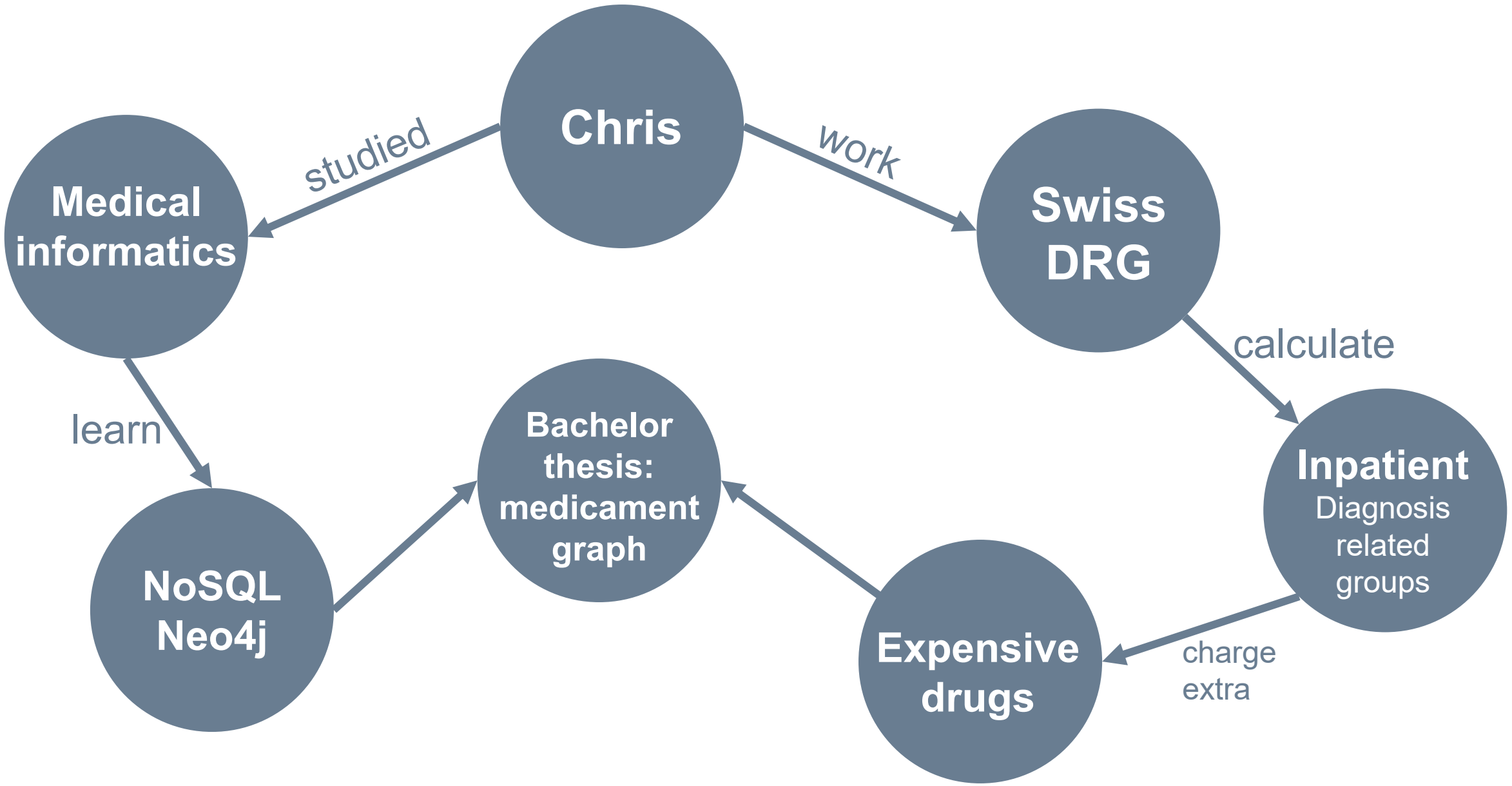
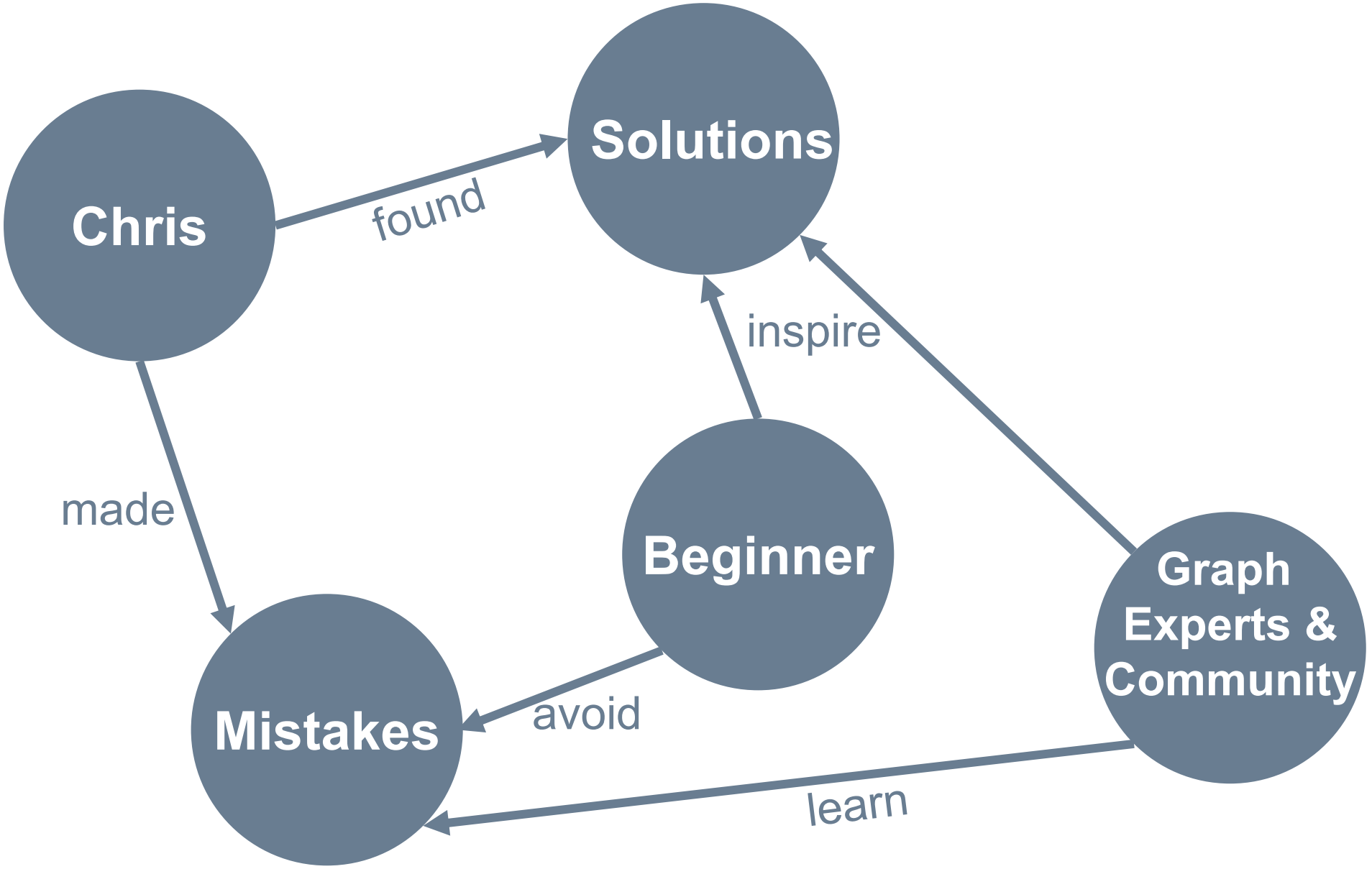
Berner Fachhochschule
Haute école spécialisée bernoise
Bern University of Applied Sciences

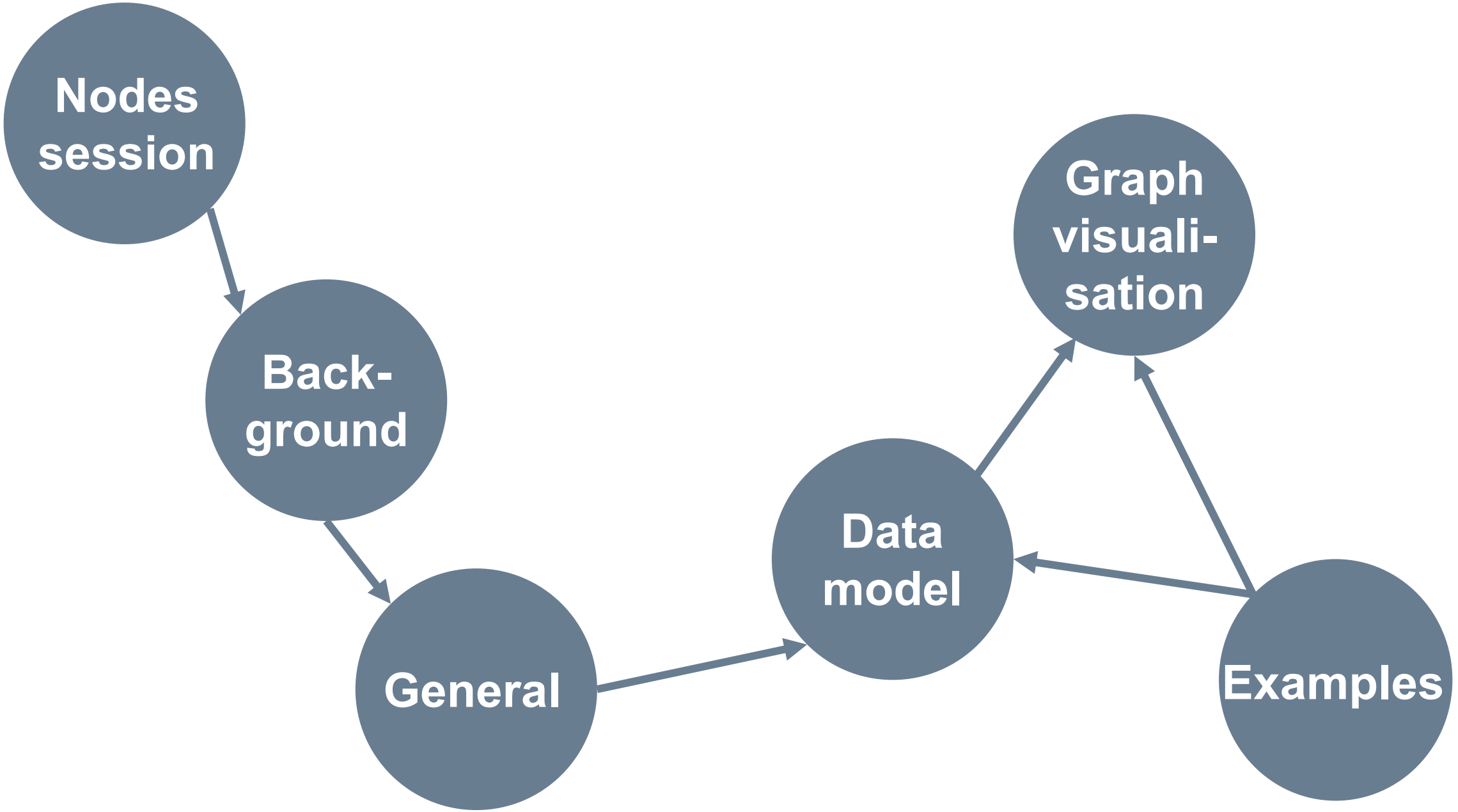
Knowledge Graph of Drugs Data for Swiss Healthcare System

Tipps and tricks from a Neo4j beginner for your (first) knowledge graph

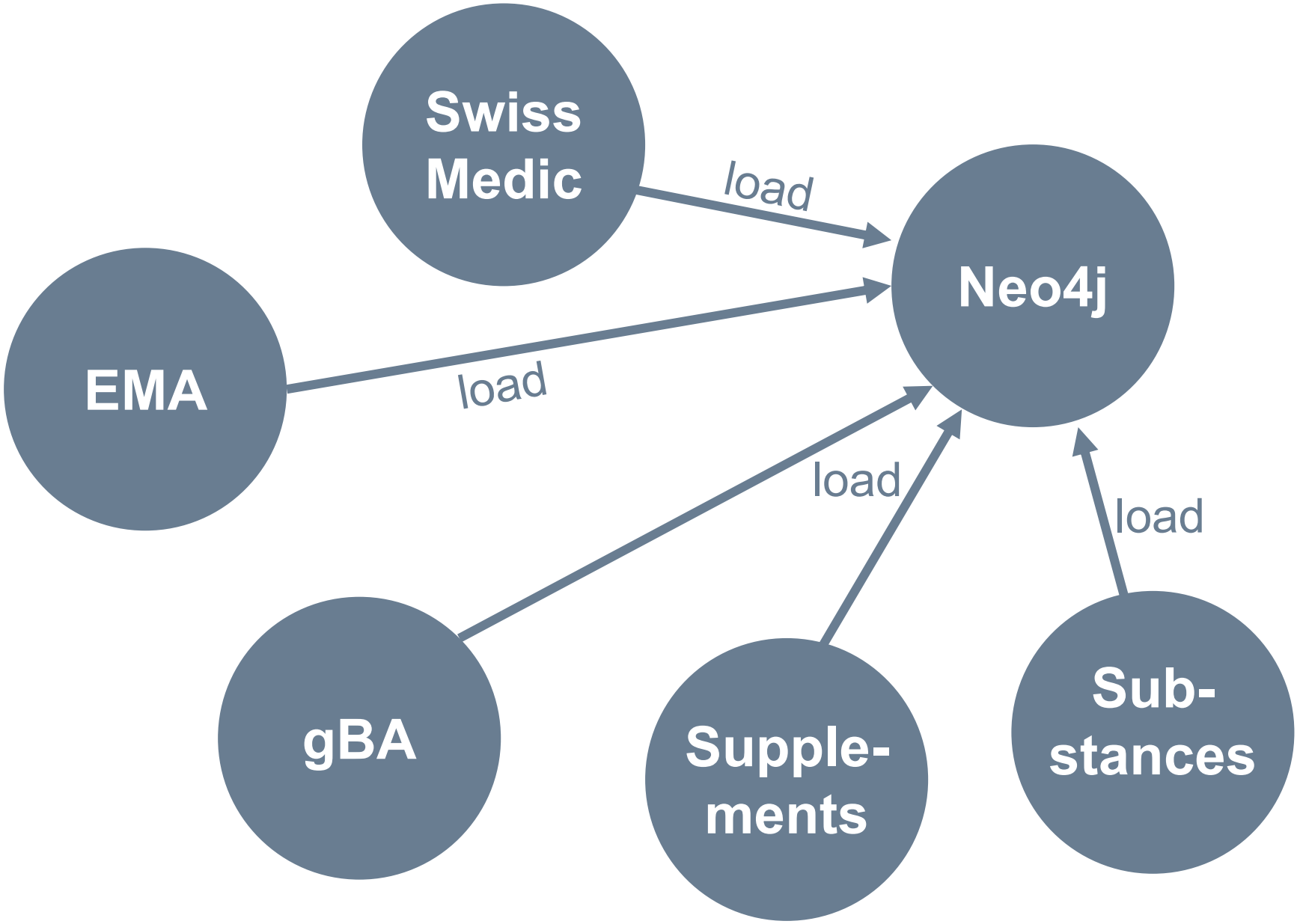
Christian Franke, 6th November 2025, Nodes 2025


Chris
studied
Medical informatics
work
Swiss DRG
calculate
Inpatient
Diagnosis related groups
charge extra
Expensive drugs
Bachelor thesis: medicament graph
learn
NoSQL Neo4j

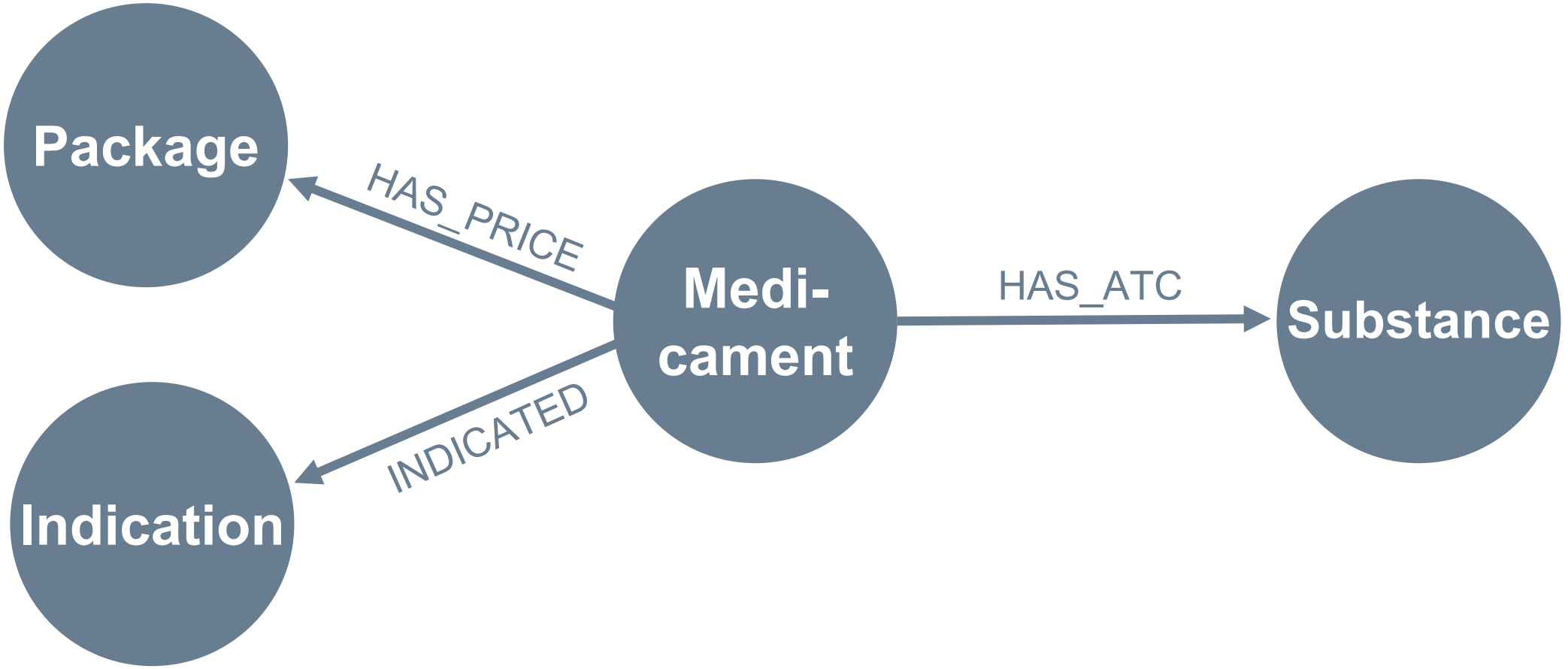

Solutions
Chris
found
inspire
made
Beginner
Graph Experts & Community
Mistakes
avoid
learn


Nodes session
Back-ground
General
Data model
Graph visuali-sation
Examples

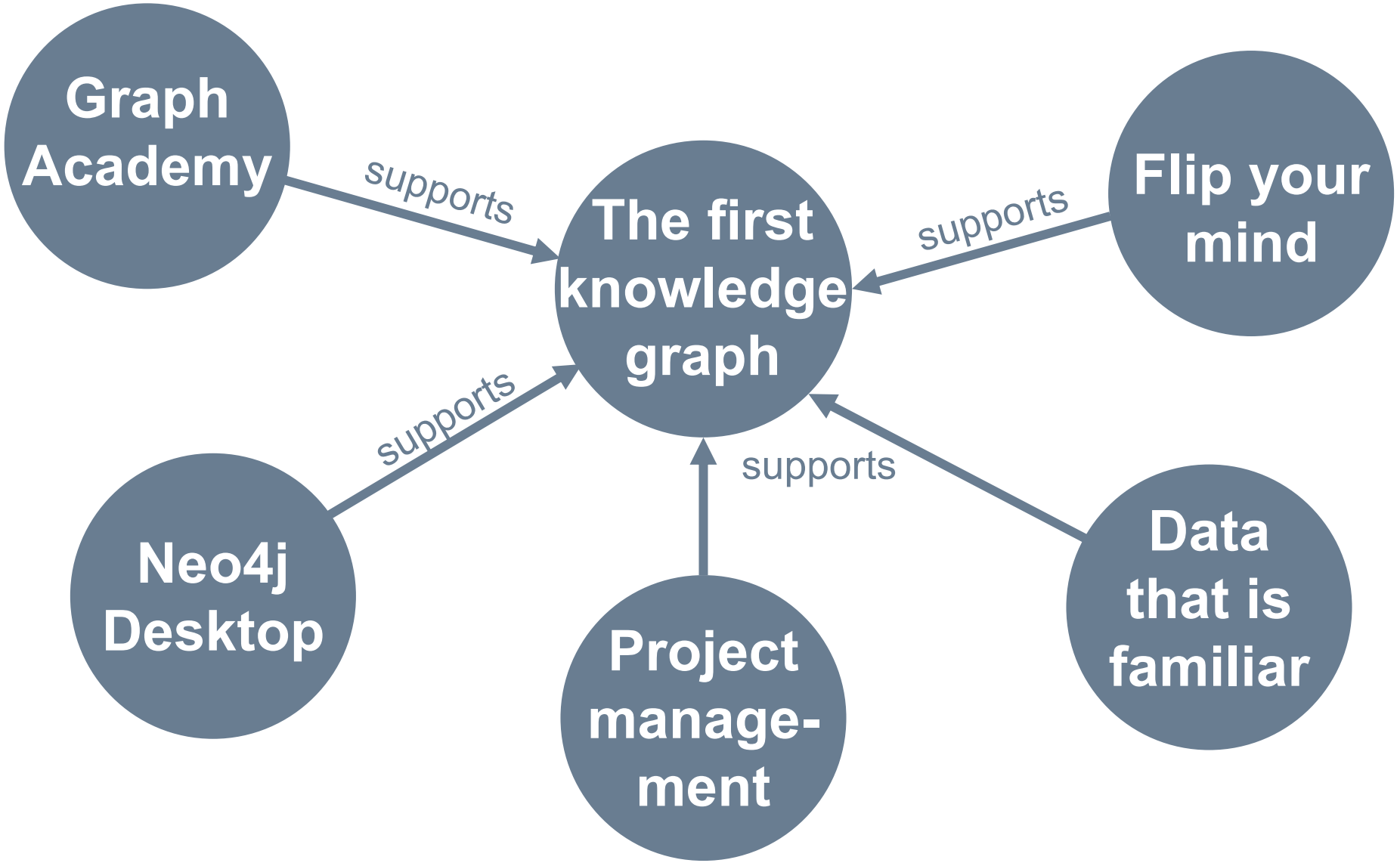
Back-
ground


Swiss Medic
load
Neo4j
EMA
load
load
load
gBA
Supple-ments
Sub-stances

Simplified data model

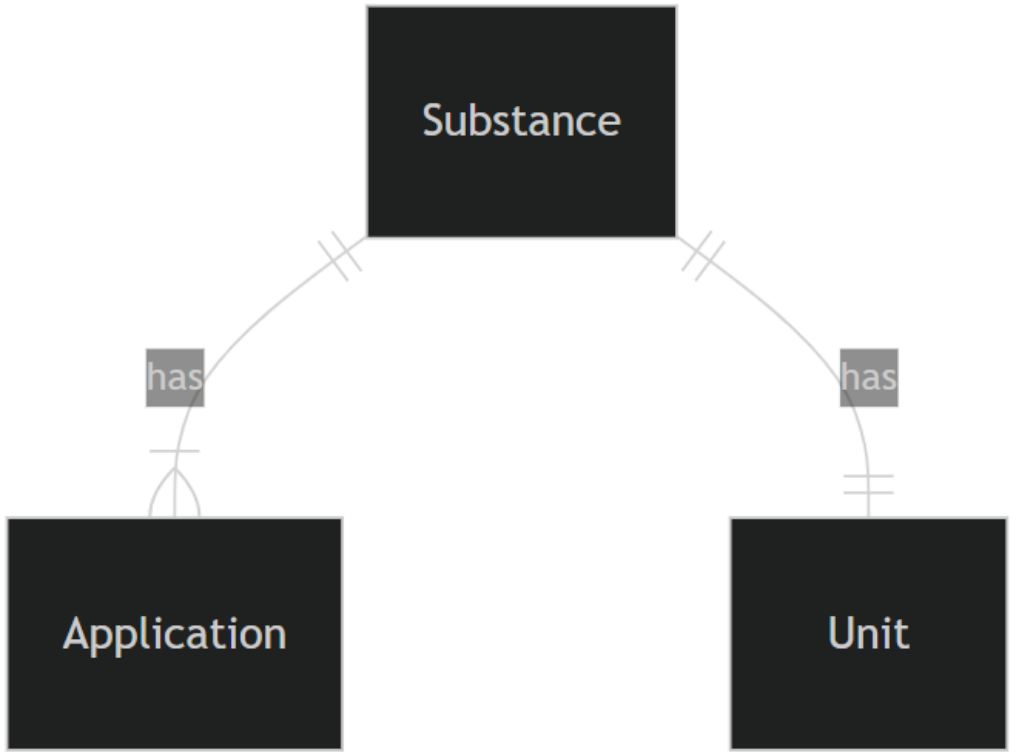


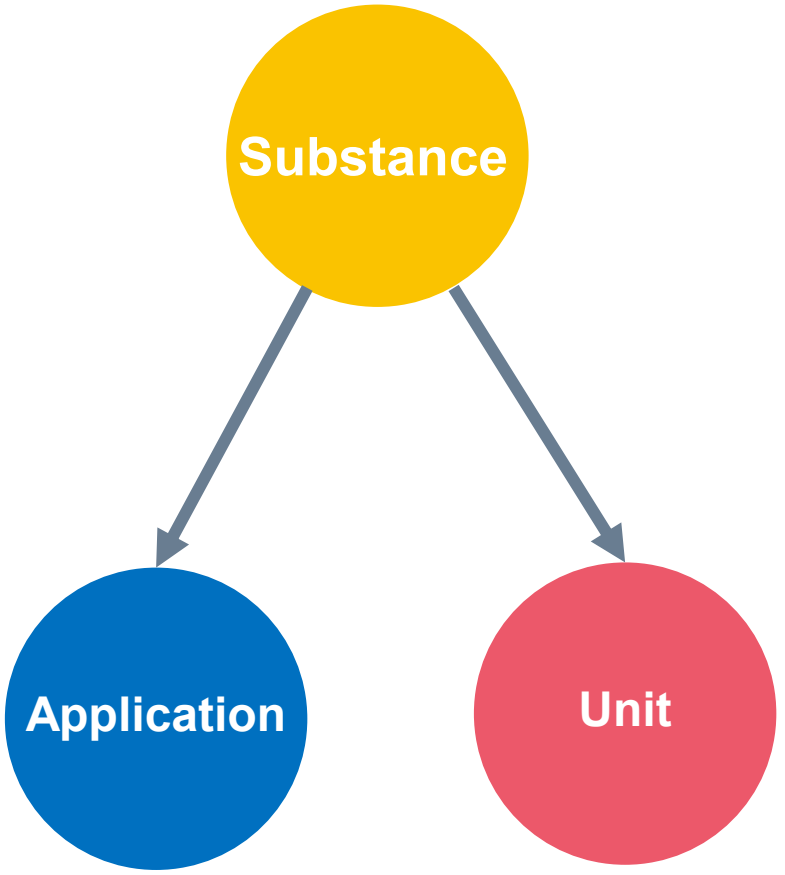
General


Graph Academy
supports
The first knowledge graph
supports
Flip your mind
supports
Neo4j Desktop
supports
Project manage-ment
Data that is familiar

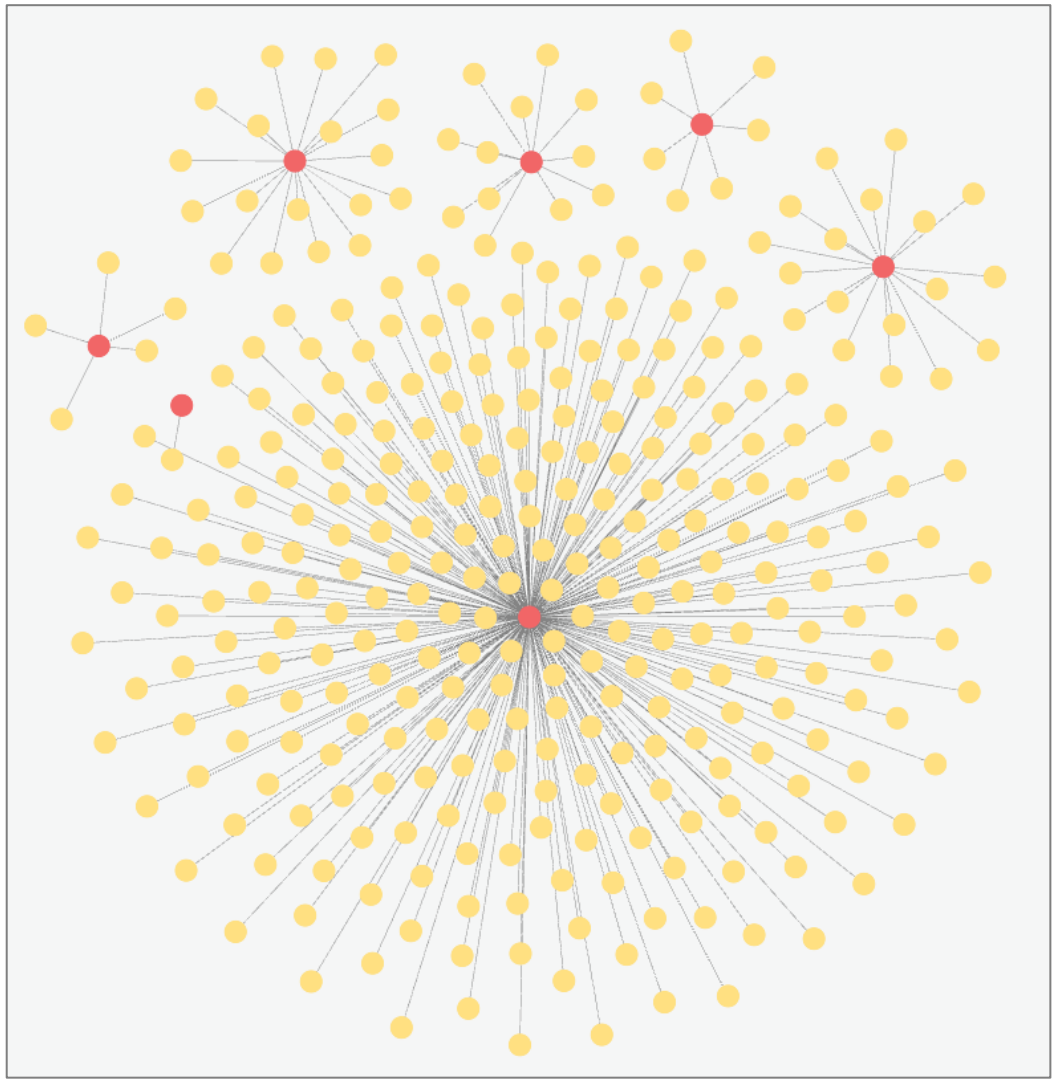
Example 1

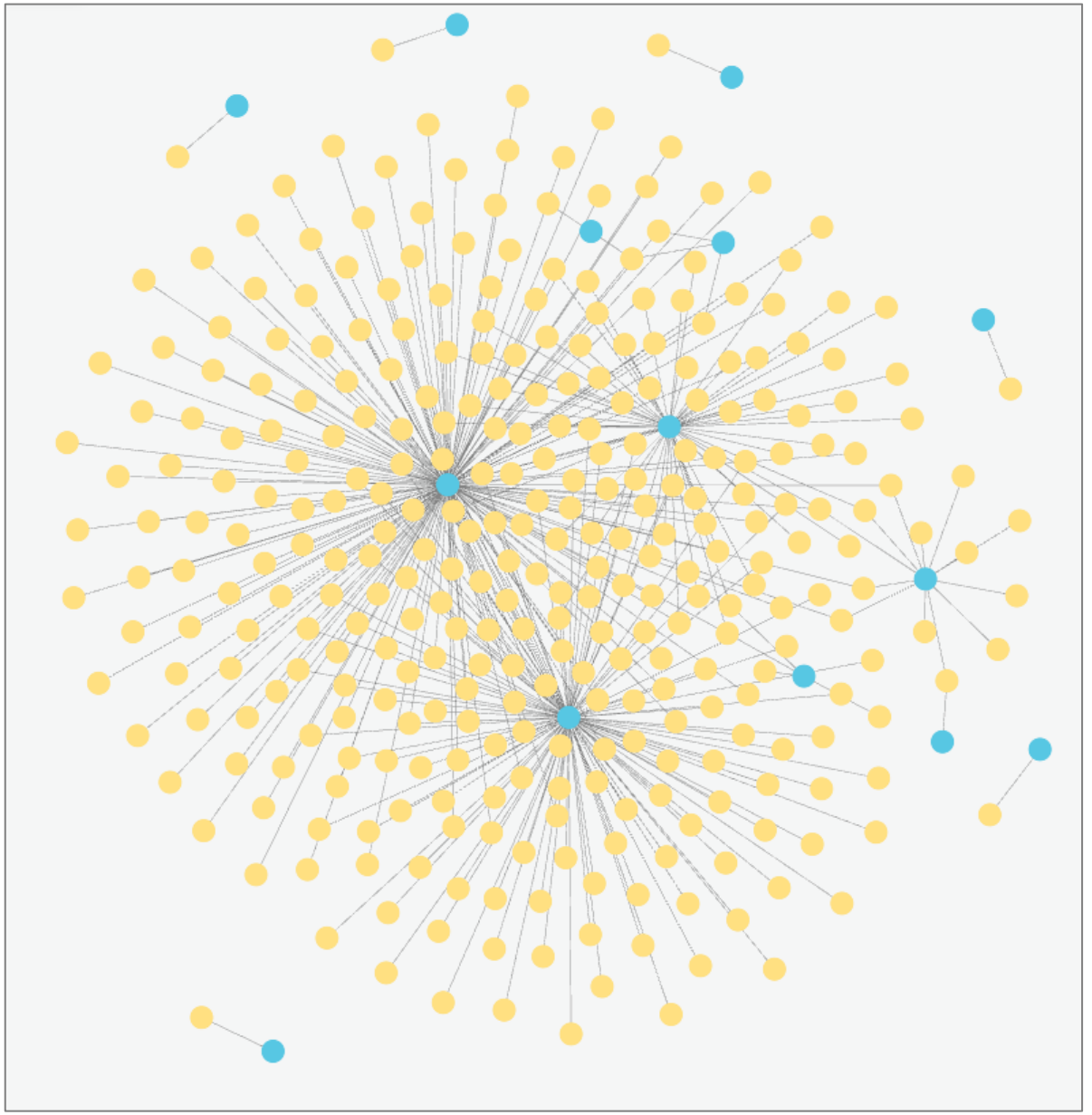
Creating labels and nodes can be easy.





Just start with well know data.

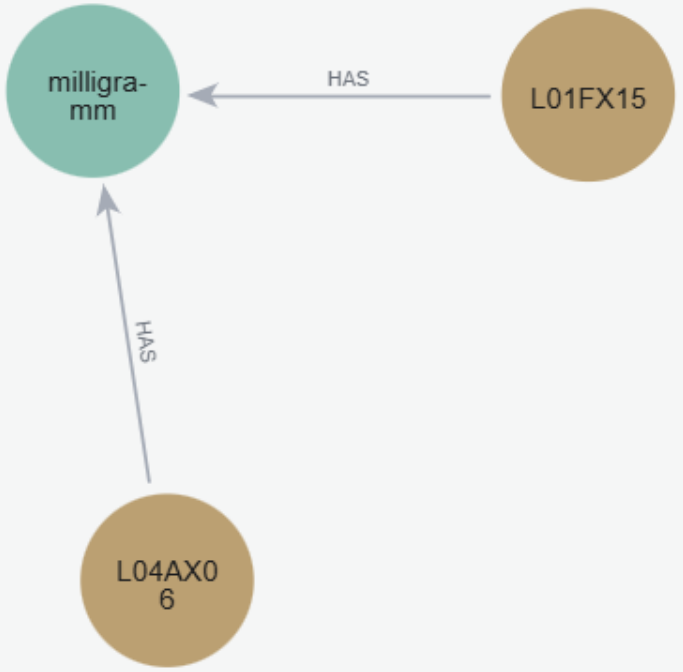




Graph algorithms are useless with “common attributes”

```
1 MATCH p = shortestPath((a:Substance)-[*..5]-(b:Substance))
2 WHERE a.Code='L04AX06' AND b.Code ='L01FX15'
3 RETURN p
```

Graph | Table | RAW



Results overview

Nodes (3)

* (3) | Substance (2) | Unit (1)

Relationships (2)

* (2) | HAS (2)

After adding data as attributes to nodes and deleting the “common labels”…

```
neo4j$ MATCH (n:Substance)-[:HAS]->(u:Unit) SET n.Unit = u.Code; MATCH (n:Substance)-[:HAS]->(a:Appl
```

```
MATCH (n:Substance)-[:HAS]→(u:Unit) SET n.Unit = u.Code;
```

```
MATCH (n:Substance)-[:HAS]→(a:Application) SET n.Application = a.Code;
```

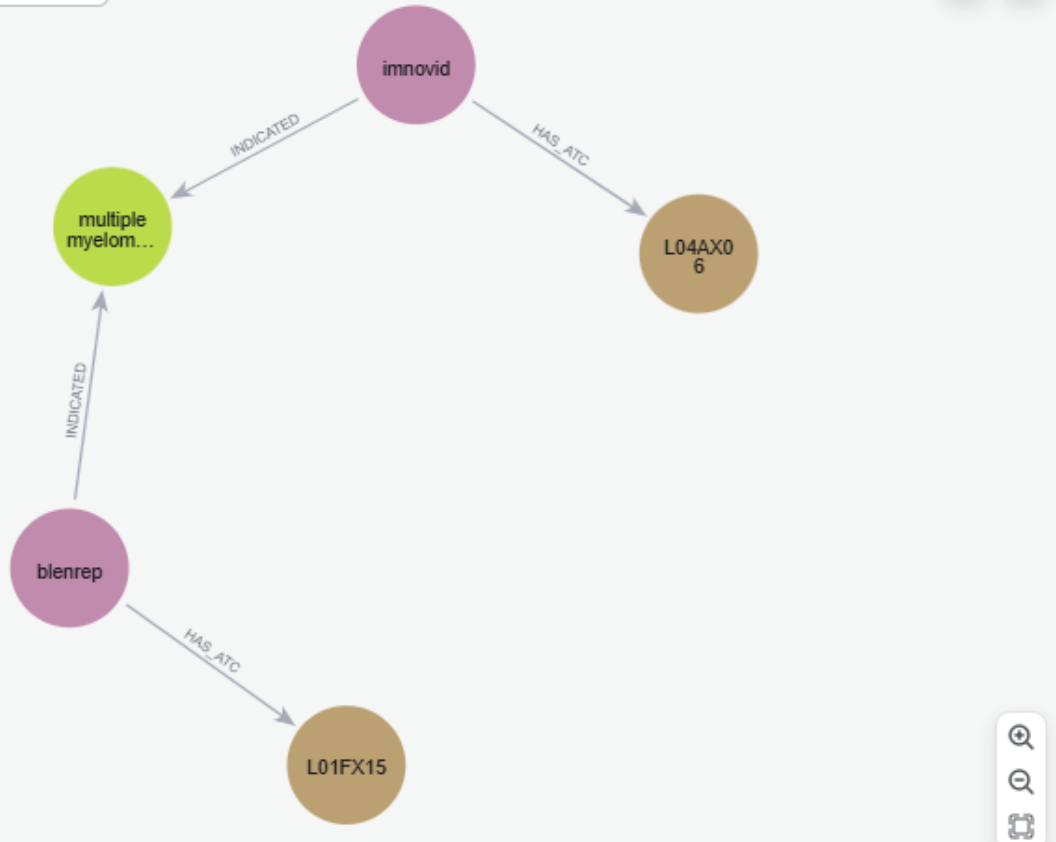
```
MATCH (n:Application) DETACH DELETE n;
```

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

… shortest path can be useful!

```
1 MATCH p = shortestPath((a:Substance)-[*..5]-(b:Substance))
2 WHERE a.Code='L04AX06' AND b.Code ='L01FX15'
3 RETURN p
```

Graph | Table | RAW



Results overview

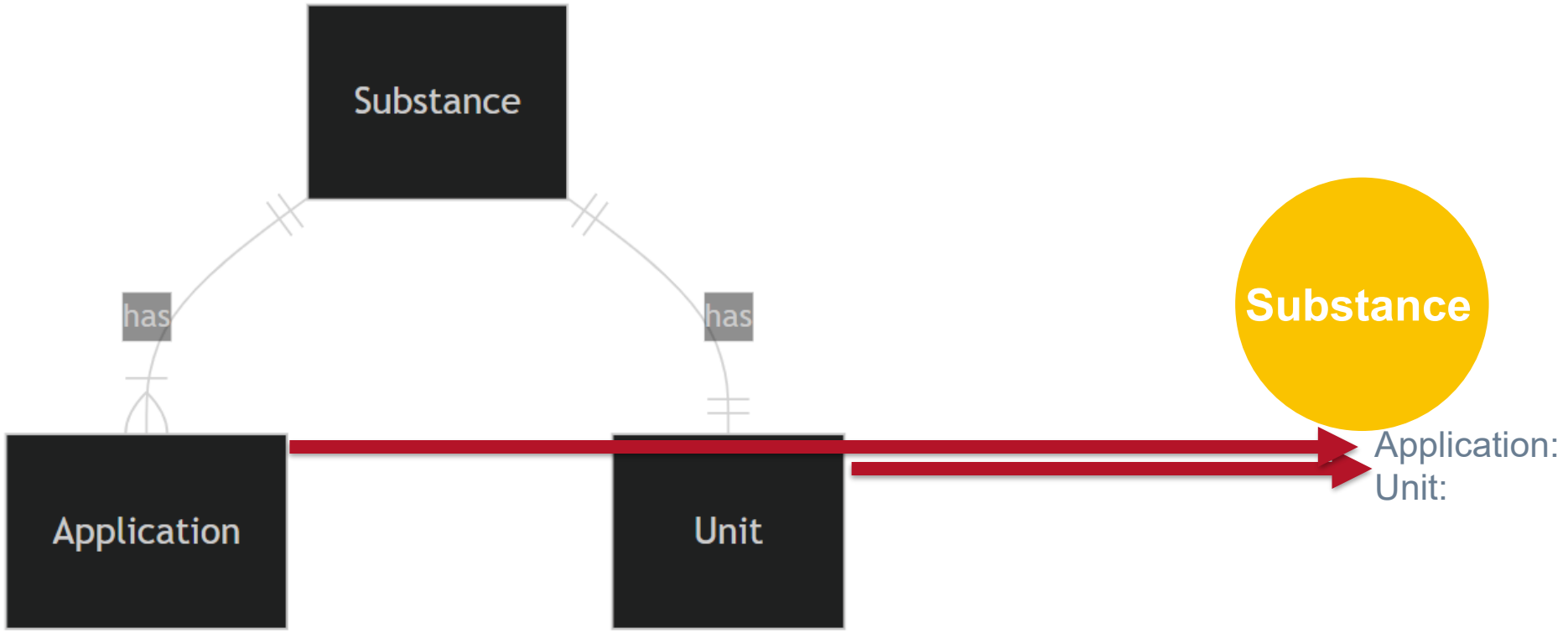
Nodes (5)

* (5) | Indication (1) | Medicament (2) | Substance (2)

Relationships (4)

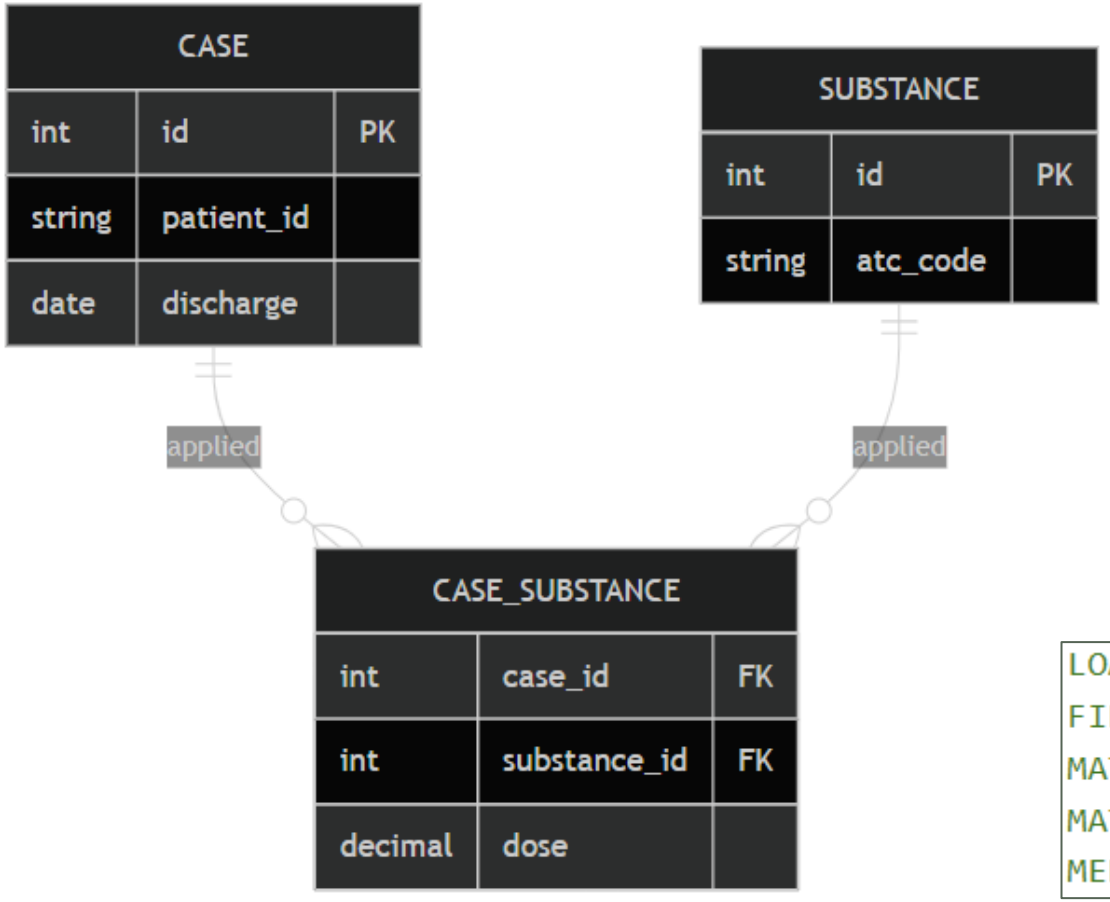
* (4) | HAS_ATC (2) | INDICATED (2)

Dimension tables (with only few rows) could be modelled as attributes and not as own labels.



Example 2

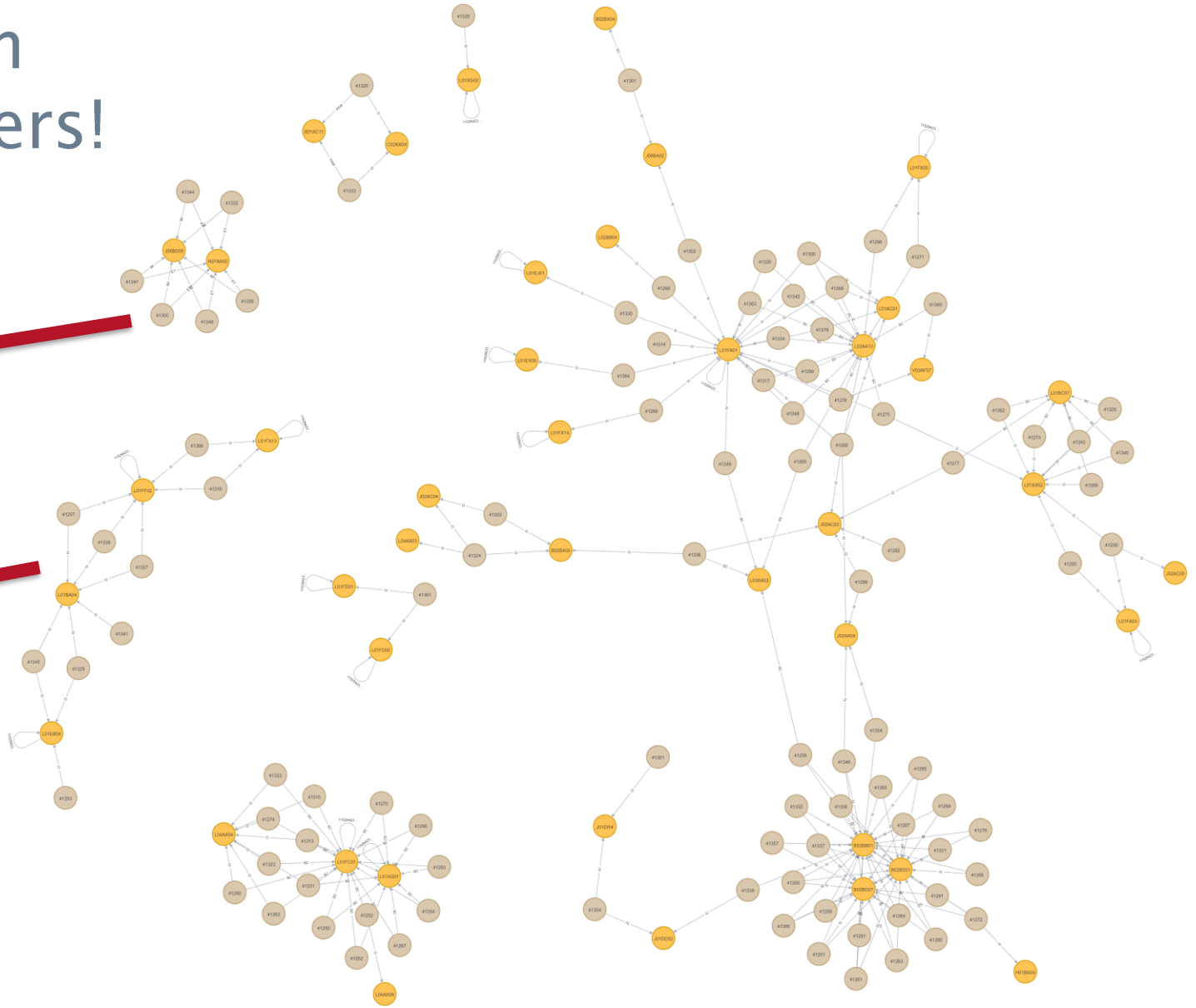
n:m table = graph relation




```
LOAD CSV WITH HEADERS FROM 'file:///case_substance.csv' AS row
FIELDTERMINATOR ';'
MATCH (c:Case {id: toInteger(row.case_id)})
MATCH (s:Substance {id: toInteger(row.substance_id)})
MERGE (s)-[:APPLIED {dose: toInteger(row.dose)}]->(c)
```

n:m table = graph relation
Not correlations but clusters!

	A	B	C	D	E	F	…	N
A	6	6						
B		6						
C			3	2				
D				6	3			
E					5	2		
F						2		
…							…	
N								…



Example 3

MERGE … SET is like UPDATE

Code	DDD	Application
L04AX06	3	O
B01AC21	4.3	
J01AA15	0.3, 0.1	O, P

```
1 MERGE (n:Substance {Code: 'L04AX06'}) SET n.DDD = 3, n.application = 'O';
2 MERGE (n:Substance {Code: 'B01AC21'}) SET n.DDD = 4.3;
3
4 // Creating "two" rows is not , because SET "updates" the node
5 MERGE (n:Substance {Code: 'J01AA15'}) SET n.DDD = 0.3, n.application = 'O';
6 MERGE (n:Substance {Code: 'J01AA15'}) SET n.DDD = 0.1, n.application = 'P';
```

Code	DDD	Application
L04AX06	3	
B01AC21	4.3	
J01AA15	0.3	O
J01AA15	0.1	P

```
nodes2025$ MATCH (n:Substance) RETURN n.Code, n.DDD, n.application
```

Table RAW

	n.Code	n.DDD	n.application
1	"L04AX06"	3	"O"
2	"B01AC21"	4.3	null
3	"J01AA15"	0.1	"P"

Attributes with arrays are normal and not the exception (unwind in neo4j is like unnest in PostgreSQL)

```
MERGE (n:Substance {Code: 'J01AA15'})
  SET n.DDD = [0.3,0.1], n.application = ['O','P'] ;
```

```
MATCH (n:Substance)
UNWIND(n.DDD) as ddd_without_array
RETURN n.Code, ddd_without_array;
```

Table | RAW

	n.Code	ddd_without_array
1	"L04AX06"	3
2	"B01AC21"	4.3
3	"J01AA15"	0.3
4	"J01AA15"	0.1

```
MATCH (n:Substance)
UNWIND(n.DDD) as ddd_without_array
RETURN round(sum(ddd_without_array),2);
```

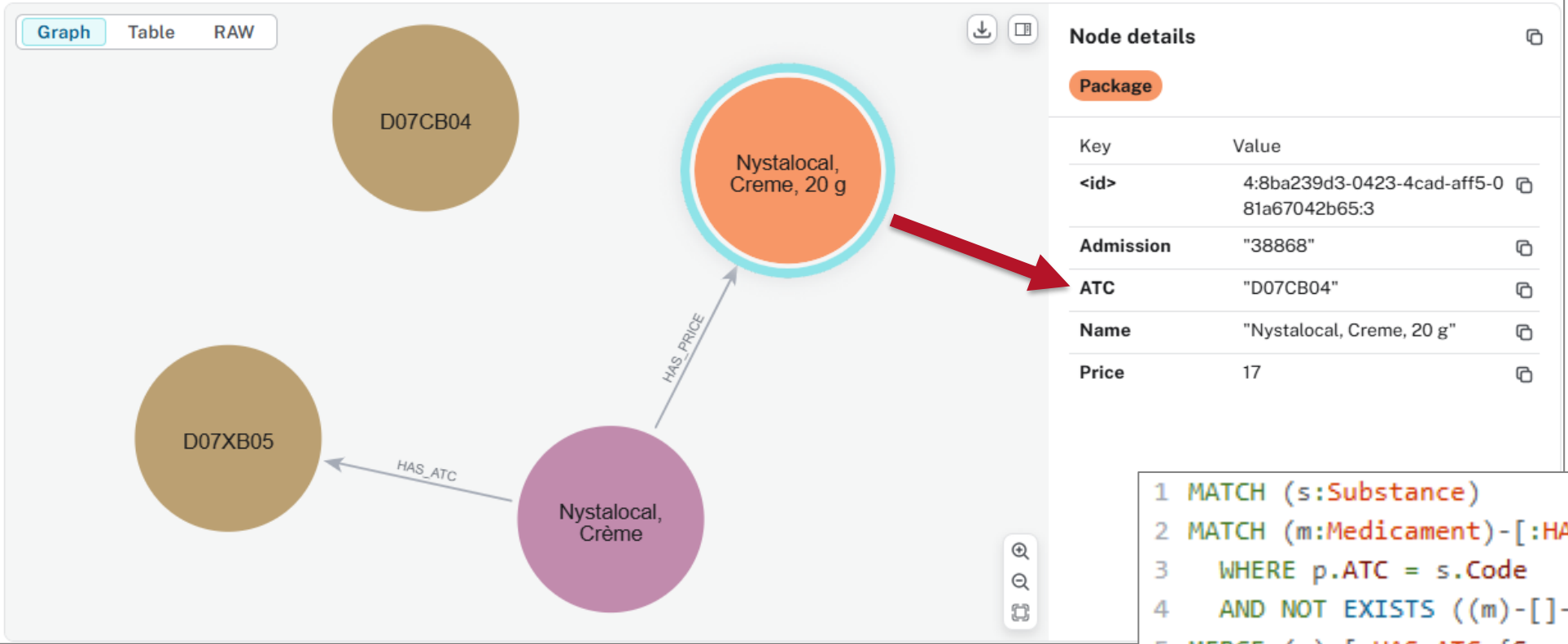
Table | RAW

	round(sum(ddd_w
1	7.7

Example 4

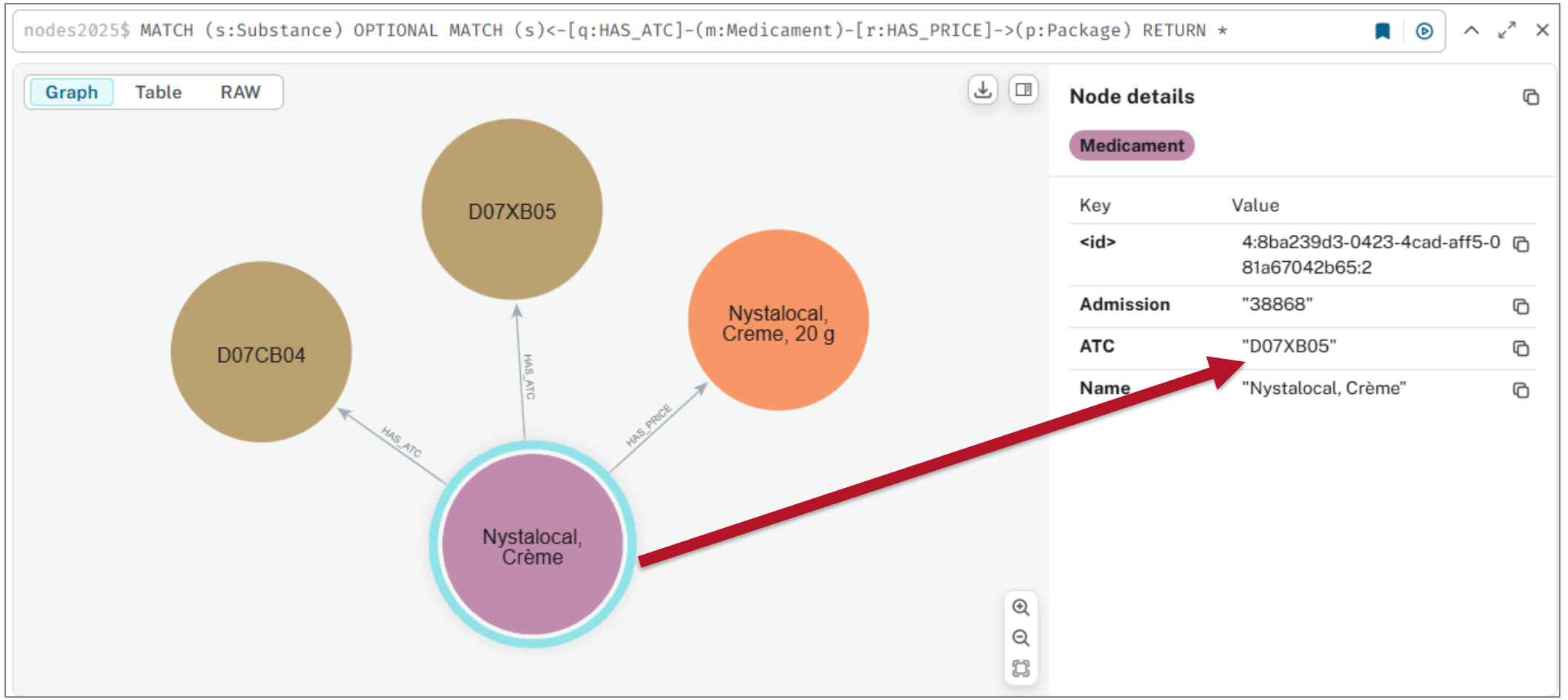
You can add “indirect” relations with information from other nodes in a further modelling phase, if you stored them.

```
MATCH (s:Substance)
OPTIONAL MATCH (s)<-[q:HAS_ATC]-(m:Medicament)-[r:HAS_PRICE]->(p:Package)
RETURN *
```



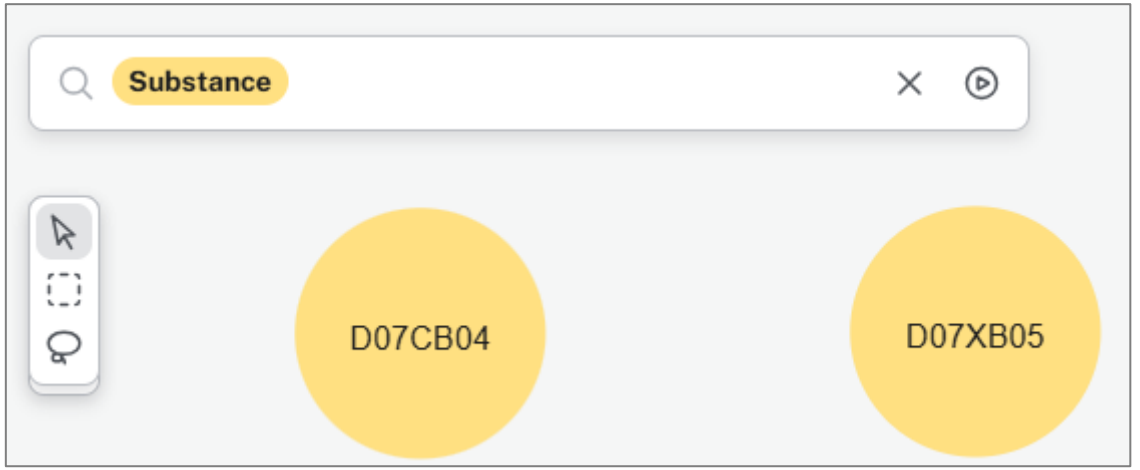

```
MATCH (s:Substance)
MATCH (m:Medicament)-[:HAS_PRICE]->(p:Package)
  WHERE p.ATC = s.Code
  AND NOT EXISTS ((m)-[]->(s))
MERGE (m)-[:HAS_ATC {Source: 'Indirect'}]->(s);
```

With “multiple” relations you enrich your graph and you need fewer complex data cleaning.



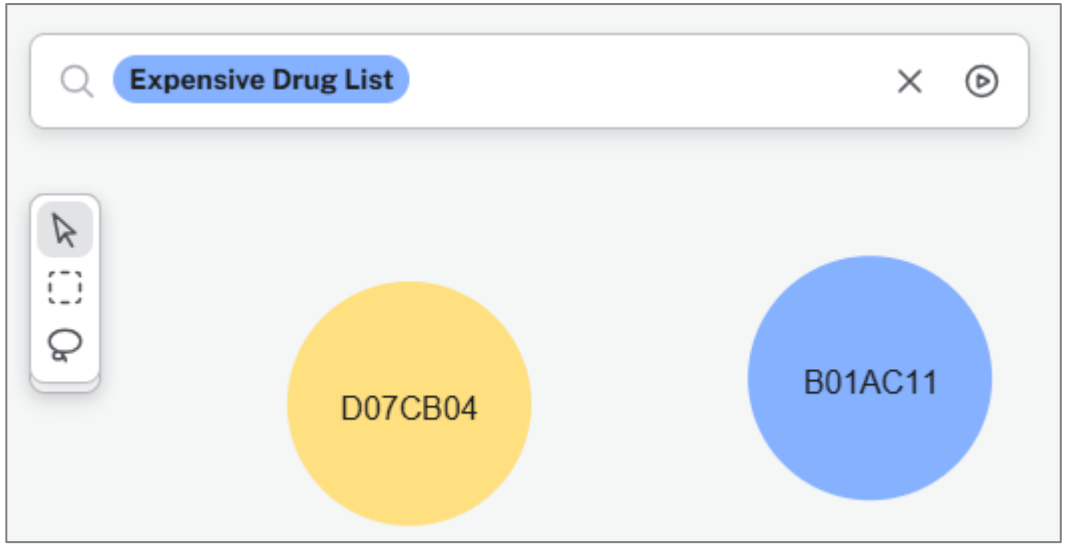
Example 5

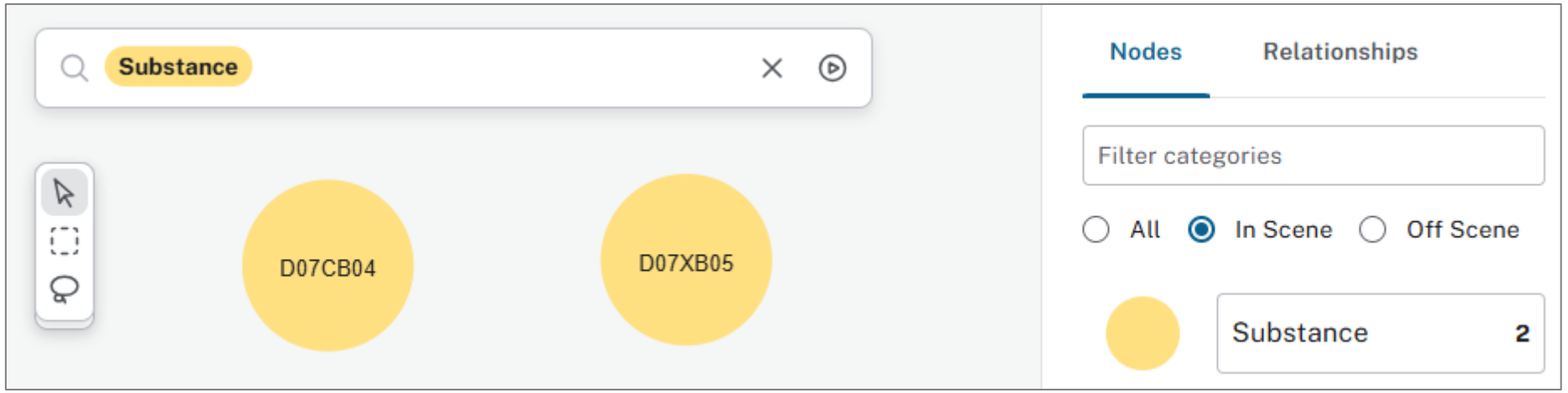
Now we use Explore/Bloom with same substance nodes like before.



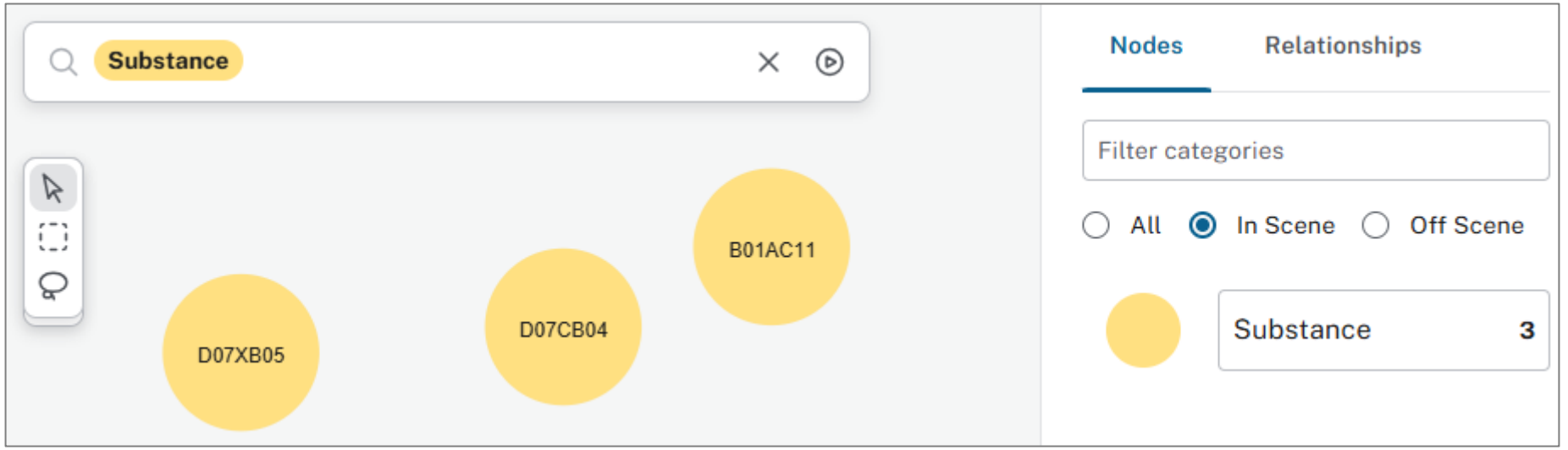
We introduce a new label.

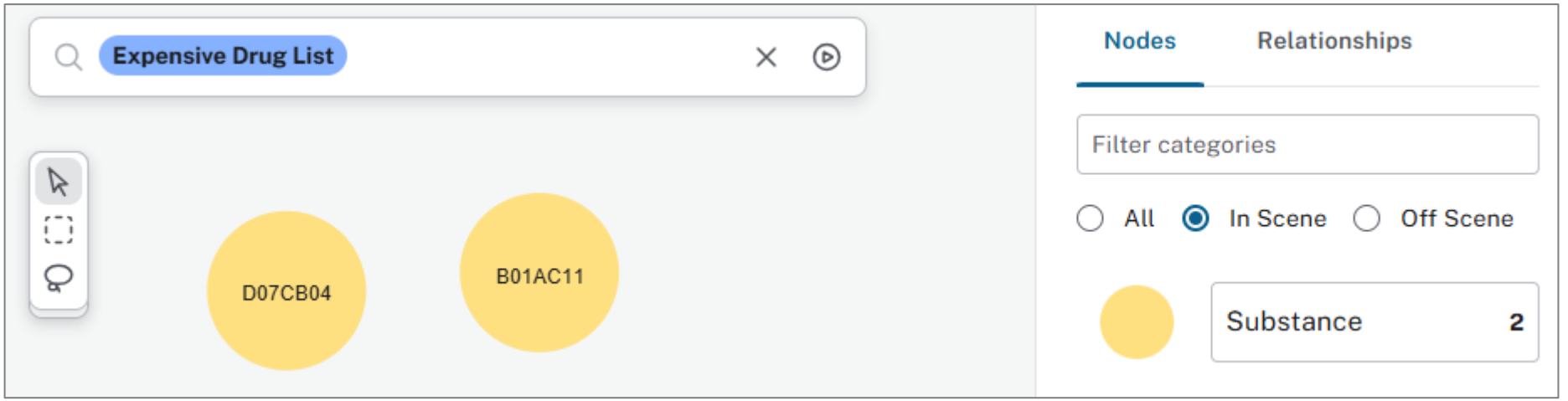
```
// Create new label
MERGE (n:Substance {Code: 'D07CB04'})
SET n:`Expensive Drug List`;

// Create new node with new label
MERGE (n:`Expensive Drug List` {Code: 'B01AC11'})
SET n.Description = 'Iloprost';
```





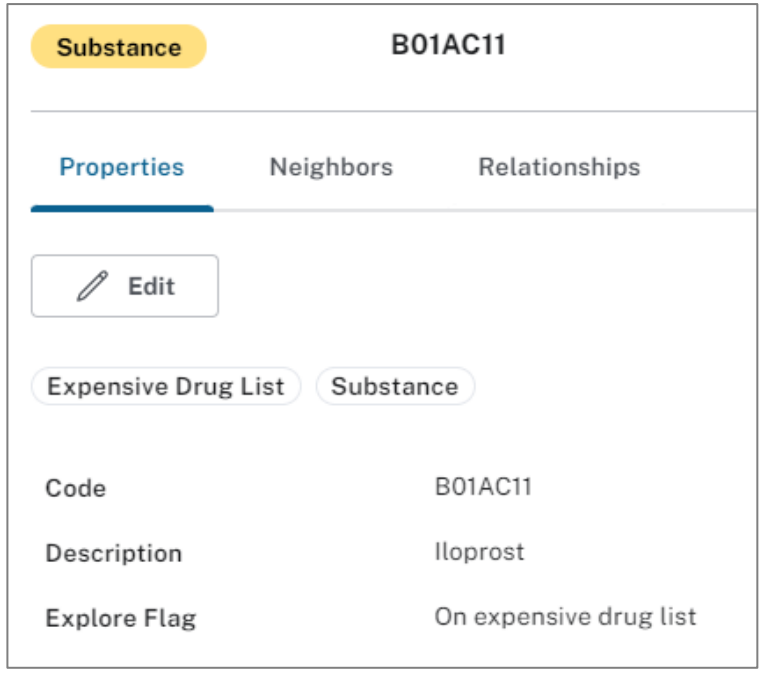
```
// Set new node with "old" label
MATCH (n:`Expensive Drug List`) WHERE n.Code = 'B01AC11'
SET n:Substance;
```

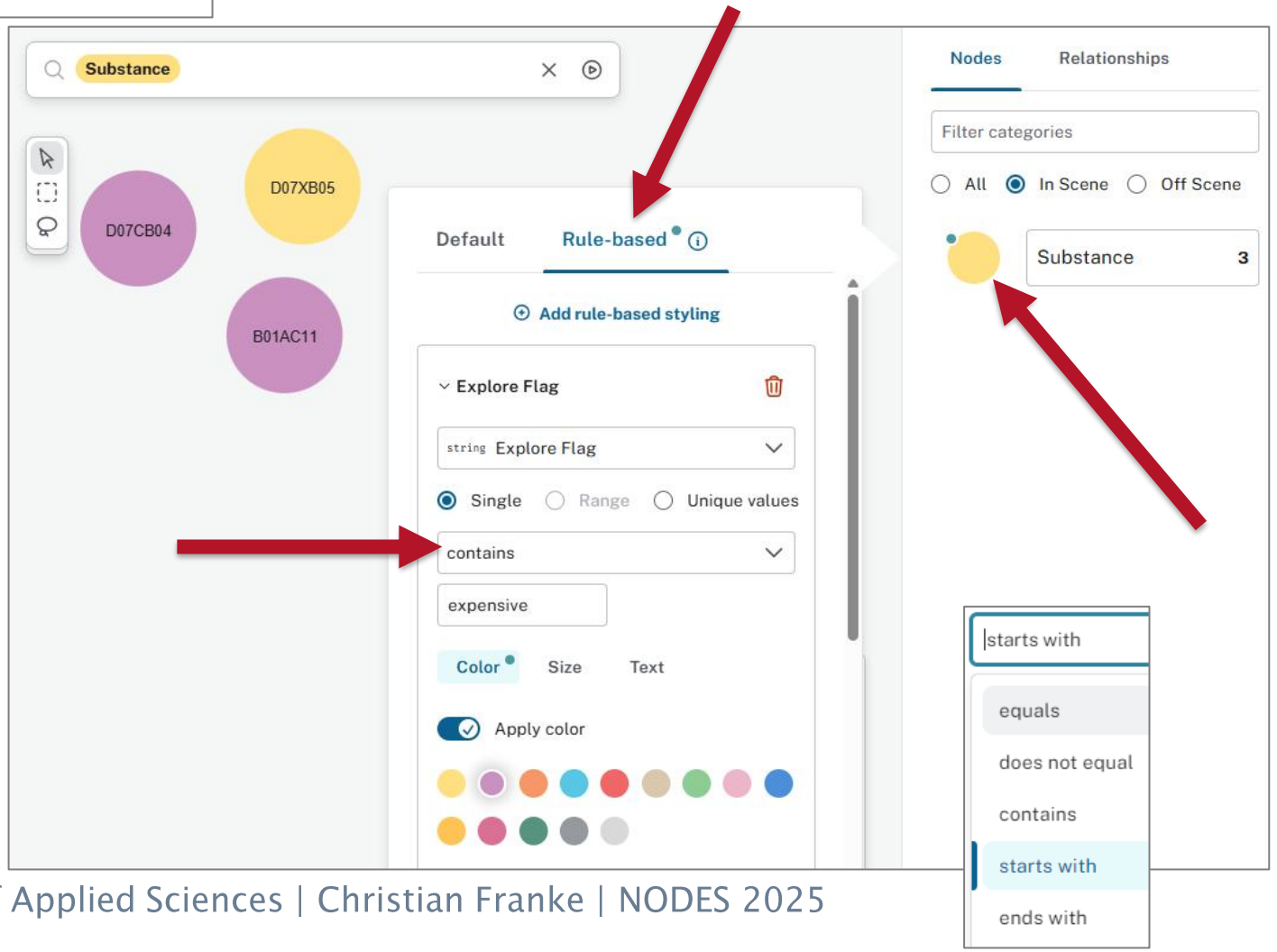


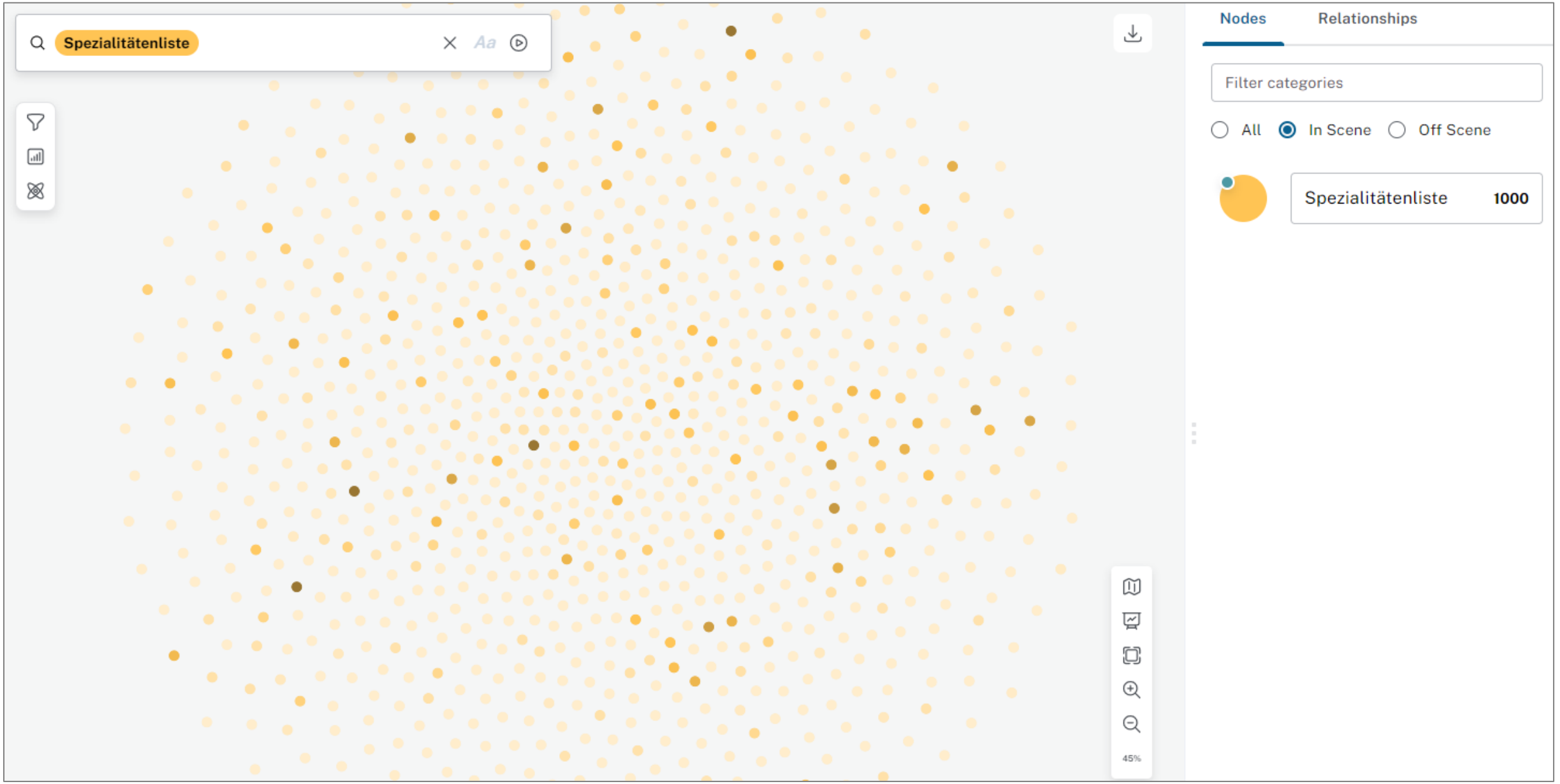


After setting an attribute, you can use rule-based colors.

```
MATCH (n:`Expensive Drug List`)
SET n.`Explore Flag` = 'On expensive drug list';
```



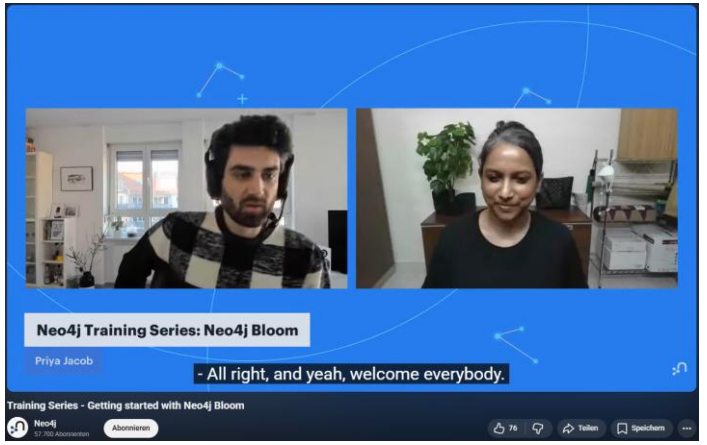



Spezialitätenliste
Aa
Nodes
Relationships
Filter categories
All
In Scene
Off Scene
Spezialitätenliste
1000
45%

Example 6

If perspectives have version numbers, you probably must change deeplinks in other applications.

```
https://console-preview.neo4j.io/tools/explore?
search=Substance%20informations%20for%20L01FY01&
perspective=Medi-Graph%20V1.0&
run=true
```



Training Series – Getting started with Neo4j Bloom
https://www.youtube.com/live/7yS2e4p0_H4

Summary

Summary

- Be careful with transferring “hierarchy” or “dimension tables” from RDBMS to graphs: probably better use attributes and not lables
- In graphs you can create edges instead of n:m tables
- Arrays are normal in graphs and not the exception (“forget” 3NF or BCNF)
- Flexible relationships in graphs are useful, especially if there is “tidy” data in your source
- Store important codes as attribute in the nodes, so that you can “recycle” them
- Bloom/Explore offers a lot of individual solutions with saved cyphers, colors, symbols, sizes etc.
- Don’t use version numbers for important perspectives in Bloom/Explore

Contact and further information

linkedin.com/in/christian-franke-b7b7a238b/

github.com/teletrabbie/nodes2025

christianfranke.quarto.pub