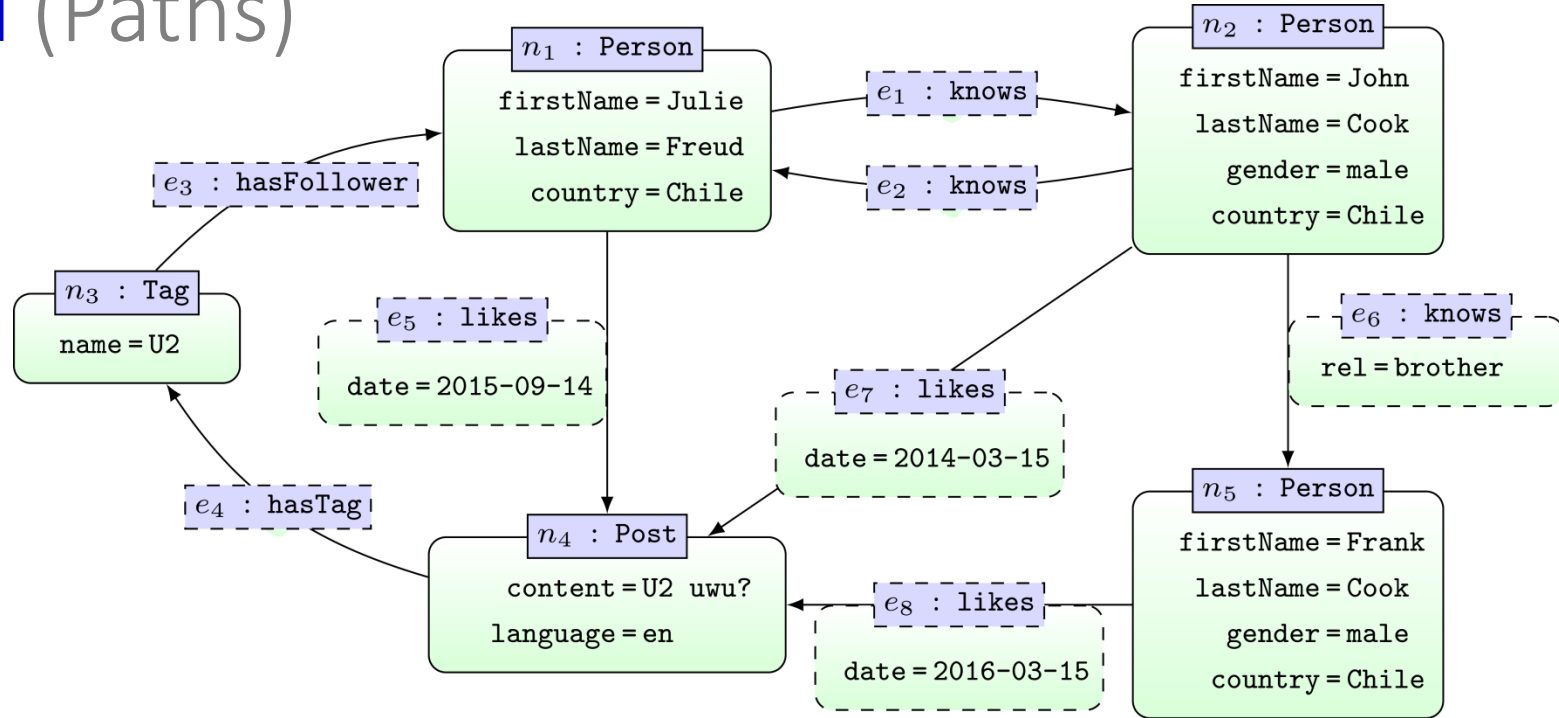


```
MATCH (x1)-[:knows*..2]->(x2)
RETURN x1.firstName, x2.firstName
```

x1.firstName	x2.firstName
Julie	John
Julie	Frank
Julie	Julie
John	Julie
John	John
John	Frank

... or maximum path length

MATCH (Paths)

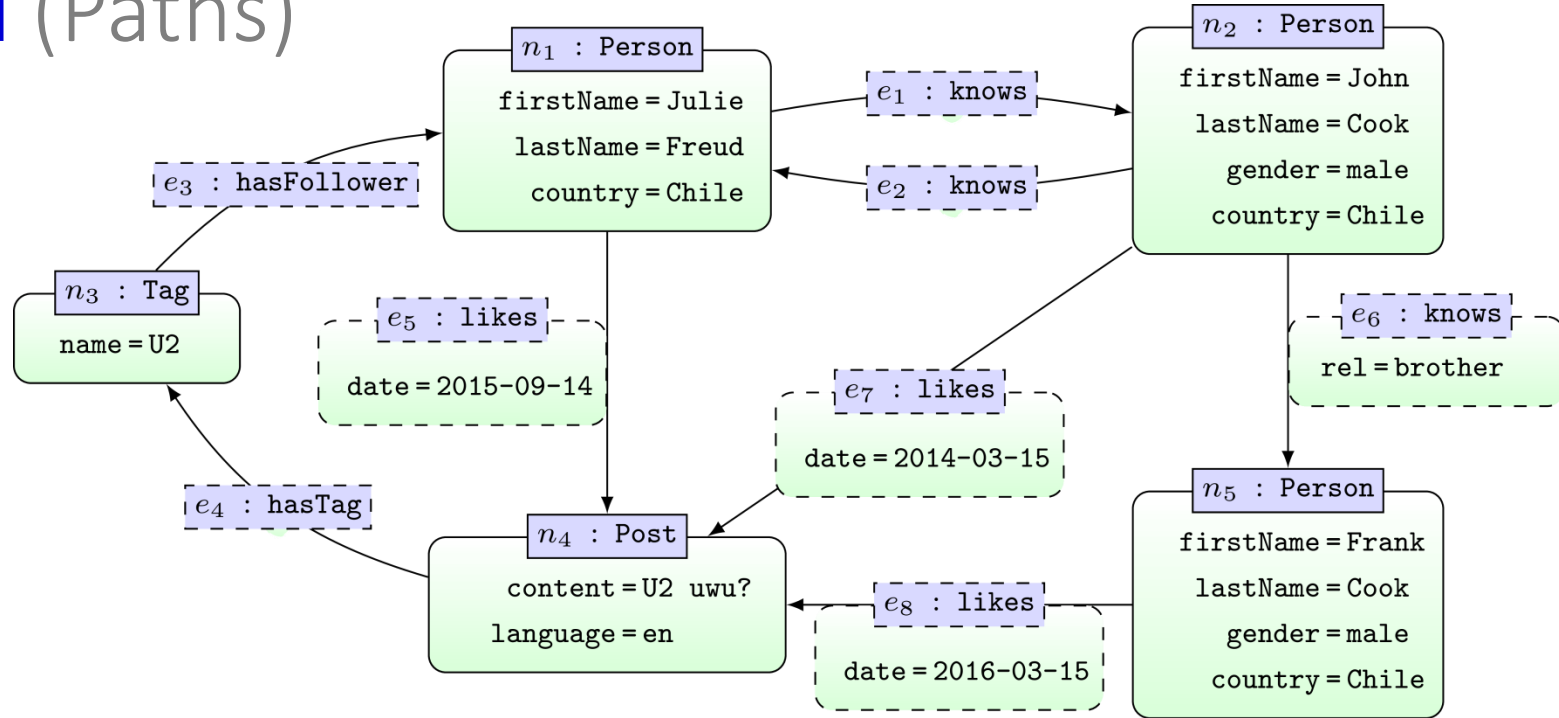


```
MATCH (x1)-[:knows*0..1]->(x2)
RETURN x1.firstName, x2.firstName
```

x1.firstName	x2.firstName
Julie	Julie
Julie	John
John	John
John	Julie
John	Frank
Frank	Frank

... 0-length path is the node itself; will match any node

MATCH (Paths)



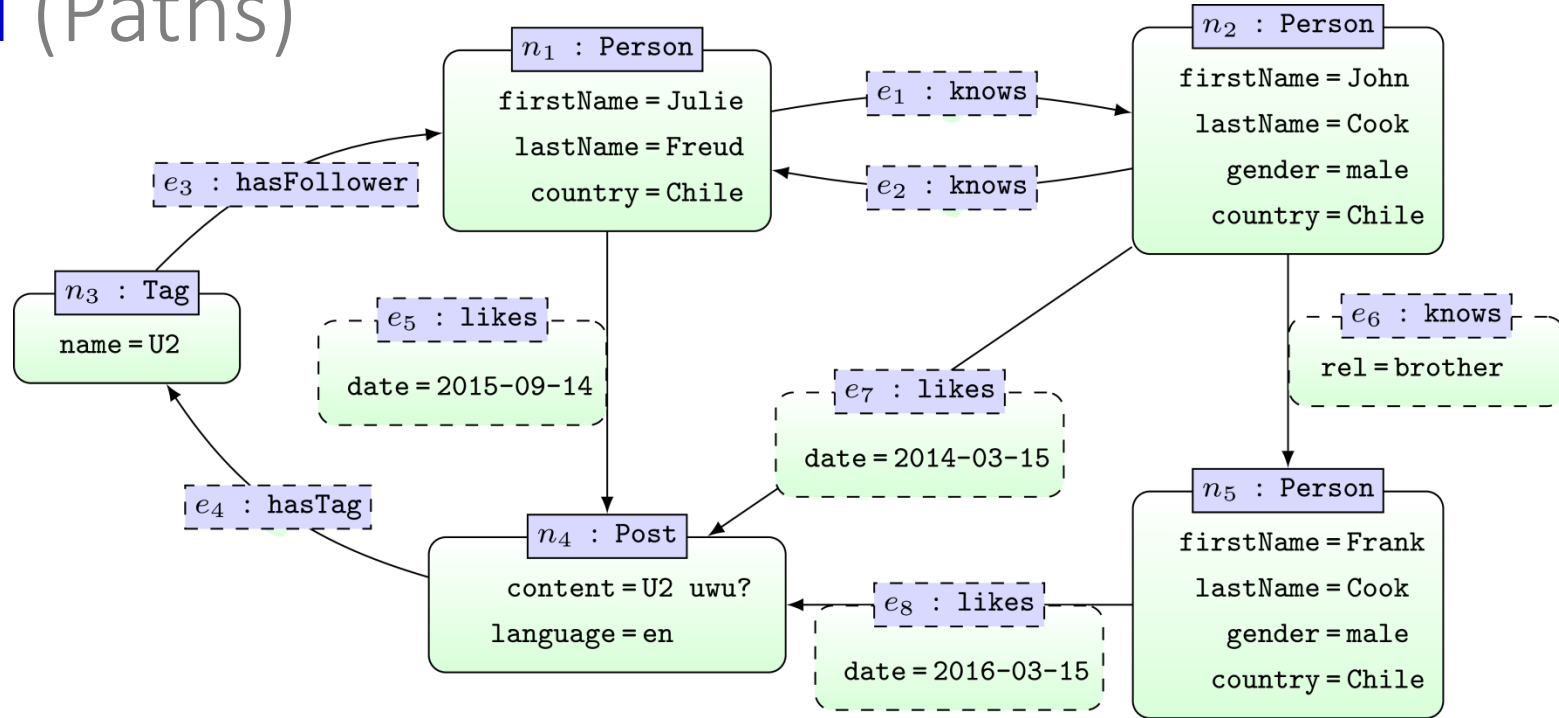
```
MATCH p = (x1)-[:knows*3]->(x2)
RETURN p
```

P

```
(:Person {firstName:"John",...})-[:knows]->(:Person {firstName:"Julie",...})-[:knows]->(:Person {firstName:"John",...})-[:knows rel:"brother"]->(:Person {firstName:"Frank",...})
```

... can return a full path

MATCH (Paths)

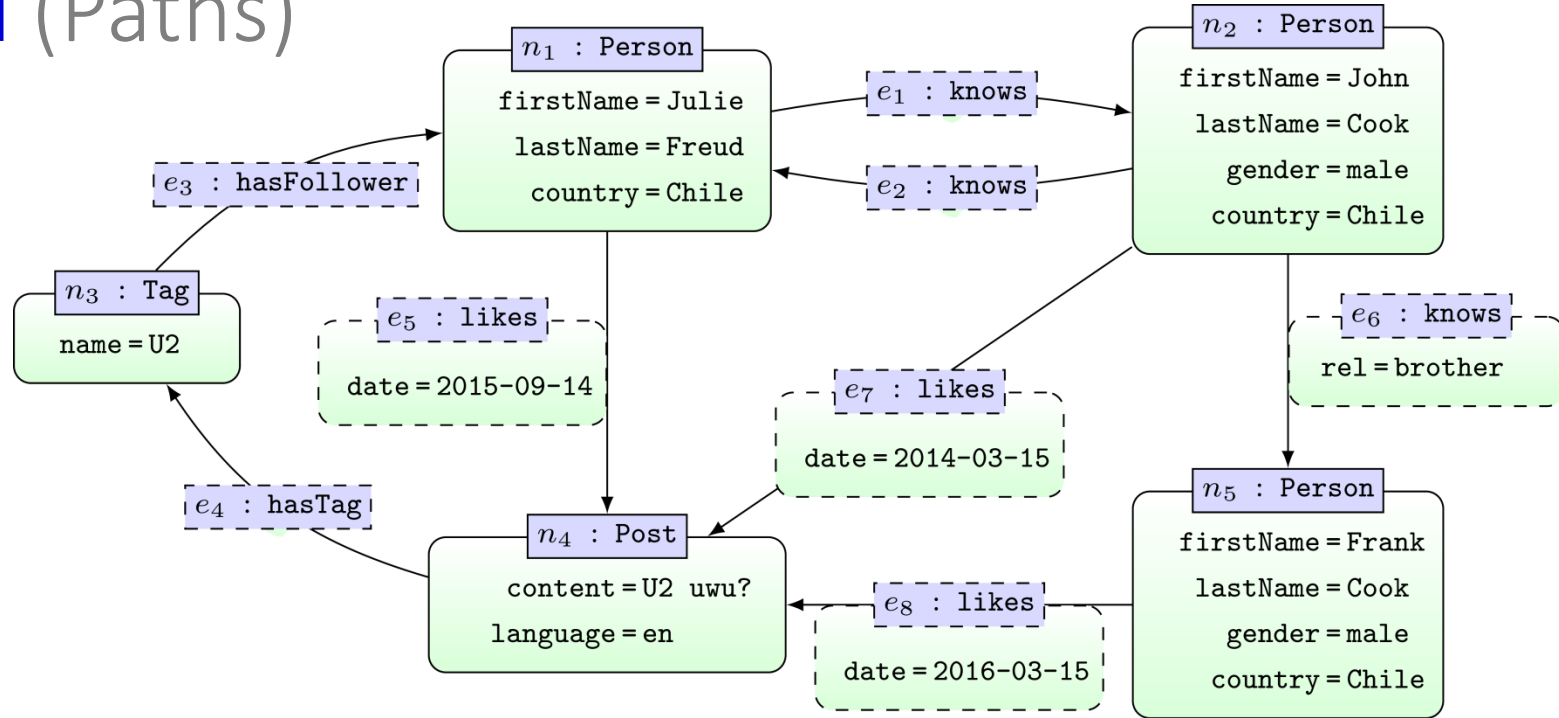


```
MATCH p = (x1)-[:knows*3]->(x2)
RETURN p
```

```
p
(:Person {firstName:"John",...})-[:knows]->(:Person {firstName:"Julie",...})-[:knows]->
(:Person {firstName:"John",...})-[:knows rel:"brother"]->(:Person {firstName:"Frank",...})
```

... can return a full path

MATCH (Paths)



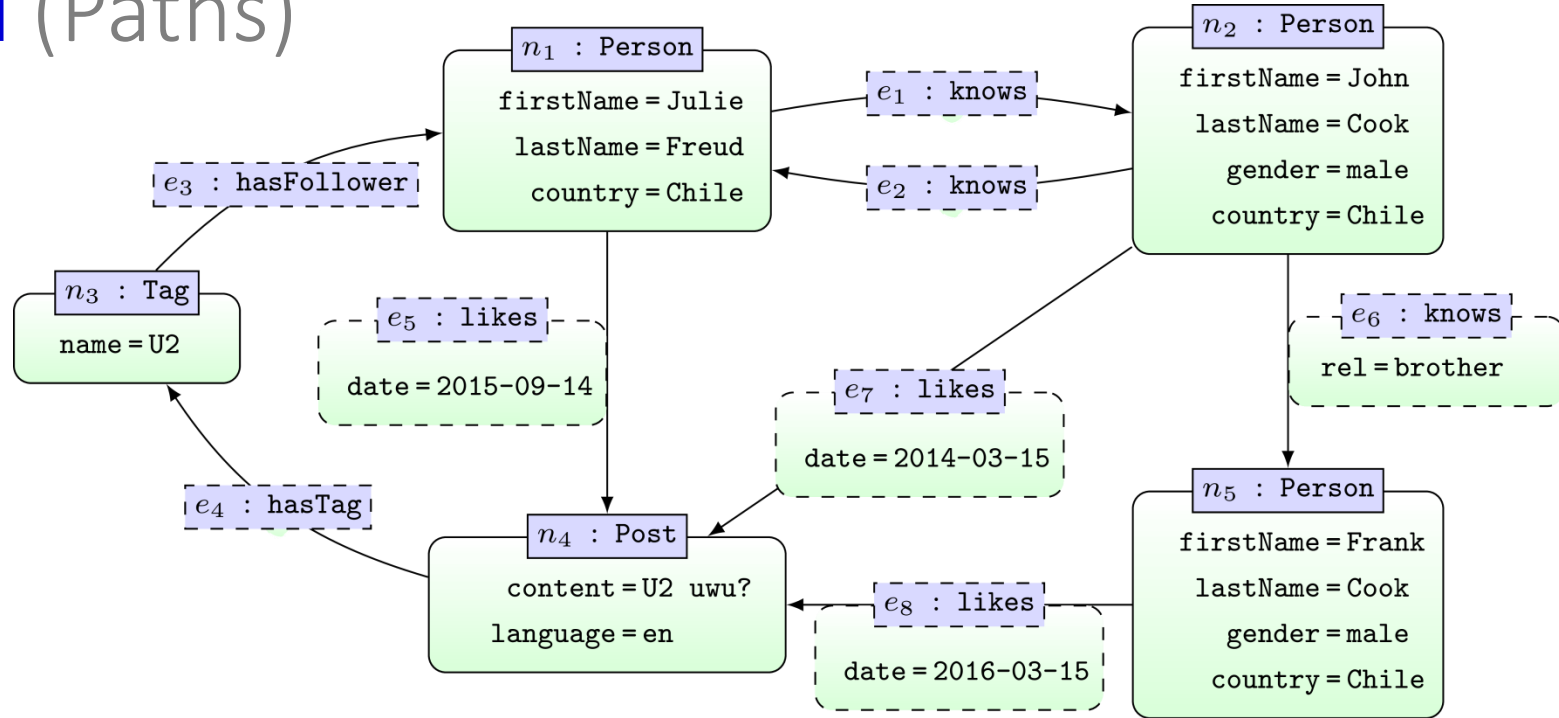
```
MATCH p = (x1)-[:knows*3]->(x2)
RETURN p
```

p

n₂ · e₂ · n₁ · e₁ · n₂ · e₆ · n₅

... can return a full path

MATCH (Paths)



```

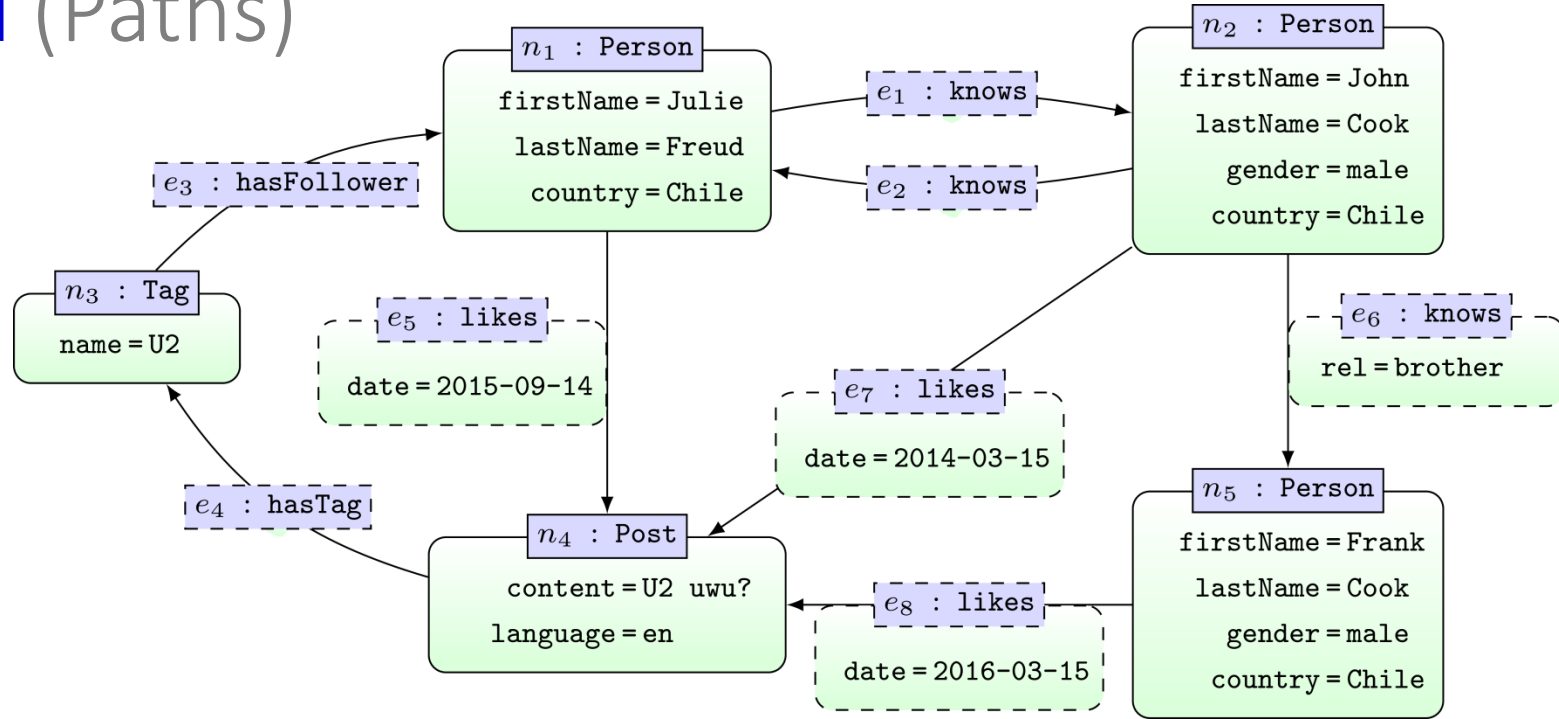
MATCH (f {firstName: 'Frank'}),
      (j {firstName: 'Julie'}),
      p = shortestPath((f)-[*]-(j))
RETURN p
  
```

p

n₅ · e₆ · n₂ · e₂ · n₁

... returns any shortest path (matching criteria)

MATCH (Paths)



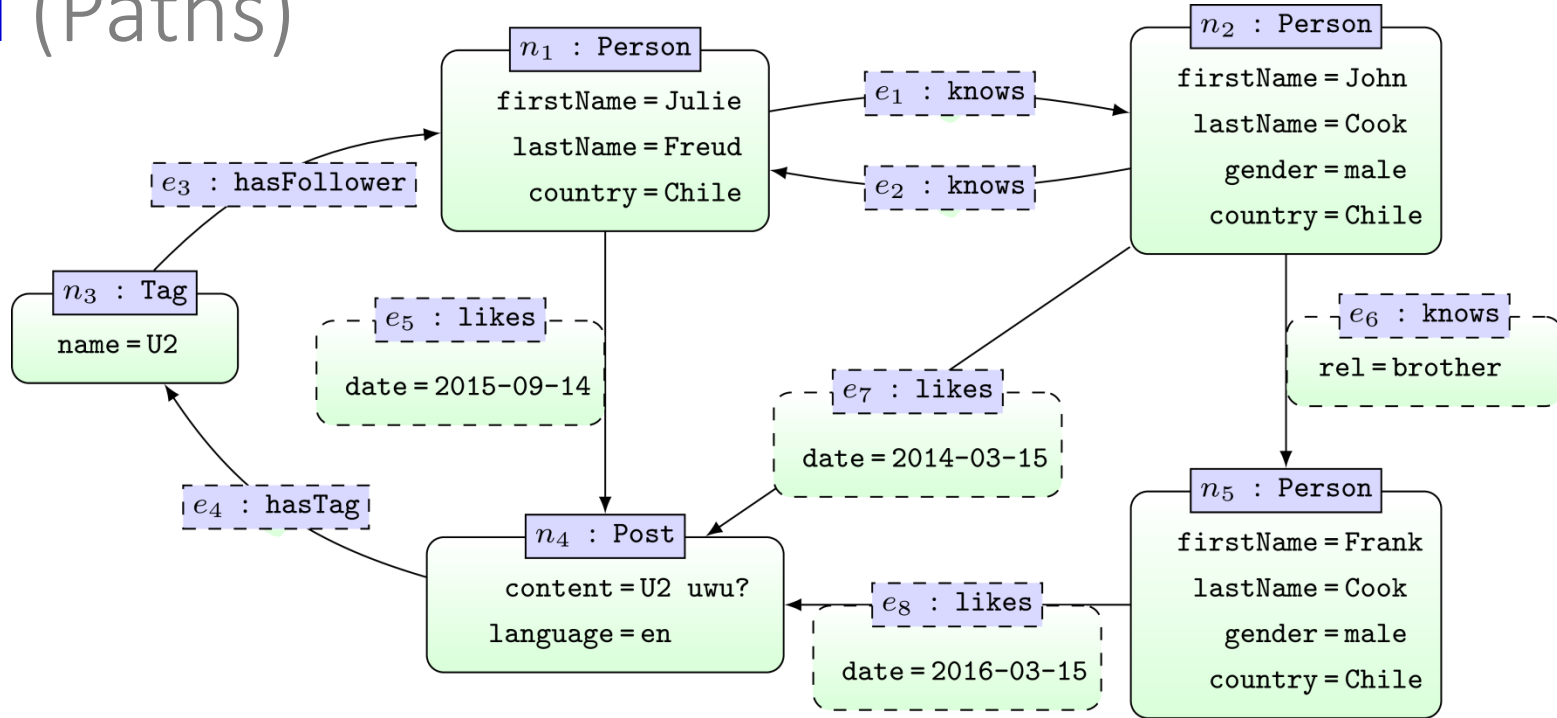
```

MATCH (f {firstName: 'Frank'}),
      (j {firstName: 'Julie'}),
      p = allShortestPaths((f)-[*]-(j))
RETURN p
  
```

p
$n_5 \cdot e_6 \cdot n_2 \cdot e_2 \cdot n_1$
$n_5 \cdot e_6 \cdot n_2 \cdot e_1 \cdot n_1$
$n_5 \cdot e_8 \cdot n_4 \cdot e_5 \cdot n_1$

... returns all shortest paths (matching criteria)

MATCH (Paths)



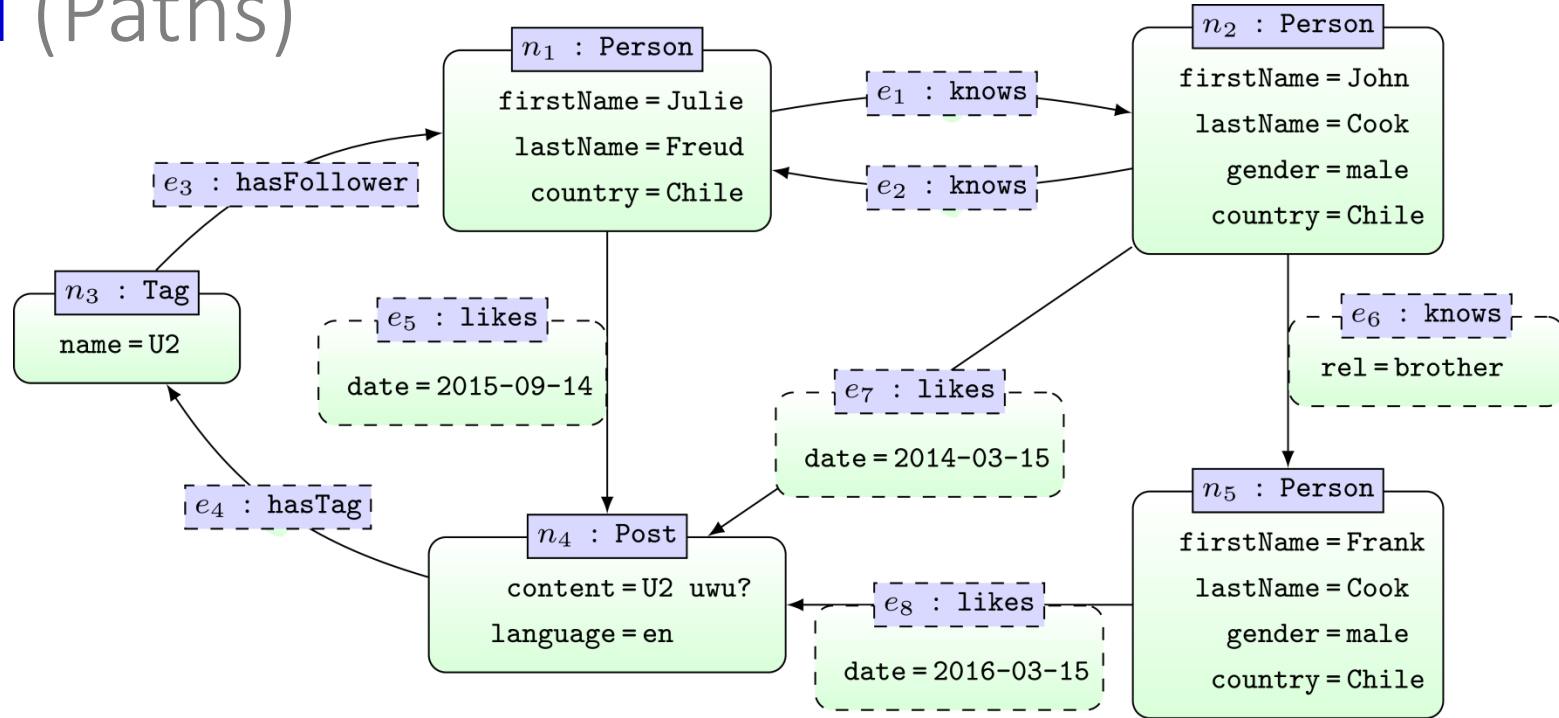
```

MATCH (f {firstName: 'Frank'}),
      (j {firstName: 'Julie'}),
      p = shortestPath((f)-[*]->(j))
RETURN p
  
```

p

$n_5 \cdot e_8 \cdot n_4 \cdot e_4 \cdot n_3 \cdot e_3 \cdot n_1$

MATCH (Paths)



```
MATCH (c1 {lastName: 'Cook'}),  
      (c2 {lastName: 'Freud'}),  
      p = shortestPath((c1)-[*]->(c2))  
RETURN p
```

p

n₅ · e₈ · n₄ · e₄ · n₃ · e₃ · n₁
n₂ · e₂ · n₁

... returns a shortest path for each matching pair of nodes

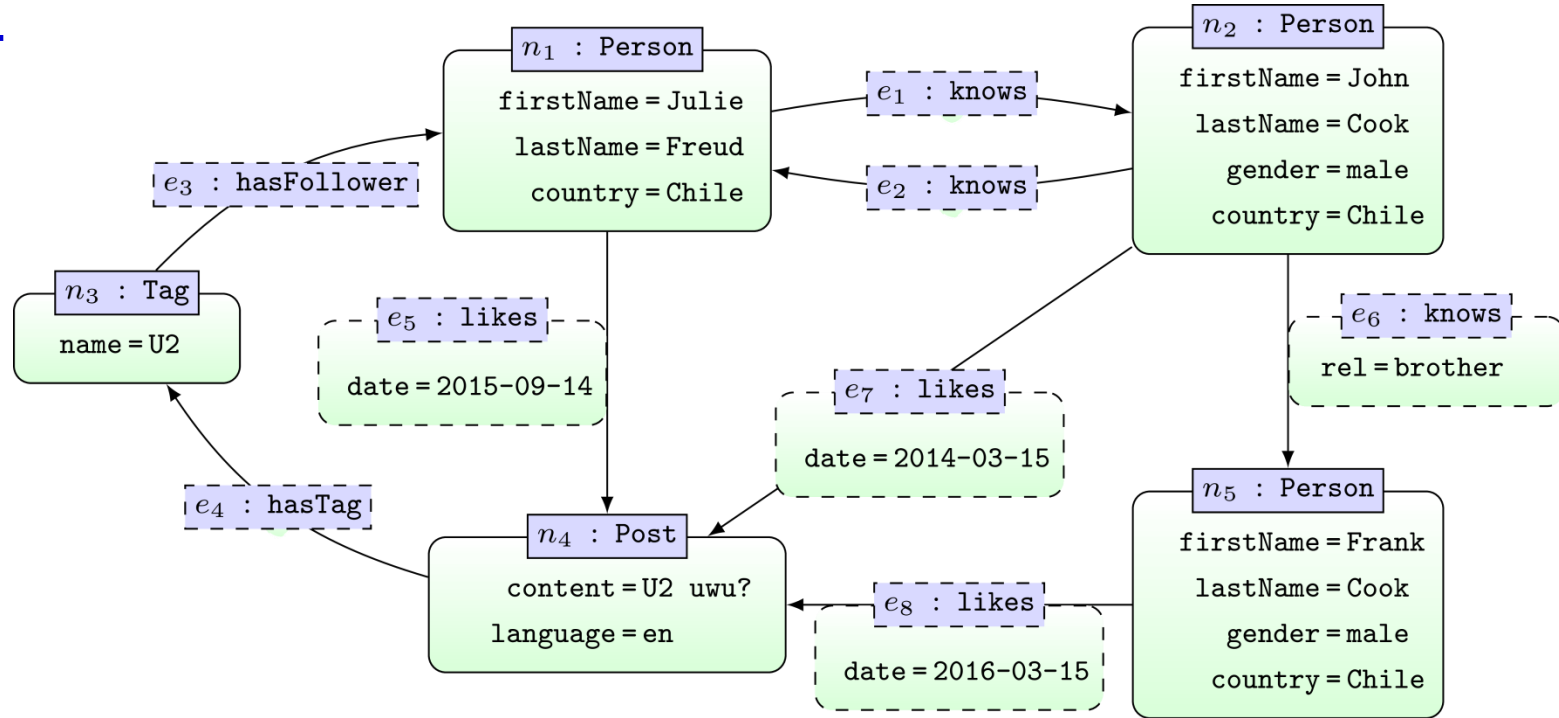
CYPHER:

WHERE

WHERE

- Boolean:
 - AND, OR, XOR, NOT
- (In)equalities:
 - <, >, <>, <=, >=
- Exists attribute property:
 - EXISTS
- Boolean:
 - STARTS WITH, ENDS WITH, CONTAINS, =~ (Regex)

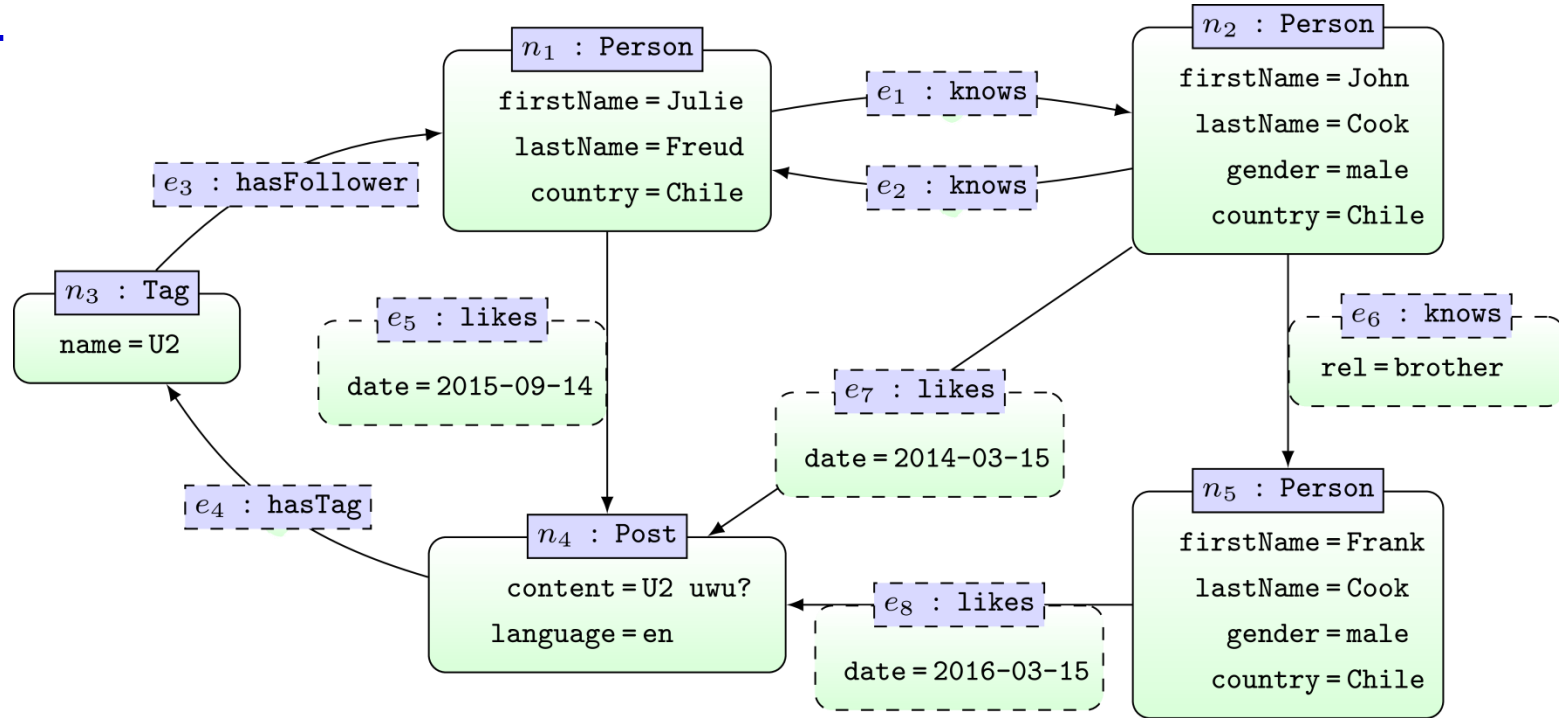
WHERE



```
MATCH (x)-[r:likes]->(y:Post)
WHERE r.date > '2010-01-01' AND r.date < '2015-01-01'
RETURN x.firstName
```

x.firstName
John

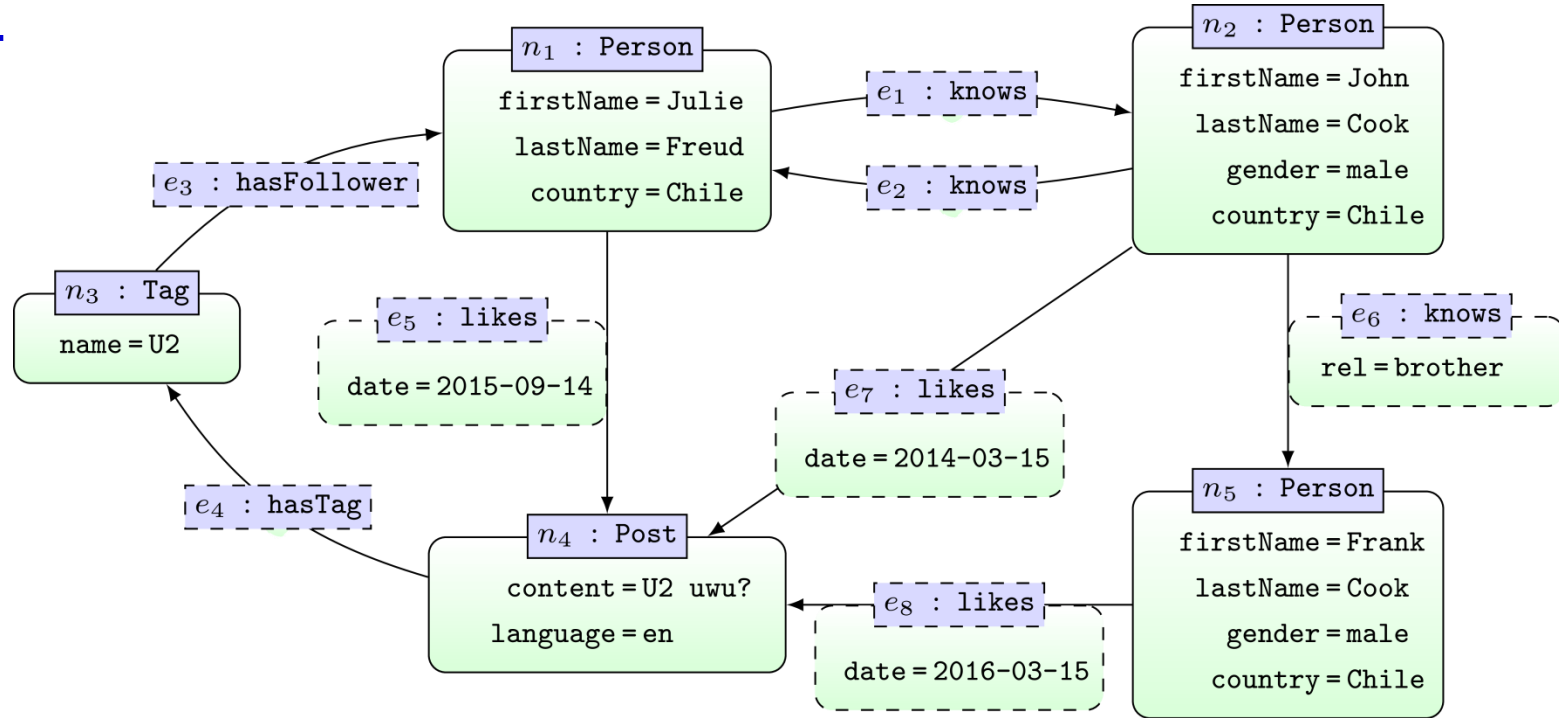
WHERE



```
MATCH (x)
WHERE EXISTS(x.gender)
RETURN x.firstName
```

x.firstName
John
Frank

WHERE

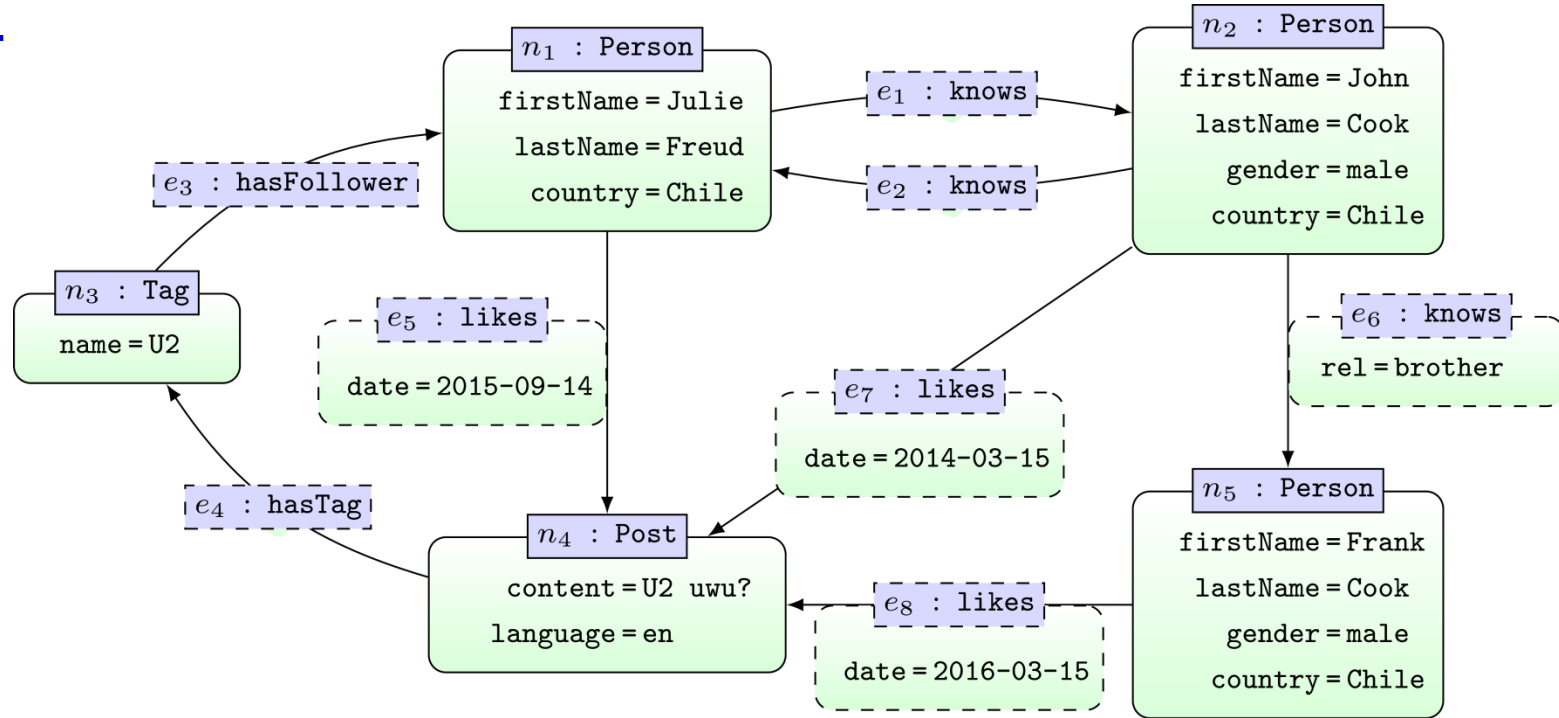


```
MATCH (x)
WHERE x.firstName STARTS WITH 'J'
RETURN x.firstName
```

x.firstName

John
Julie

WHERE



```
MATCH (x)
WHERE x.name =~ '[0-9]*'
RETURN x.name
```

x.name

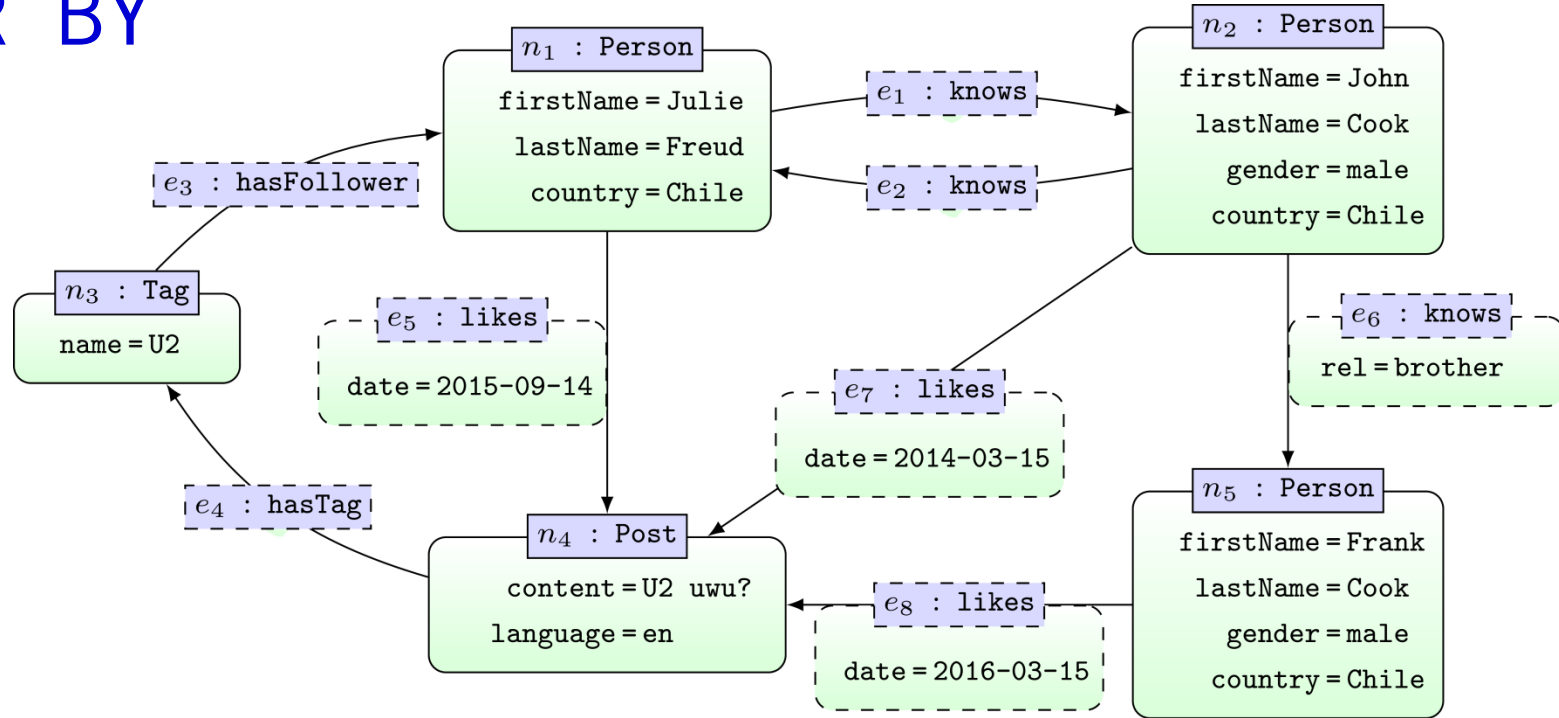
U2

CYPHER:

ORDER BY/SKIP/LIMIT

<https://neo4j.com/docs/developer-manual/3.4/cypher/clauses/order-by/>
<https://neo4j.com/docs/developer-manual/3.4/cypher/clauses/skip/>
<https://neo4j.com/docs/developer-manual/3.4/cypher/clauses/limit/>

ORDER BY

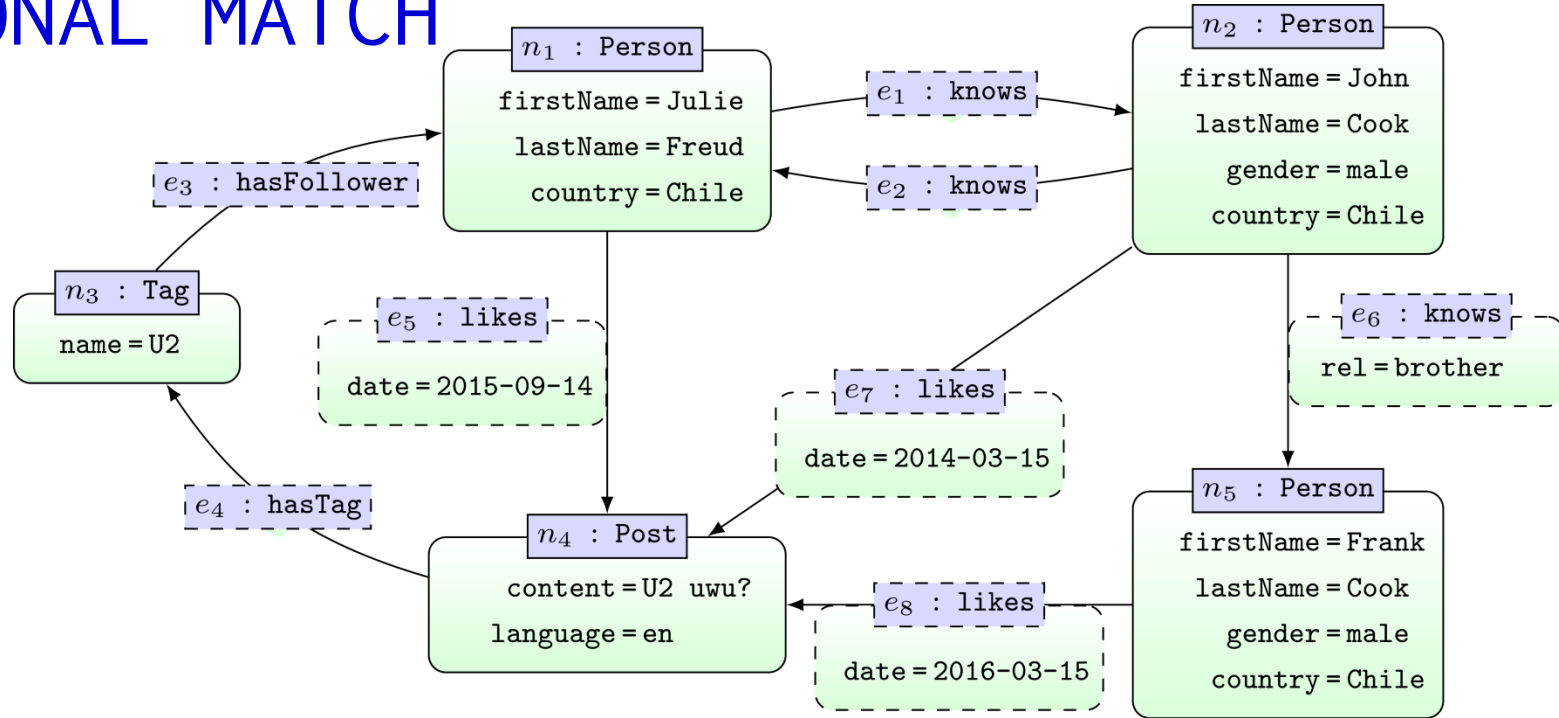


```
MATCH ()-[r:likes]->(p:Post)
RETURN r.date, p.content, p.language
ORDER BY p.content, r.date DESC
SKIP 1
LIMIT 1
```

r.date	p.content	p.language
2015-09-14	U2 uwu?	en

CYPHER: OPTIONAL MATCH

OPTIONAL MATCH



```

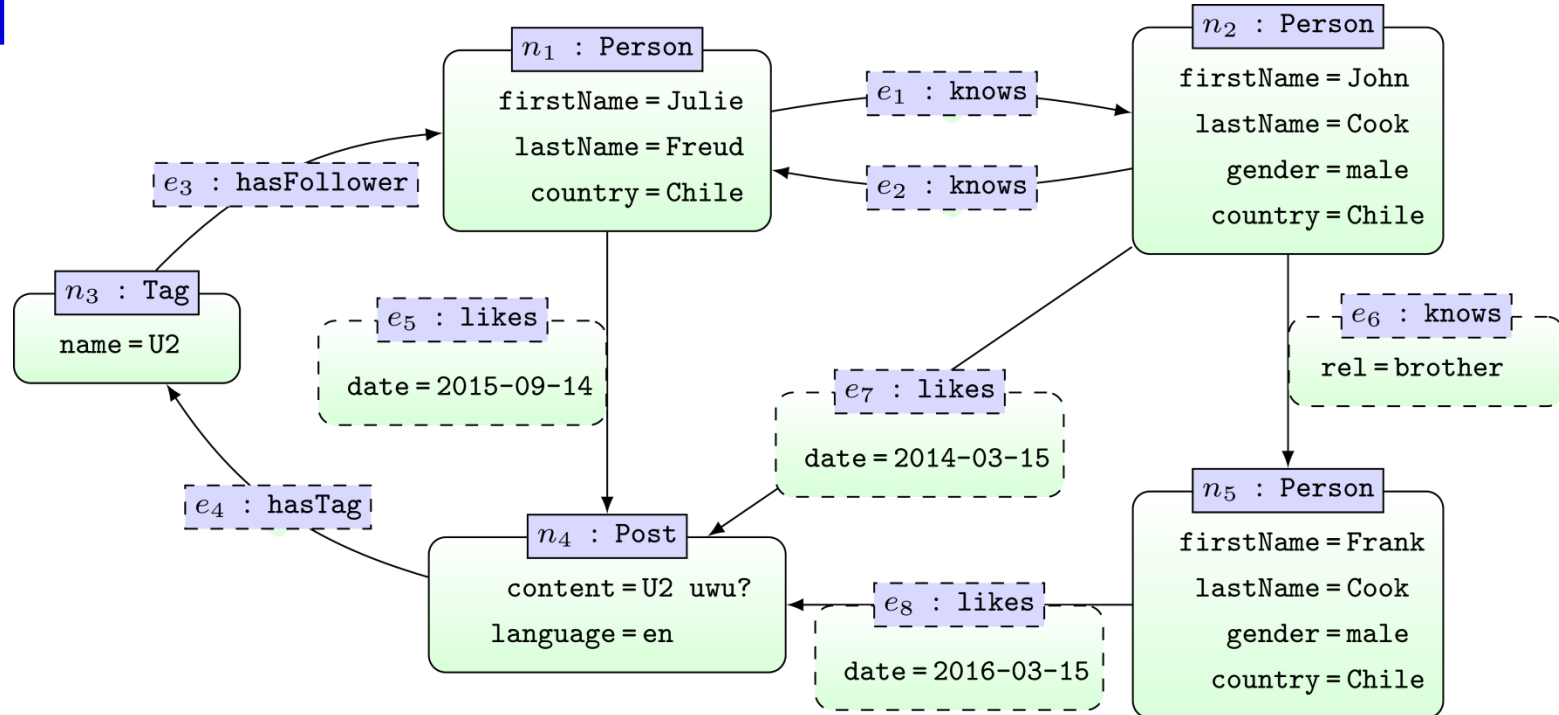
MATCH (x1)-[:knows]->(x2)
OPTIONAL MATCH (y)-[:hasFollower]->(x1)
RETURN x1.firstName,y.name
    
```

x1.firstName	y.name
Julie	U2
John	
John	

... OPTIONAL MATCH acts like a left join

CYPHER:
UNION (ALL)

UNION

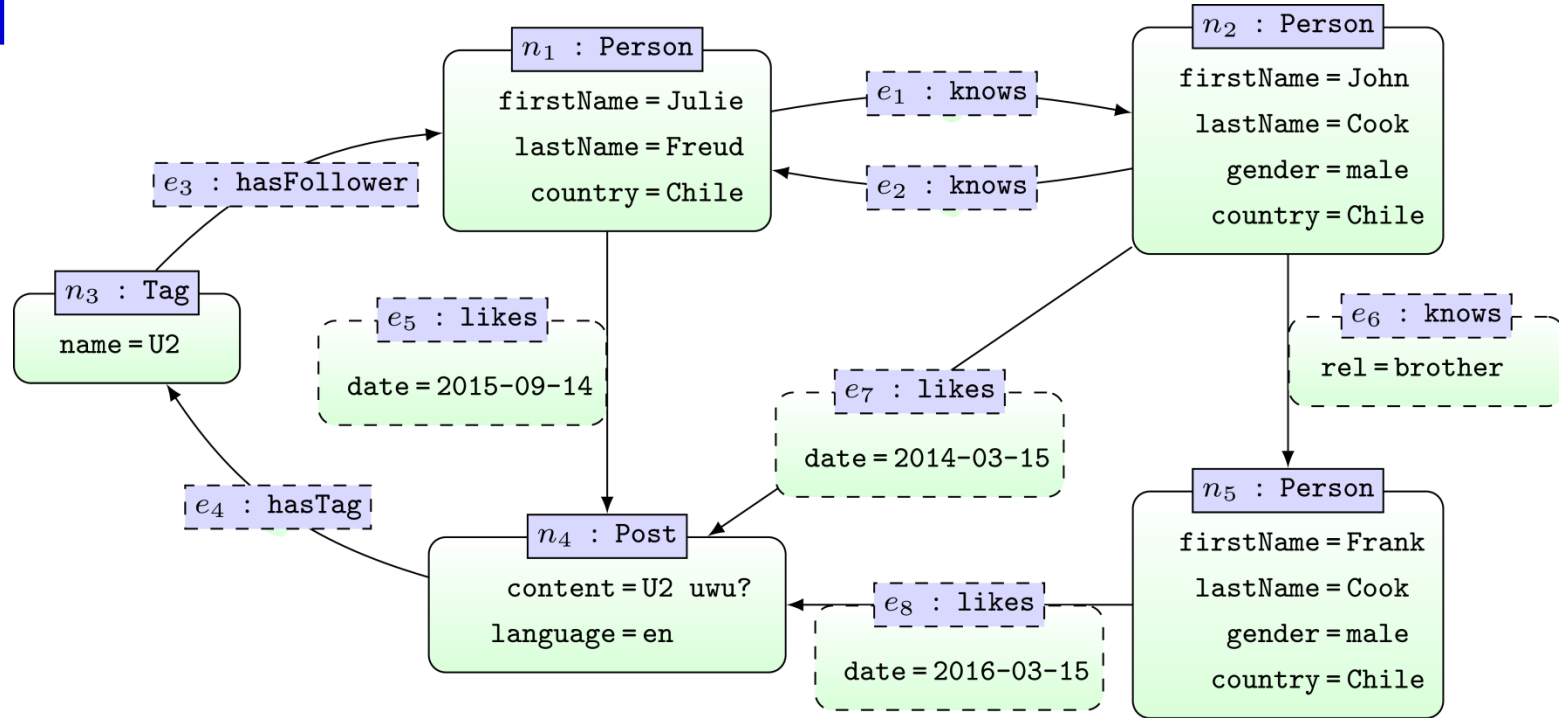


```
MATCH (x1)-[:knows]->(x2)
RETURN x1.firstName
UNION
MATCH (x1)-[:knows]->(x2)
RETURN x2.firstName
```



... column names have to be the same in the **UNION**

UNION



```
MATCH (x1)-[:knows]->(x2)
RETURN x1.firstName
UNION
MATCH (x2)-[:knows]->(x1)
RETURN x1.firstName
```

x1.firstName

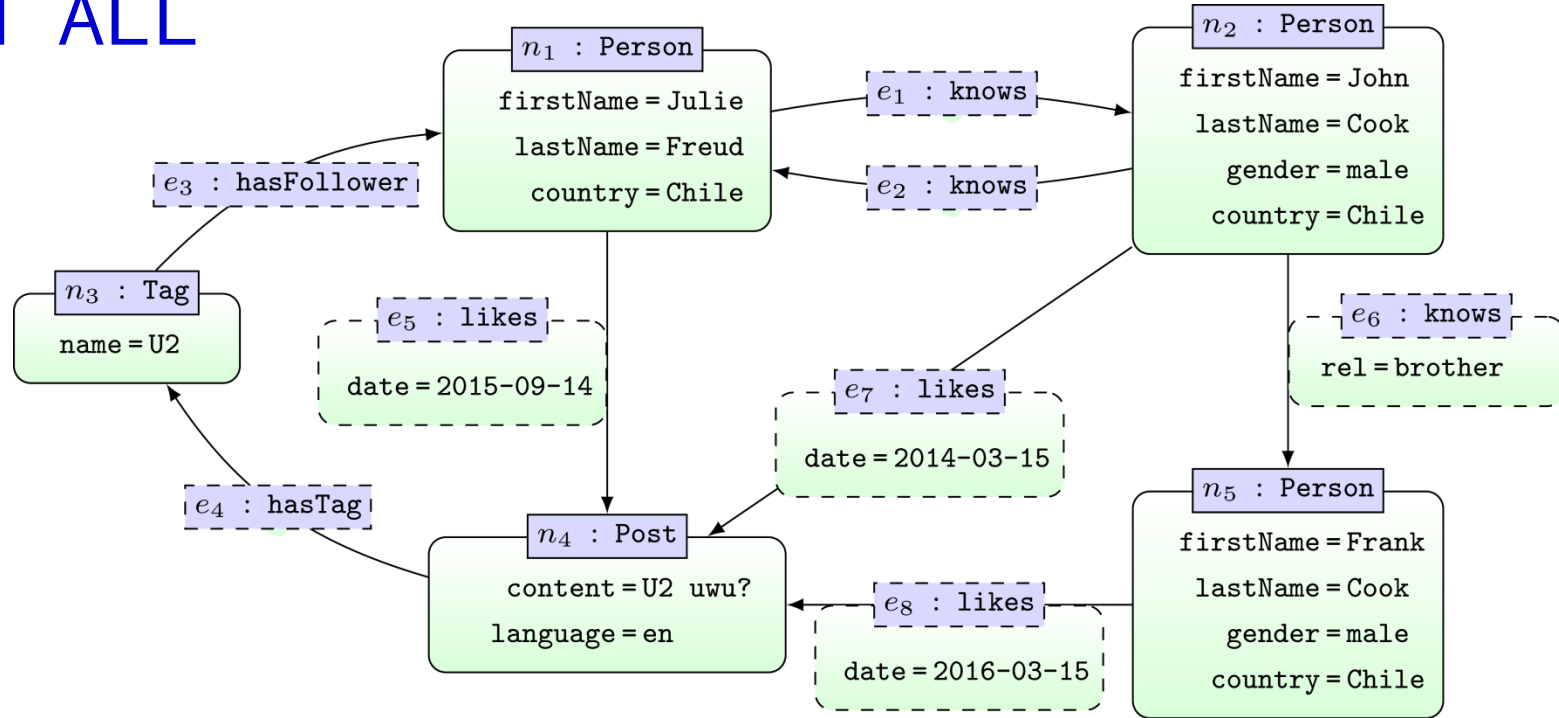
Julie

John

Frank

... UNION applies set union

UNION ALL



```

MATCH (x1)-[:knows]->(x2)
RETURN x1.firstName
UNION ALL
MATCH (x2)-[:knows]->(x1)
RETURN x1.firstName

```

<u>x1.firstName</u>
Julie
Julie
John
John
John
Frank

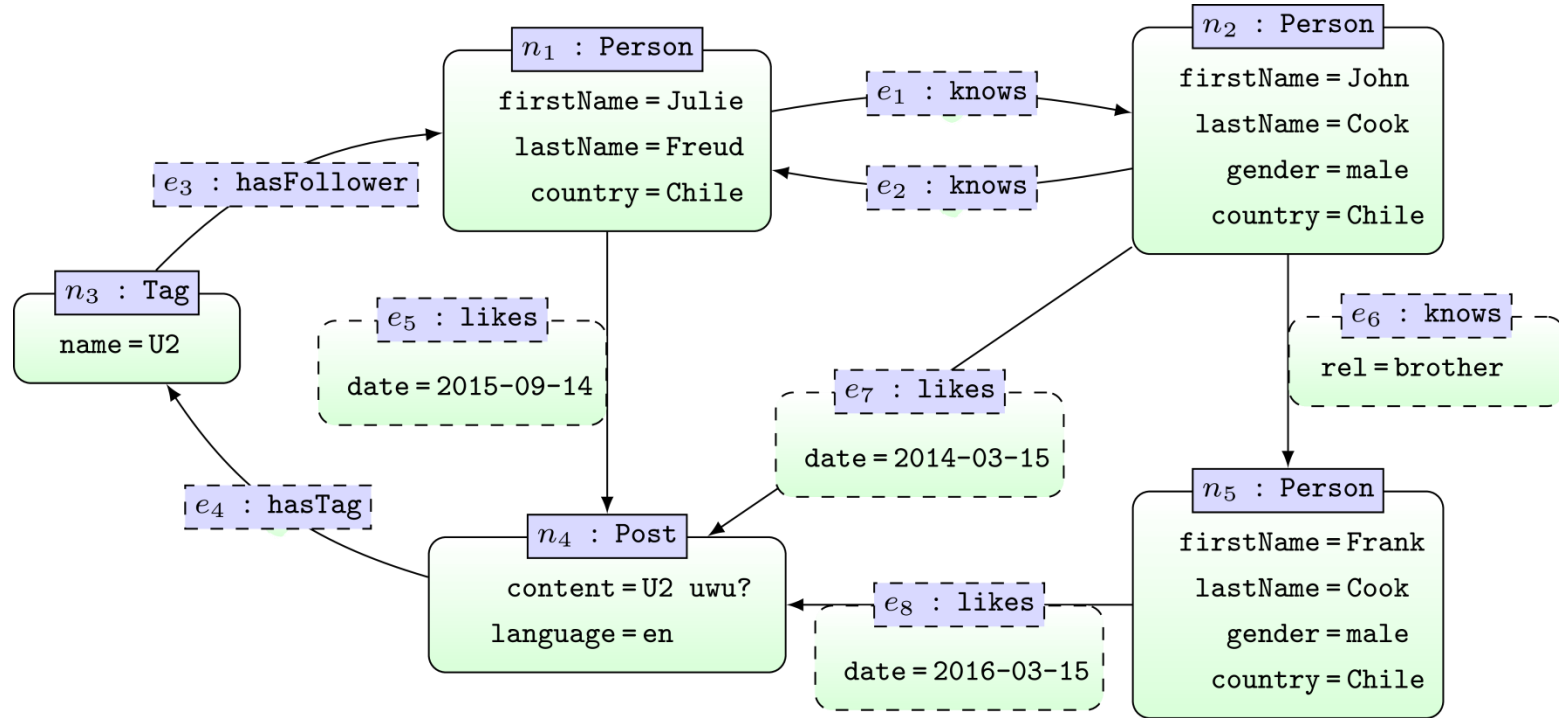
... UNION ALL applies bag union

CYPHER: AGGREGATION

Aggregation

- `count`
- `max/min`
- `avg`
- `percentileCont/percentileDisc`
 - Computes percentile of some value w.r.t. some list
 - (*continuous*: interpolates / *discrete*: rounds)
- `stDev/stDevP`
 - Computes standard deviation (*sample/population*)

COUNT

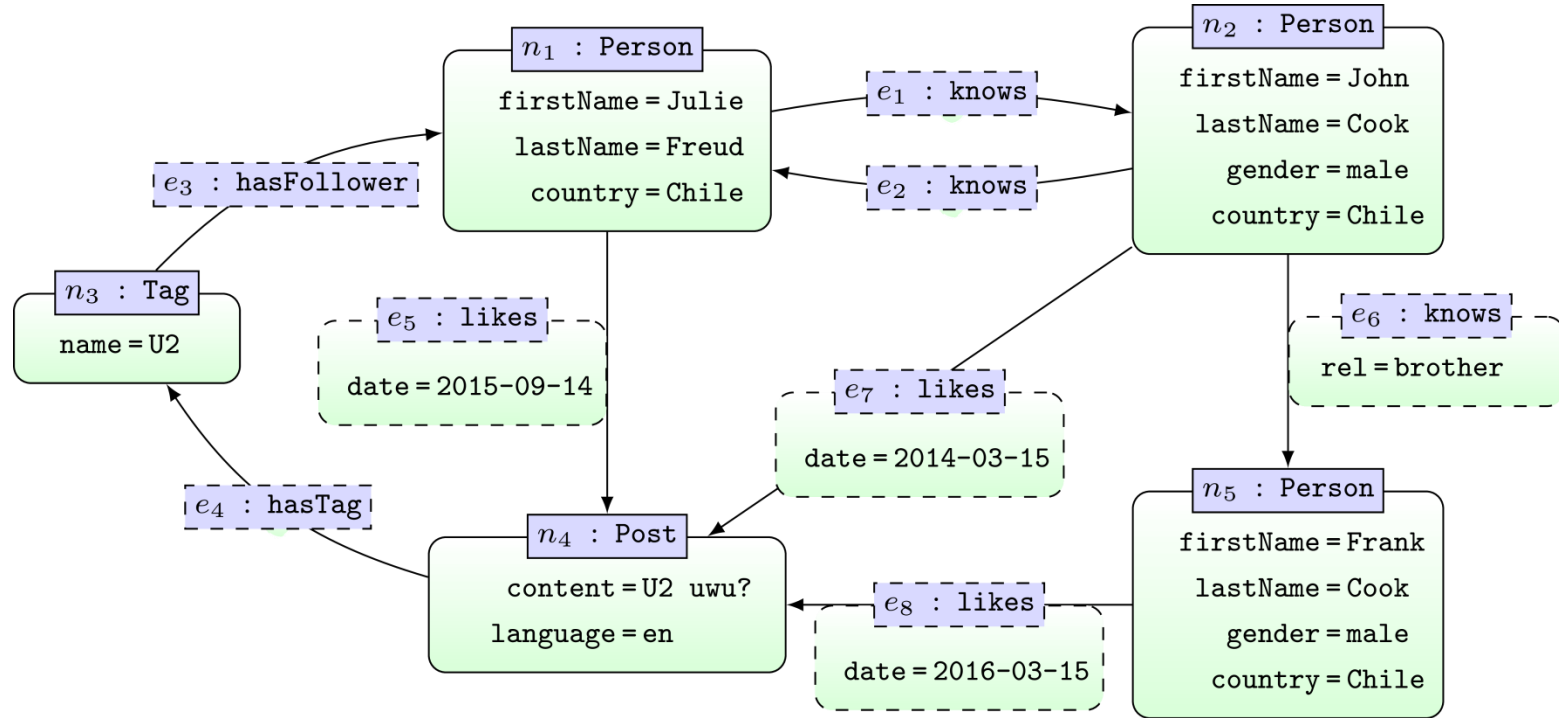


```
MATCH (x1)-[:knows*]->(x2)
RETURN x1.firstName, count(x2)
```

x1.firstName	count(x2)
Julie	3
John	4

... GROUP BY implicit on non-aggregated columns

COUNT

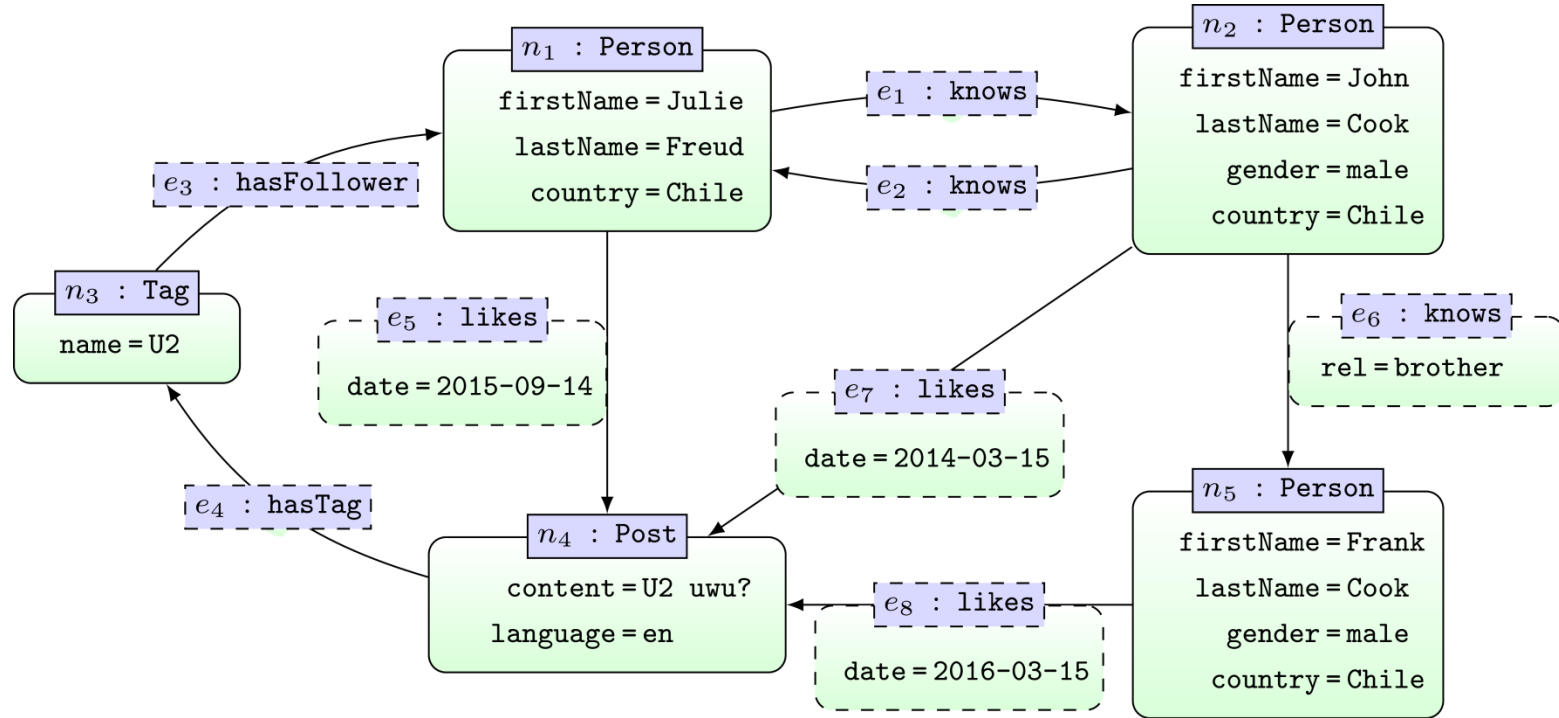


```
MATCH (x1)-[:knows*]->(x2)
RETURN DISTINCT x1.firstName, count(x2)
```

x1.firstName	count(x2)
Julie	3
John	4

... removes duplicate results, not count arguments

COUNT



```
MATCH (x1)-[:knows*]->(x2)
RETURN x1.firstName, count(distinct x2)
```

x1.firstName	count(x2)
Julie	3
John	3

... removes duplicate count arguments

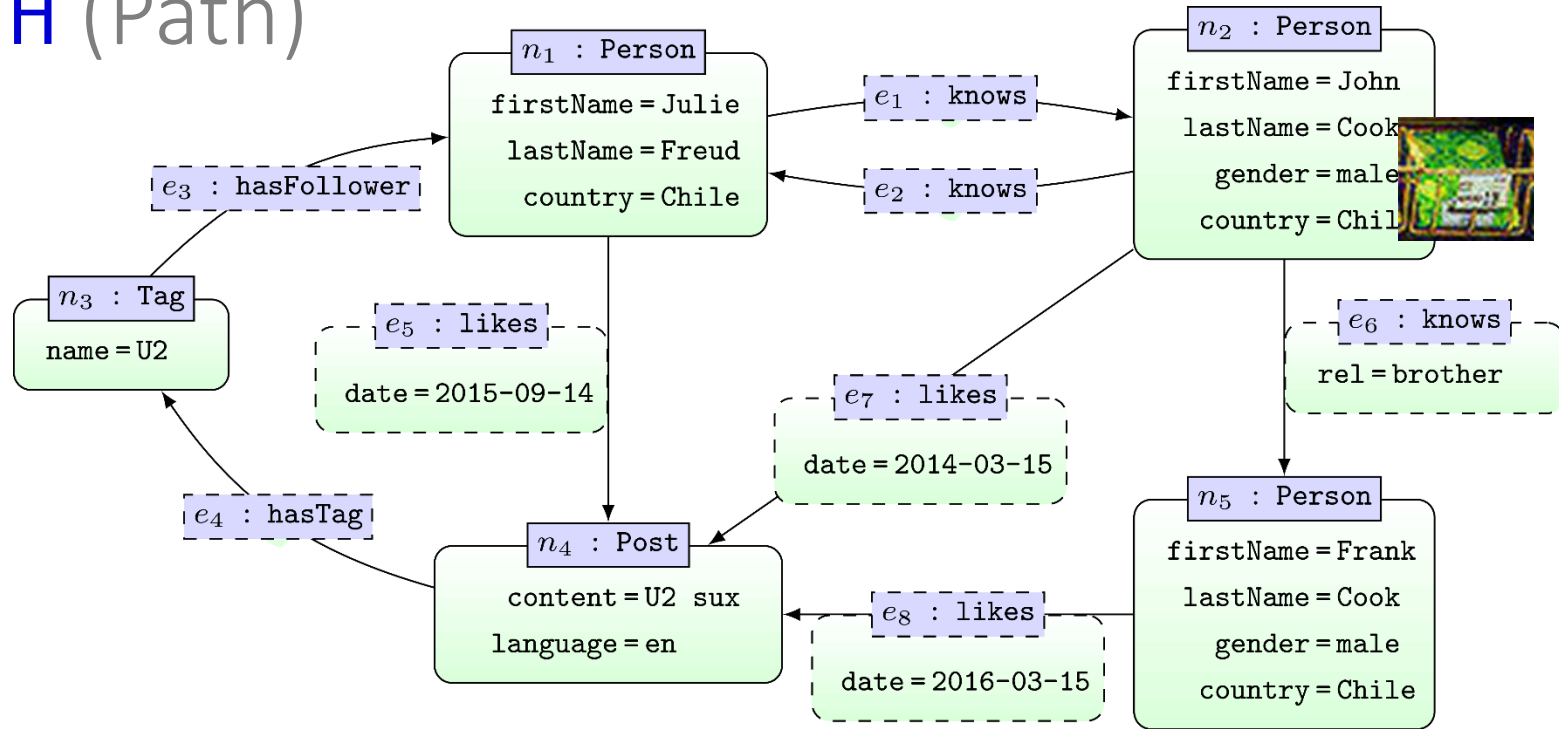
CYPHER: OTHER FUNCTIONS



<https://neo4j.com/docs/developer-manual/current/cypher/functions/>



LENGTH (Path)



```
MATCH (f {firstName: 'Frank'}),  
      (j {firstName: 'Julie'}),  
      p = shortestPath((f)-[*]->(j))  
RETURN length(p)
```

length(p)
3

CYPHER: UPDATE GRAPHS
CREATE/REMOVE/...

Update graphs

- **CREATE** nodes and relationships
 - <https://neo4j.com/docs/developer-manual/current/cypher/clauses/create/>
- **DELETE** nodes and relationships
 - <https://neo4j.com/docs/developer-manual/current/cypher/clauses/delete/>
- **DETACH DELETE** nodes with relationships
 - <https://neo4j.com/docs/developer-manual/current/cypher/clauses/delete/>
- **SET** update labels and attributes
 - <https://neo4j.com/docs/developer-manual/current/cypher/clauses/set/>
- **REMOVE** remove labels and attributes
 - <https://neo4j.com/docs/developer-manual/current/cypher/clauses/remove/>

Update graphs

- Create the nodes we've seen

```
CREATE (:Person { firstName:'Julie', lastName:'Freud', country:'Chile' });
CREATE (:Person { firstName:'John', lastName:'Cook', country:'Chile', gender:'male' });
CREATE (:Tag { name:'U2' });
CREATE (:Post { content:'U2 sux', language:'en' });
CREATE (:Person { firstName:'Frank', lastName:'Cook', country:'Chile', gender:'male' });
```

- Create the edges (sample) we've seen

```
MATCH (n1 { firstName:'Julie' }),(n2 { firstName:'John' }),(n3:Tag),(n4:Post),(n5 { firstName:'Frank' })
CREATE (n1)-[e1:knows]->(n2)
CREATE (n2)-[e2:knows]->(n1)
CREATE (n3)-[e3:hasFollower]->(n1)
CREATE (n4)-[e4:hasTag]->(n3)
CREATE (n1)-[e5:likes { date:'2015-09-14'}]->(n4)
CREATE (n2)-[e6:knows { rel:'brother'}]->(n5)
CREATE (n2)-[e7:likes { date:'2014-03-15'}]->(n4)
CREATE (n5)-[e8:likes { date:'2016-03-15'}]->(n4);...
```

- Drop all nodes and edges

```
MATCH (n) DETACH DELETE n;
```

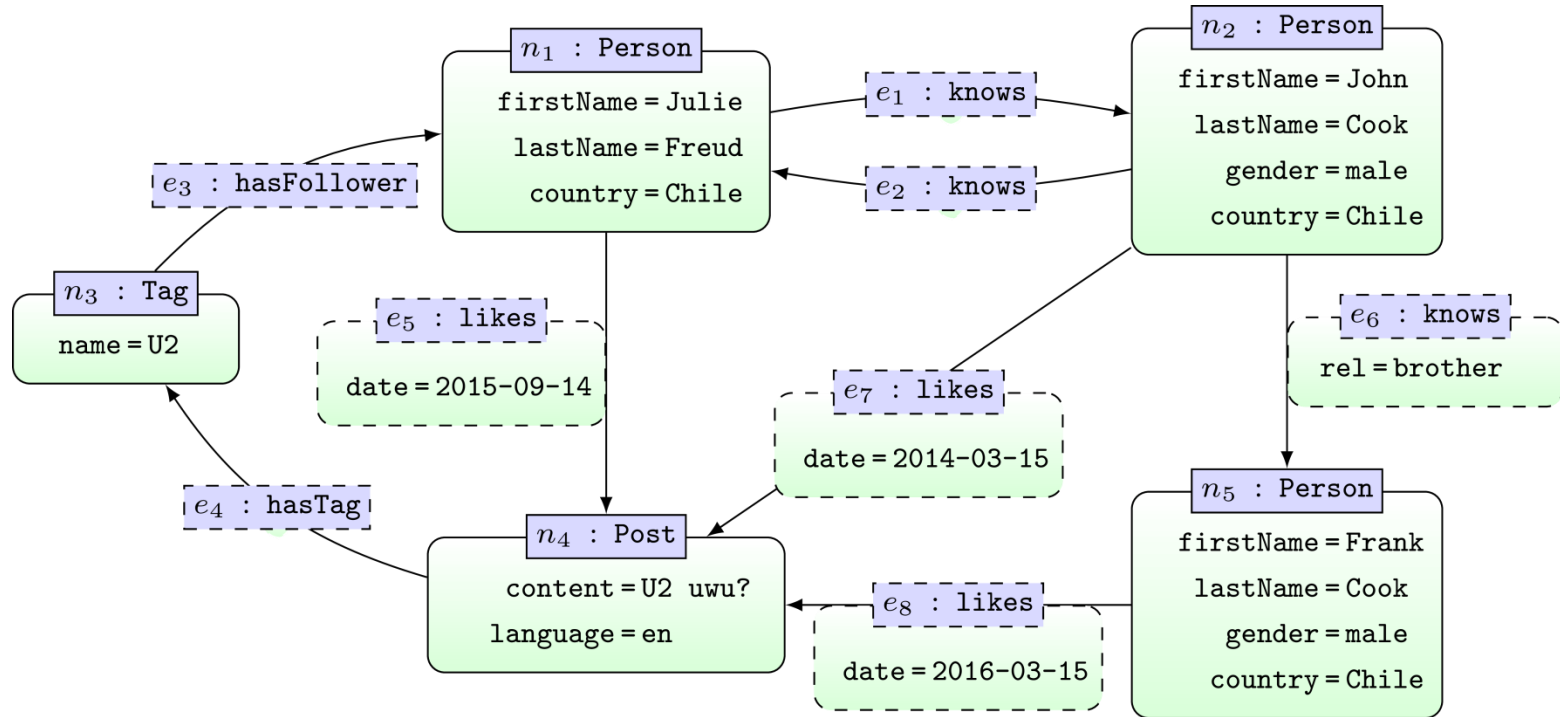
/CORE OF CYPHER
/PART OF NEO4J

Neo4j Graph Database

- Data Model: [Property Graphs](#)
- Query Language: [Cypher](#)
- Scripting Language: [Gremlin](#)
- Licence: [Open Source \(Single Machine\)](#)
[Commercial \(Cluster Edition\)](#)



Property Graph: Cypher



```
MATCH (x1:Person {firstName:"Julie"})-[:knows*]->(x2:Person)
MATCH (x2)-[:likes]->()-[:hasTag]->()-[:hasFollower]->(x1)
RETURN x2.firstName
```

x2.firstName

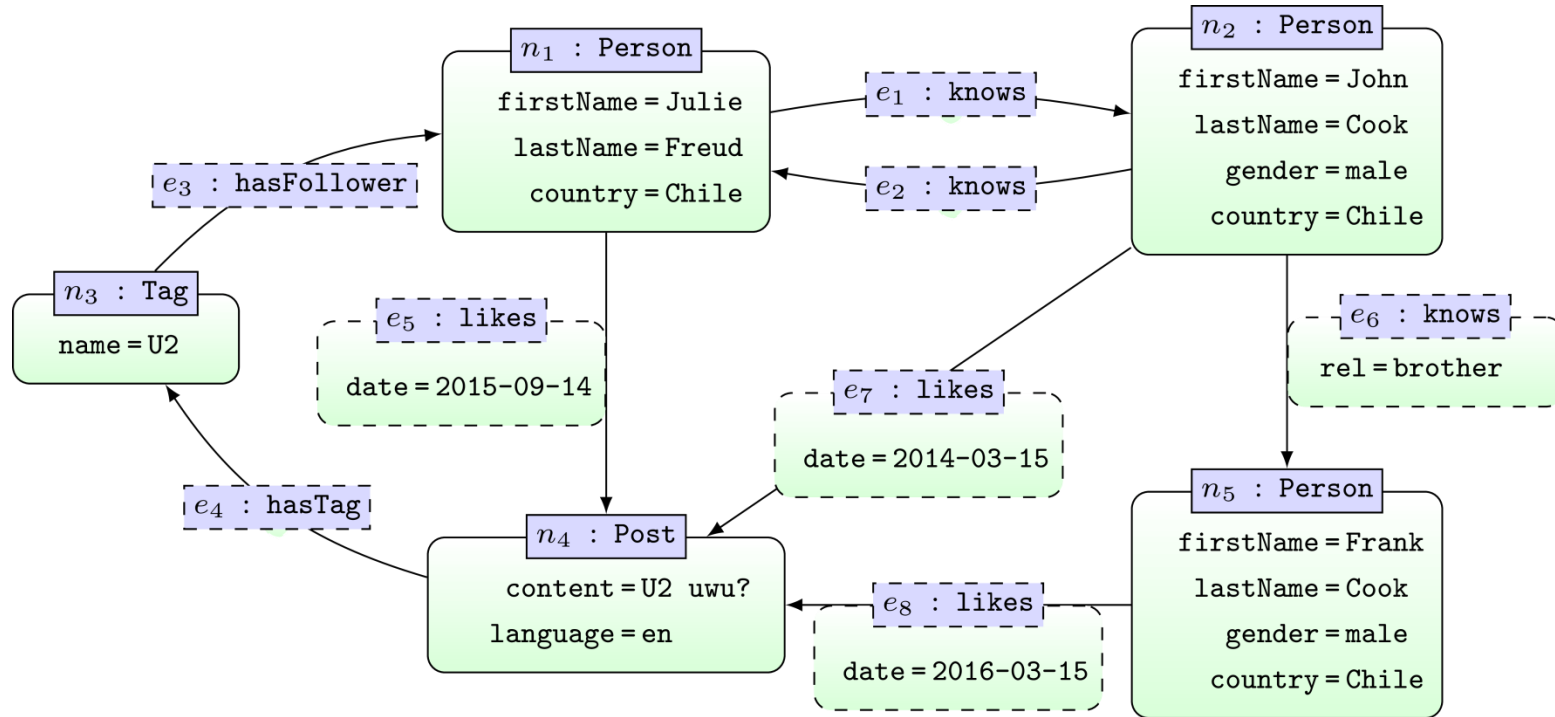
Julie

John

Frank



Property Graph: Gremlin



```
n1 = g.v.filter{firstName:"Julie"}.inV.outE('knows').loop()
n2 = n1.call().more.functions().until('done')
```

```
x2.firstName
```

```
Julie
```

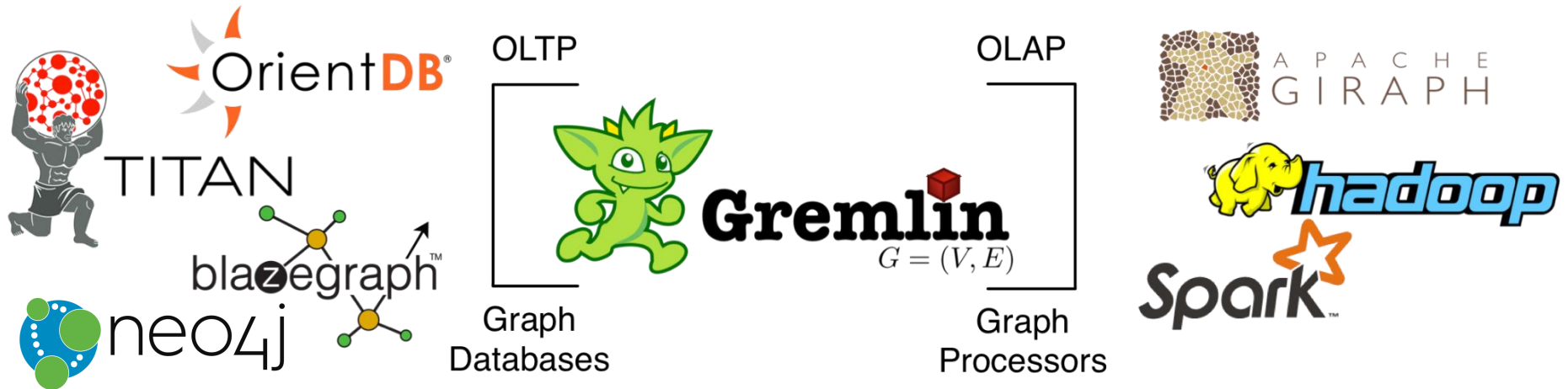
```
John
```

```
Frank
```



Gremlin
 $G = (V, E)$

Gremlin: Graph Queries + Processing





Questions?