Graph Databases, including Knowledge Graphs

Dr Mercedes Arguello Casteleiro
Lecturer, University of Southampton, UK
Visiting academic, University of Manchester, UK

University of Southampton

Top-10 Graph databases

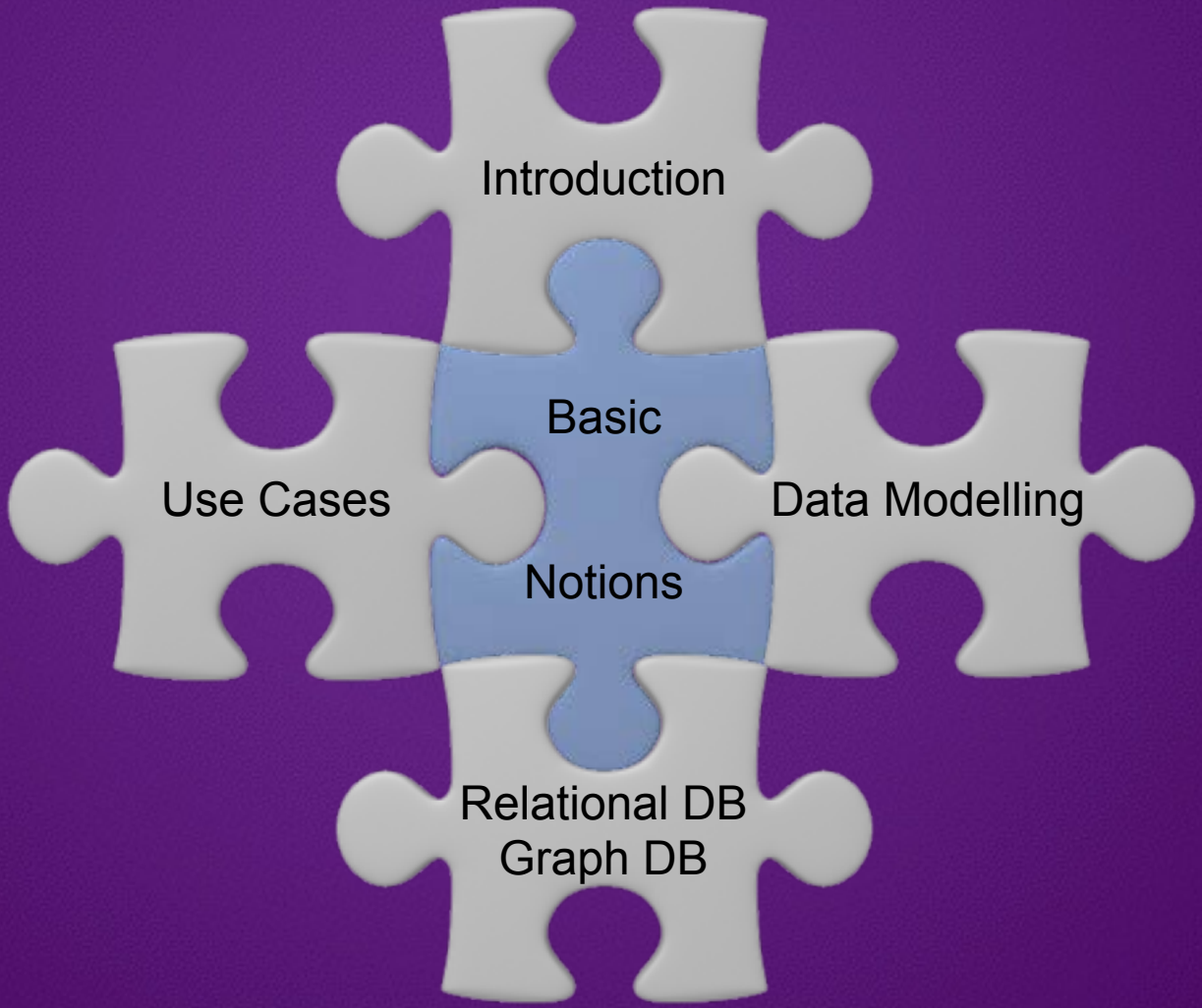
https://db-engines.com/en/ranking/graph+dbms

39 systems in ranking, June 2023

Rank Jun 2023	Rank May 2023	Rank Jun 2022	DBMS	Database Model	Score Jun 2023	Score May 2023	Score Jun 2022
1.	1.	1.	Neo4j	Graph	52.77	+1.66	-6.76
2.	2.	2.	Microsoft Azure Cosmos DB	Multi-model	36.57	+0.58	-4.41
3.	3.	3.	Virtuoso	Multi-model	5.24	-0.33	-0.93
4.	4.	4.	ArangoDB	Multi-model	4.89	+0.01	-0.61
5.	5.	5.	OrientDB	Multi-model	4.53	+0.03	-0.33
6.	6.	6.	Amazon Neptune	Multi-model	3.03	+0.13	+0.21
7.	7.	↑ 8.	JanusGraph	Graph	2.83	+0.15	+0.43
8.	8.	↑ 19.	Memgraph	Graph	2.82	+0.18	+2.38
9.	9.	↑ 15.	NebulaGraph	Graph	2.72	+0.11	+1.61
10.	10.	↓ 7.	GraphDB	Multi-model	2.55	+0.07	-0.07

MANCHESTER 1824
The University of Manchester

Part 1: Graph databases

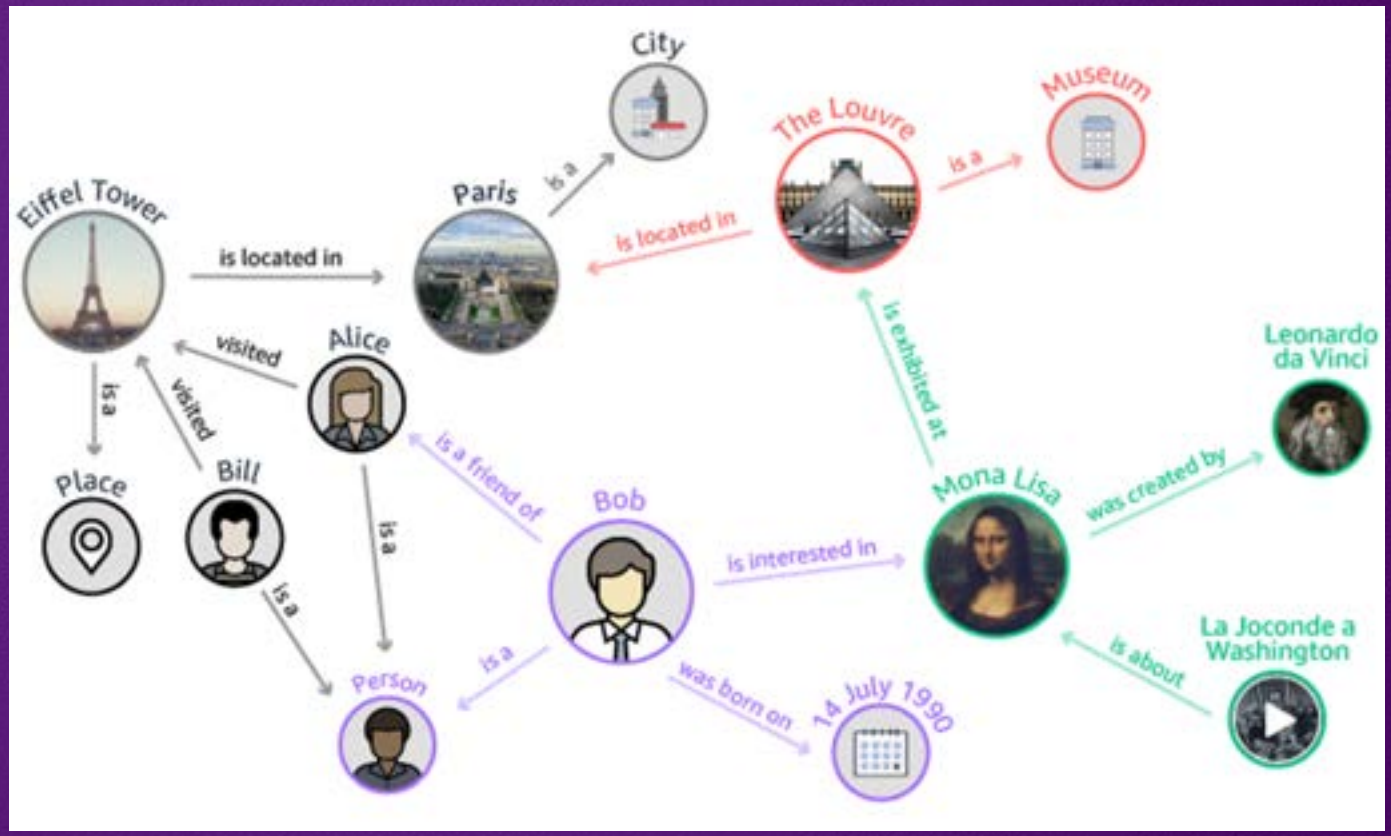


MANCHESTER 1824
The University of Manchester

Using Amazon Neptune to build an Enterprise Knowledge Graph

https://aws.amazon.com/neptune/knowledge-graphs-on-aws/

A knowledge graph captures the semantics of a particular domain

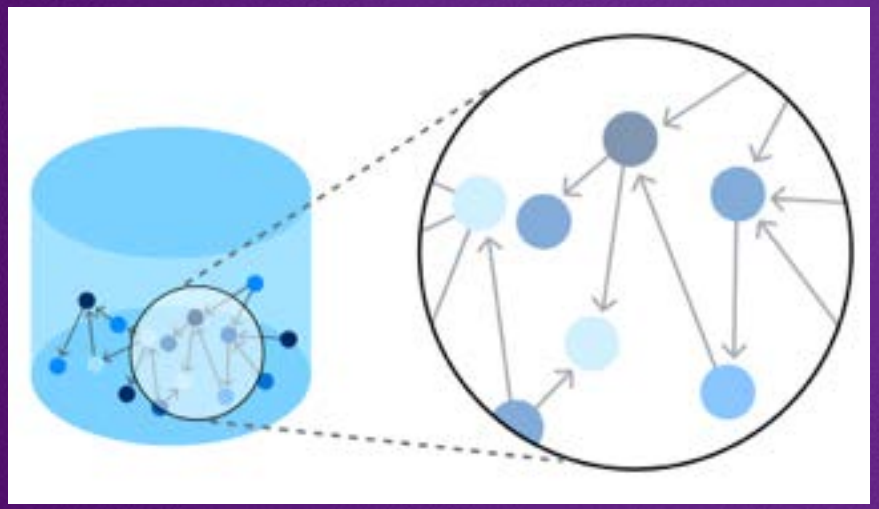


Graphs as networks

MANCHESTER 1824
The University of Manchester

Neo4j graph platform

Neo4j is the company behind Neo4j graph platform



Neo4j database is designed to store,
reveal, and
query relationships

Customers

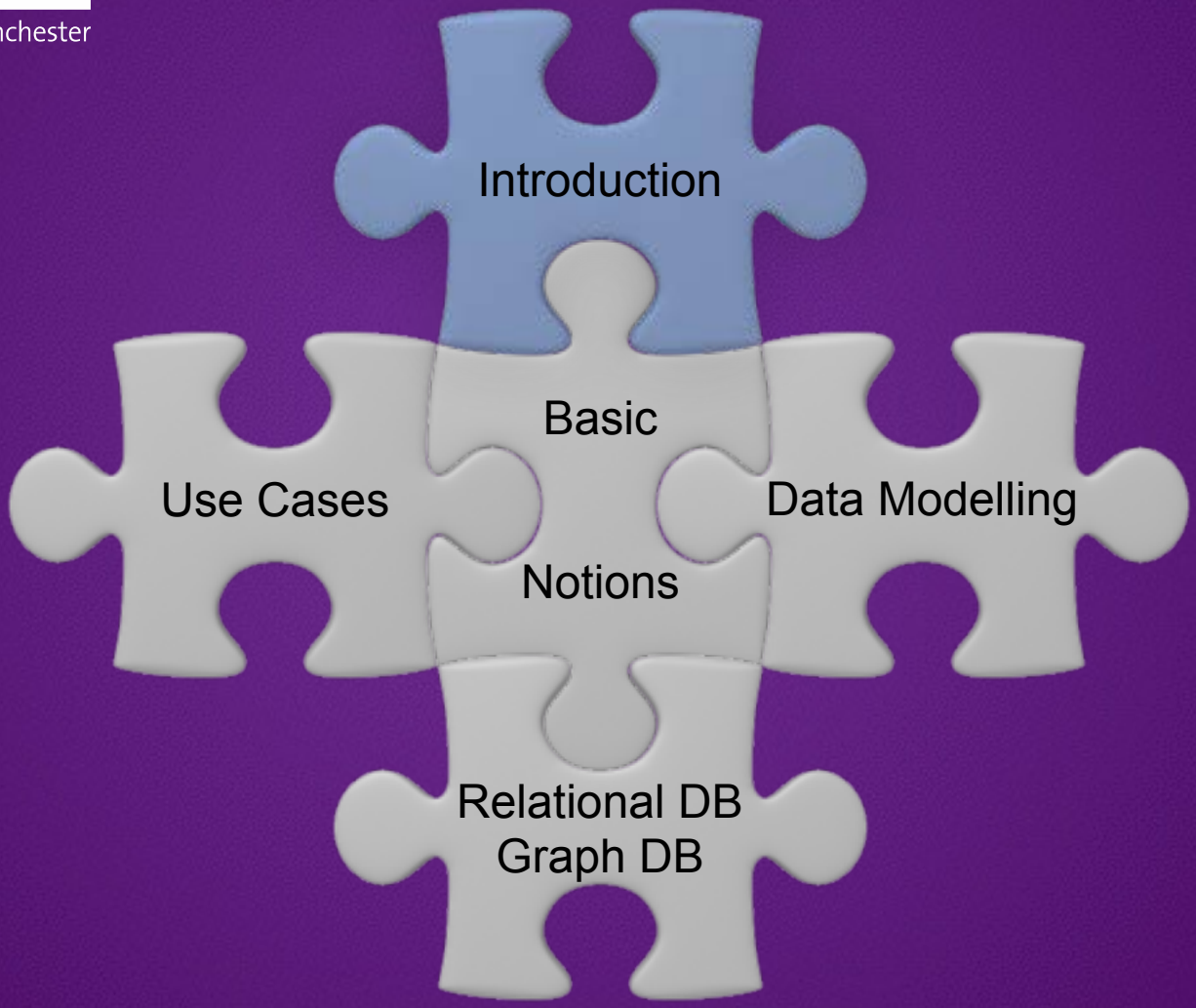
LinkedIn
UBS
COMCAST
cisco
Lufthansa
Walmart
ebay

Partners

Google Cloud Platform
amazon web services
IBM
Microsoft Azure
TATA
LINKURIOUS
pitney bowes
GraphAware
structr

https://neo4j.com/product/

MANCHESTER 1824
The University of Manchester

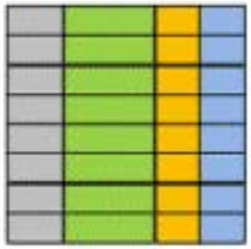


MANCHESTER 1824
The University of Manchester

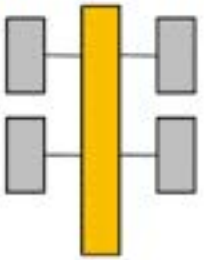
Evolution of DBs

SQL database

Relational



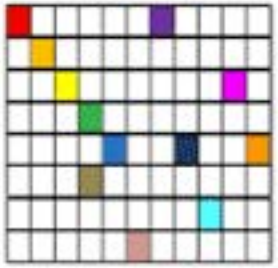
Analytical (OLAP)



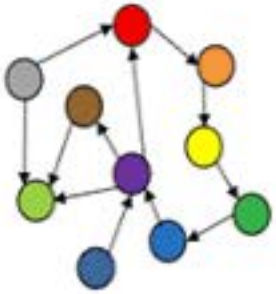
online analytical processing (OLAP)

NoSQL database

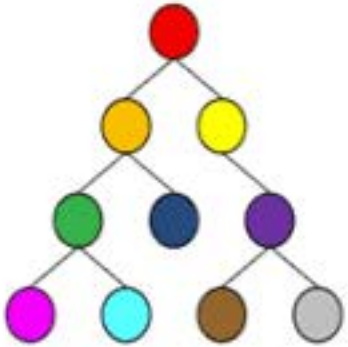
Column-Family



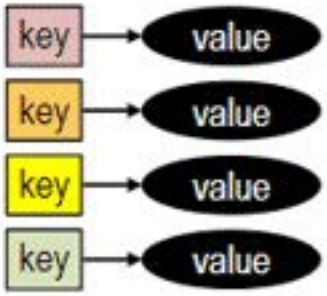
Graph



Document



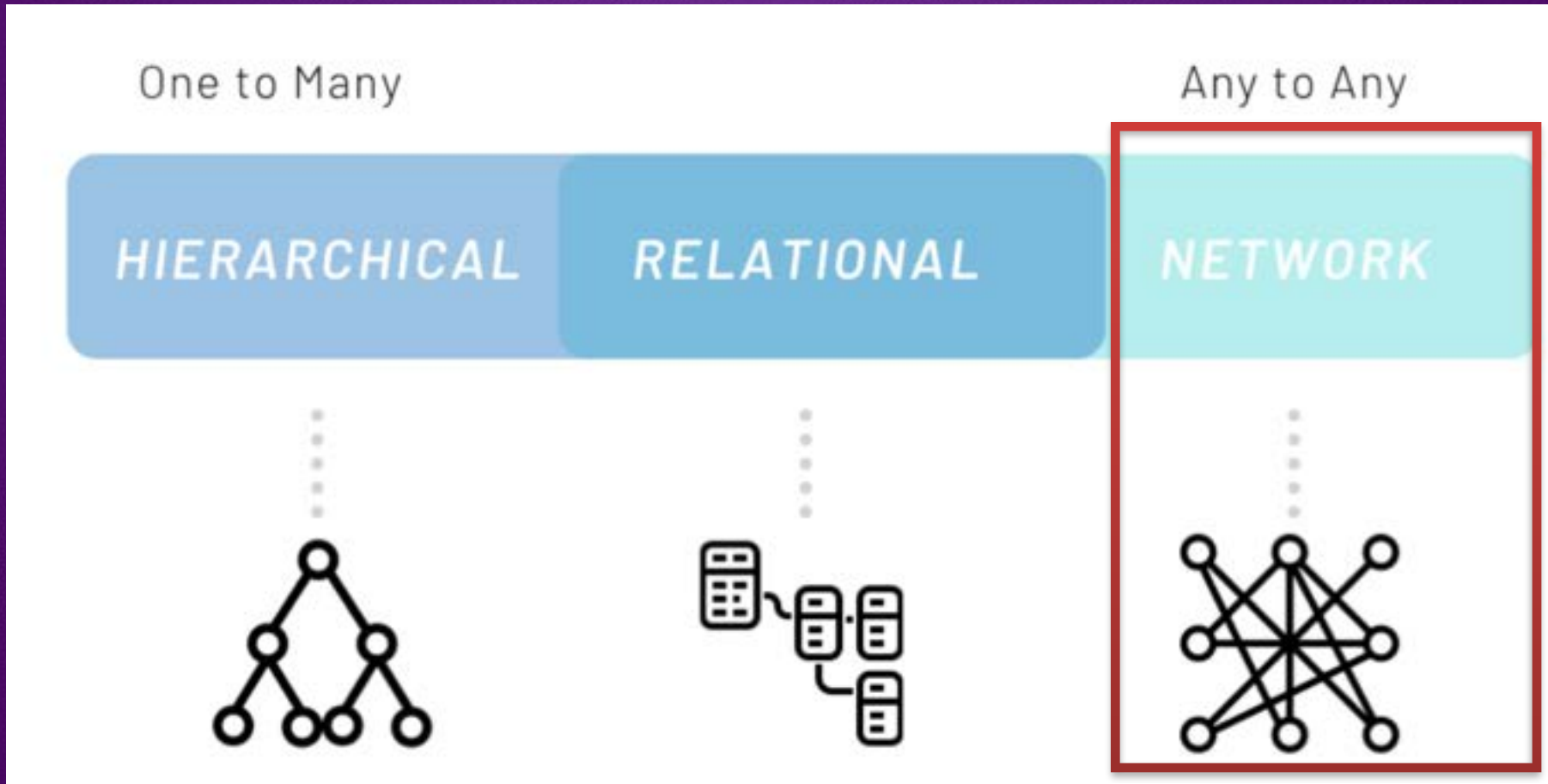
Key-Value



MANCHESTER 1824
The University of Manchester

…so… why using graph DBs?

For highly interconnected data,
graph model seems the most natural.



https://neo4j.com/developer-blog/put-data-modeling-on-the-table/

MANCHESTER 1824
The University of Manchester

What problems the graph DBs were created to solve?

Data Measurement Chart	
Data Measurement	**Size**
Bit	Single Binary Digit (1 or 0)
Byte	8 bits
Kilobyte (KB)	1,024 Bytes
Megabyte (MB)	1,024 Kilobytes
Gigabyte (GB)	1,024 Megabytes
Terabyte (TB)	1,024 Gigabytes
Petabyte (PB)	1,024 Terabytes
Exabyte (EB)	1,024 Petabytes

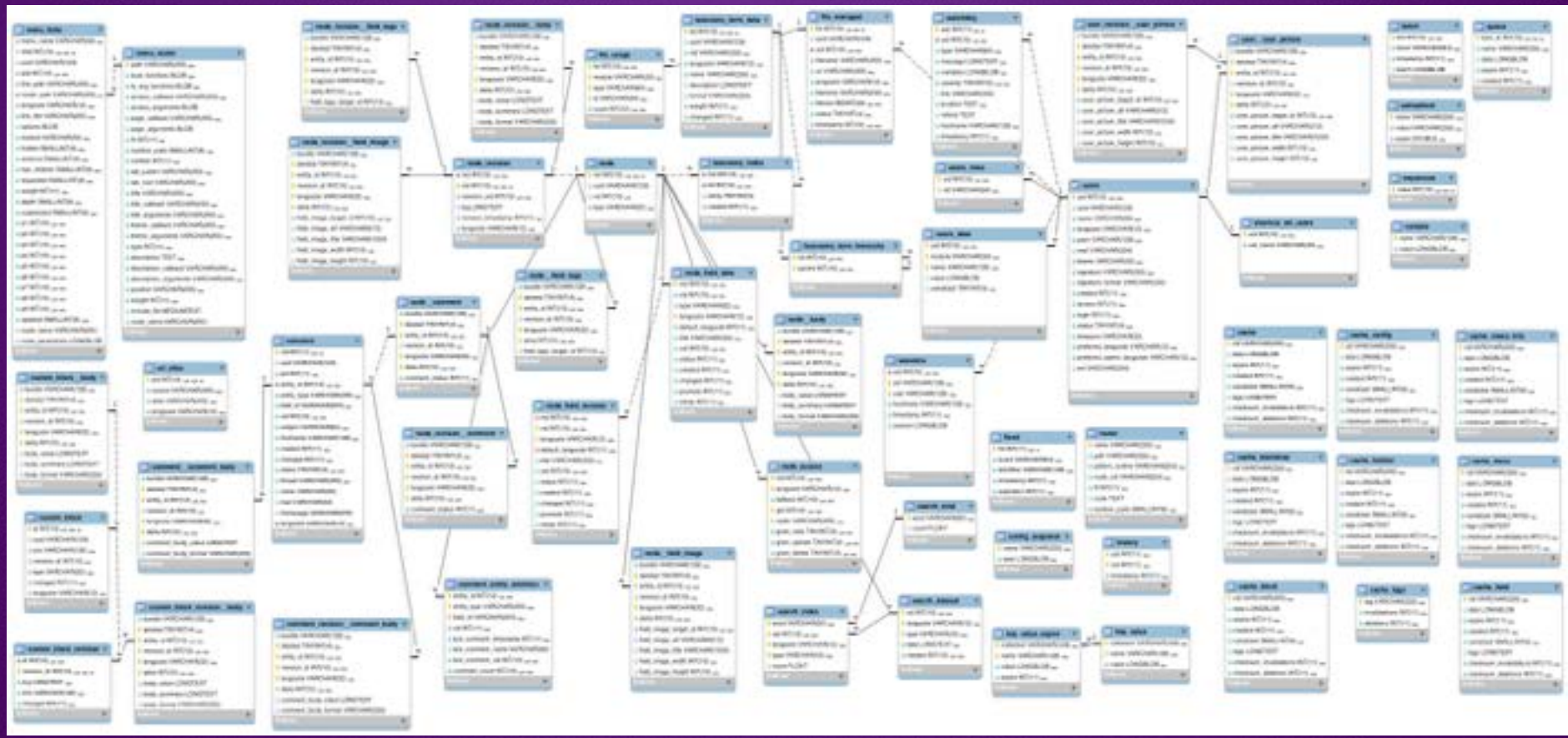
With the advent of the Cloud, there has been a data explosion with Exabytes of data available

https://innovationatwork.ieee.org/growing-cloud-computing-utilization-in-2019/

MANCHESTER 1824
The University of Manchester

What problems the graph DBs were created to solve?

The normalisation of data gets more and more expensive as the data size grows



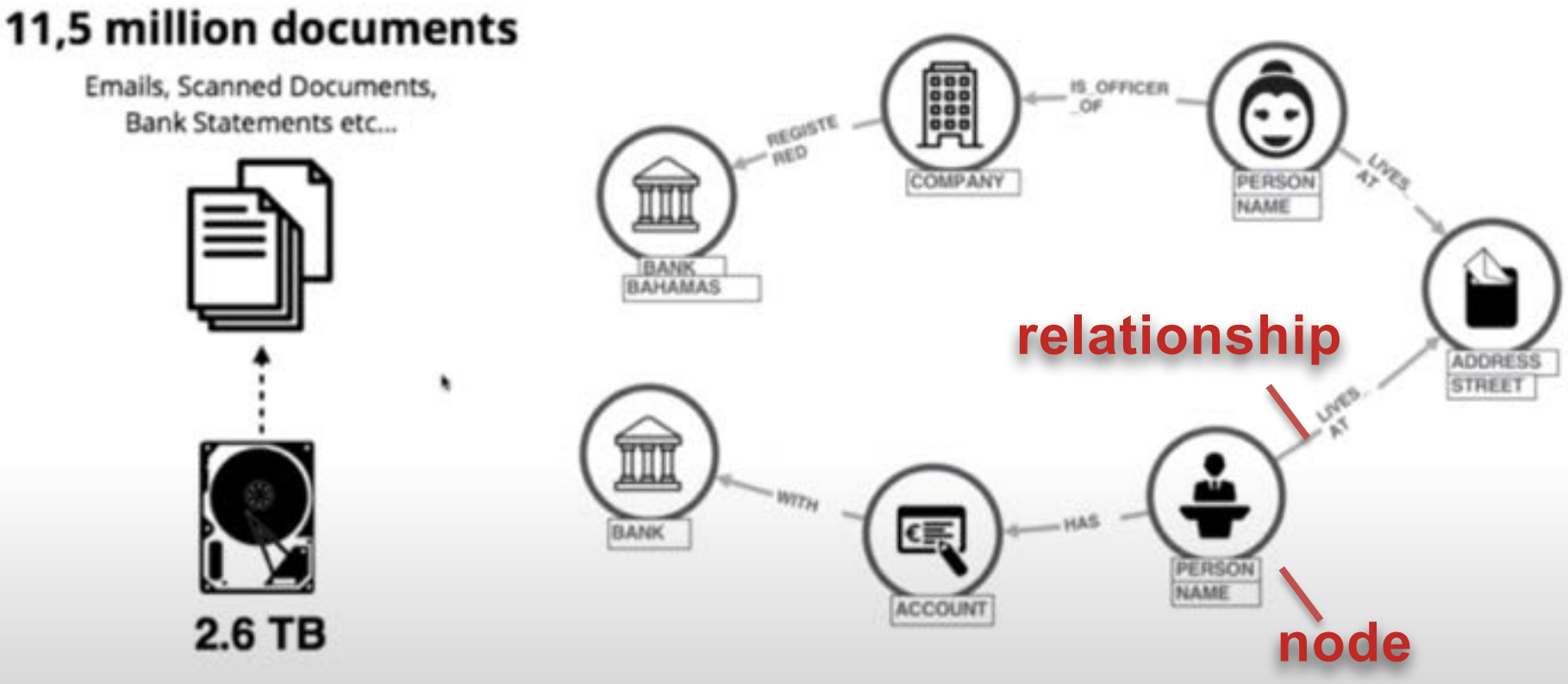
10s

https://neo4j.com/blog/history-and-future-of-graph-data/

MANCHESTER 1824
The University of Manchester

What problems the graph DBs were created to solve?

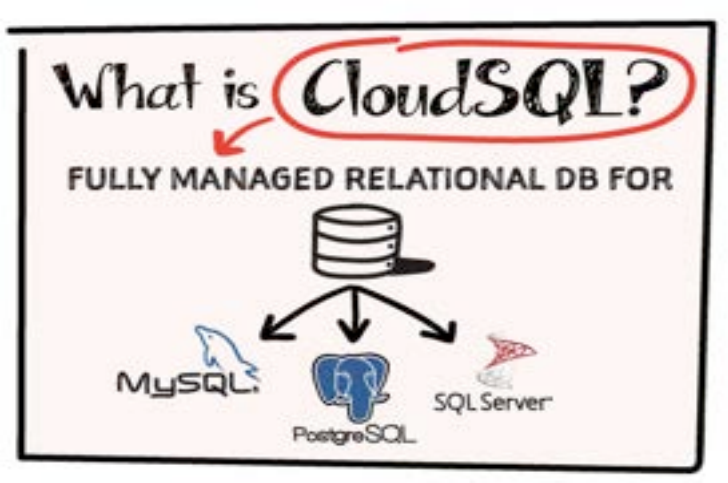
...so... how the graph DBs deal with data explosion?



https://neo4j.com/developer/guide-data-modeling/
https://www.youtube.com/watch?v=xpvafTZg0xA

MANCHESTER 1824
The University of Manchester

What problems the graph DBs were created to solve?



With the advent of the Cloud, there has been a data explosion with Exabytes of data available

https://innovationatwork.ieee.org/growing-cloud-computing-utilization-in-2019/

Another difficulty that is equally important is that, the data becomes also difficult to maintain needing complicated joined queries

12s

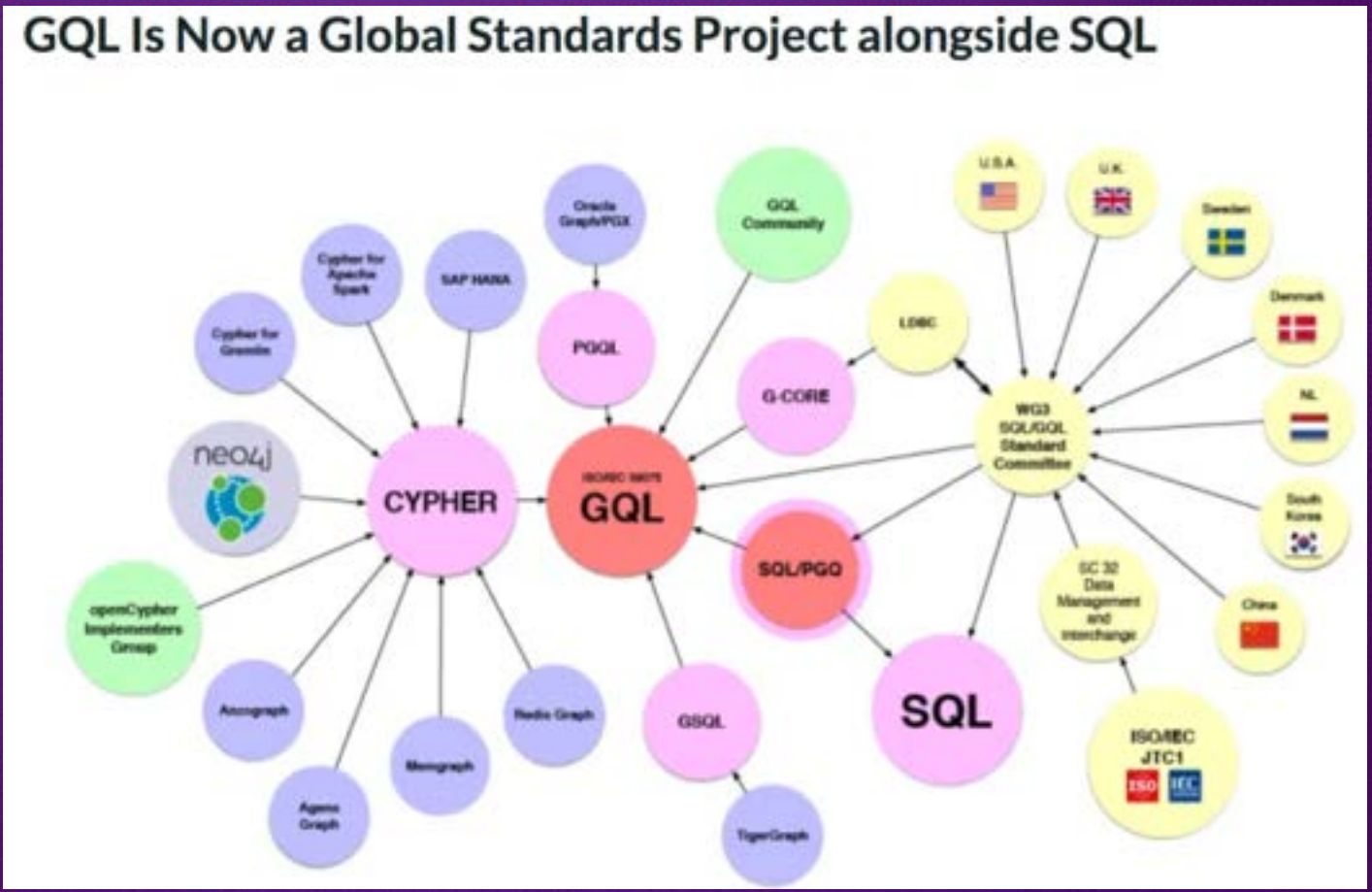
MANCHESTER 1824
The University of Manchester

GQL Manifesto

GQL stands for Graph Query Language

GQL Manifesto similar to SQL or Third manifesto

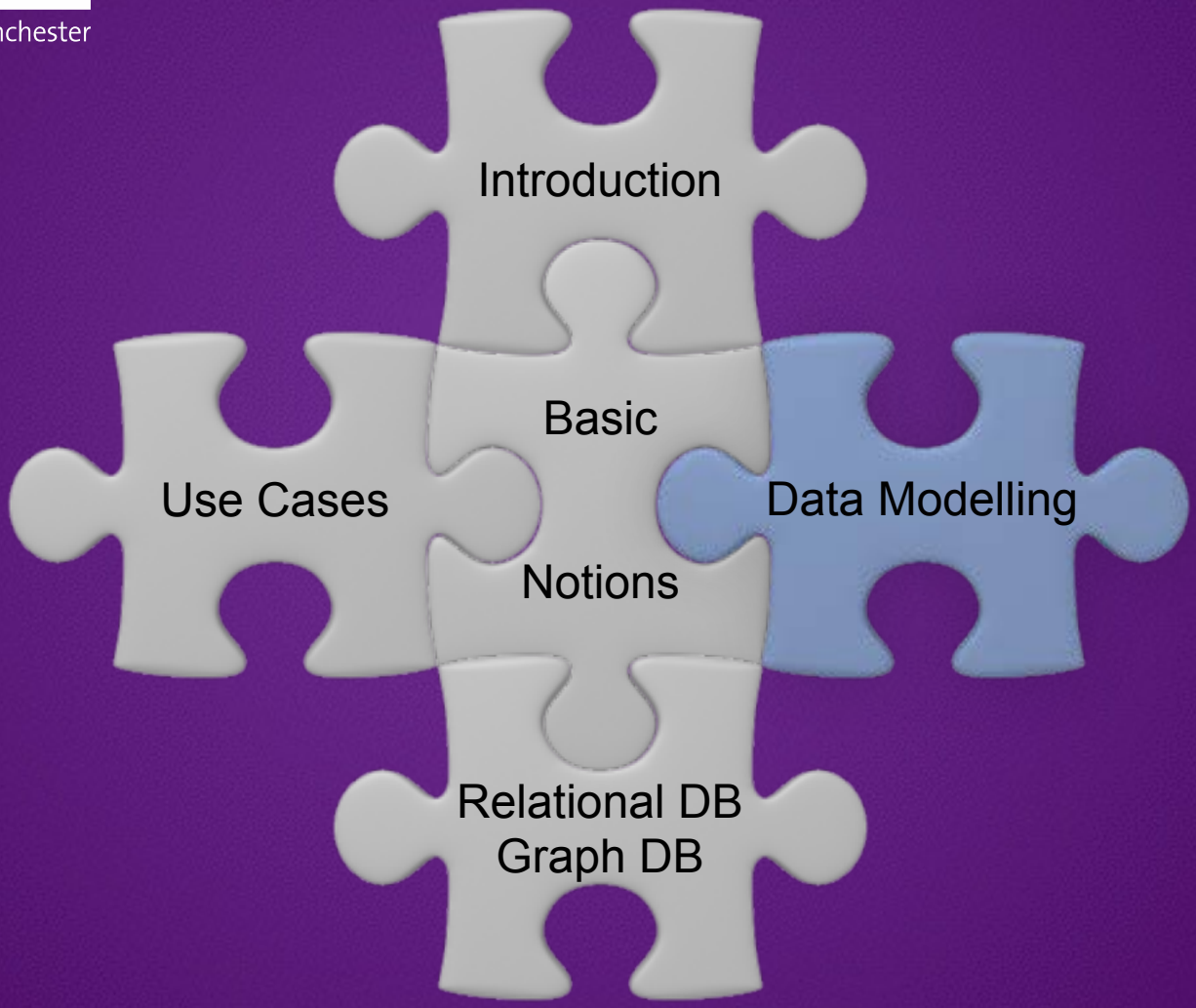
Like SQL, the new GQL (Graph Query Language) needs to be an industry standard.



GQL (last Update): September 16, 2019

https://gql.today/

MANCHESTER 1824
The University of Manchester



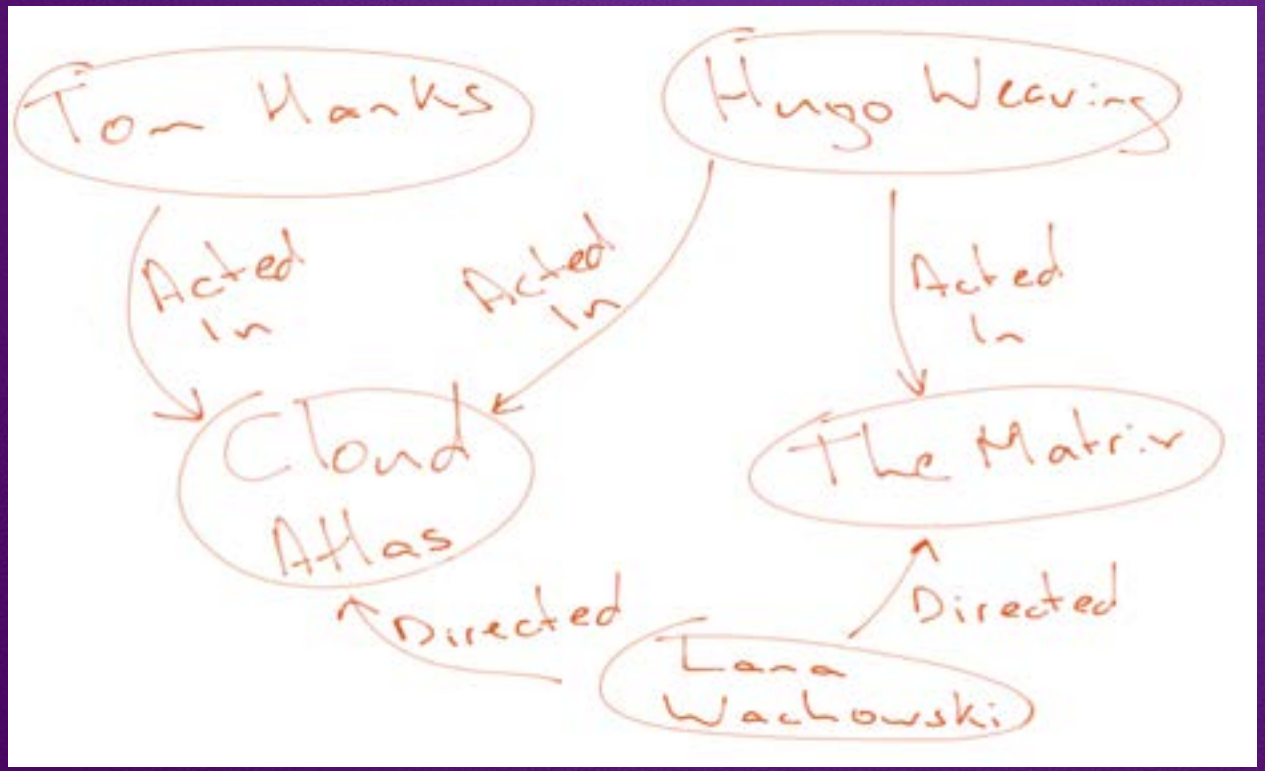
MANCHESTER 1824
The University of Manchester

Properties of graph DBs

For a developer, attractive properties of a graph DB are

Intuitiveness = whiteboard-friendly

The graph model is intuitive and easy to understand.



Intuitiveness
Speed
Agility

https://neo4j.com/developer/guide-data-modeling/

15s

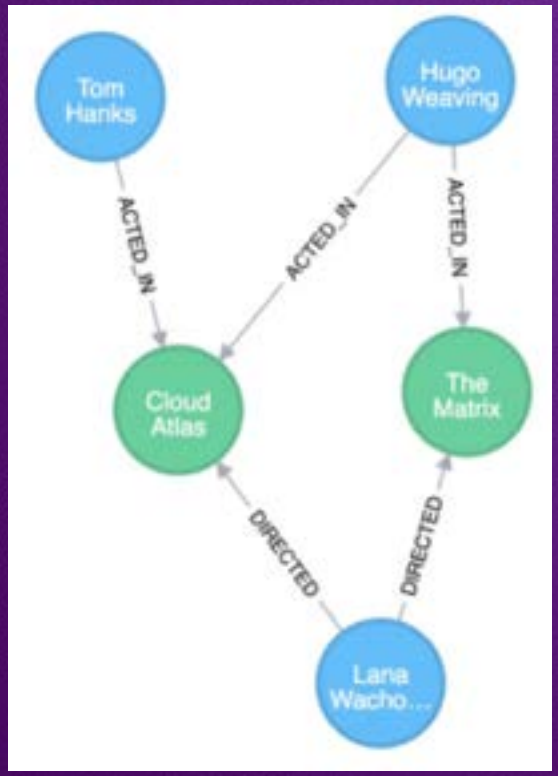
MANCHESTER 1824
The University of Manchester

Properties of graph DBs

For a developer, attractive properties of a graph DB are

Intuitiveness
Speed
Agility

Speed in development and execution



…create data…

```
CREATE (a:Person { name:"Tom Hanks",
  born:1956 })-[r:ACTED_IN { roles: ["Forrest"]}]->(m:Movie { title:"Forrest Gump",released:1994 })
CREATE (d:Person { name:"Robert Zemeckis", born:1951 })-[:DIRECTED]->(m)
RETURN a,d,r,m
```

…find interesting connections…

```
MATCH (p:Person { name:"Tom Hanks" })-[r:ACTED_IN]->(m:Movie)
RETURN m.title, r.roles
```

https://neo4j.com/developer/guide-data-modeling/

MANCHESTER 1824
The University of Manchester

Properties of graph DBs

For a developer, attractive properties of a graph DB are

Intuitiveness
Speed
Agility

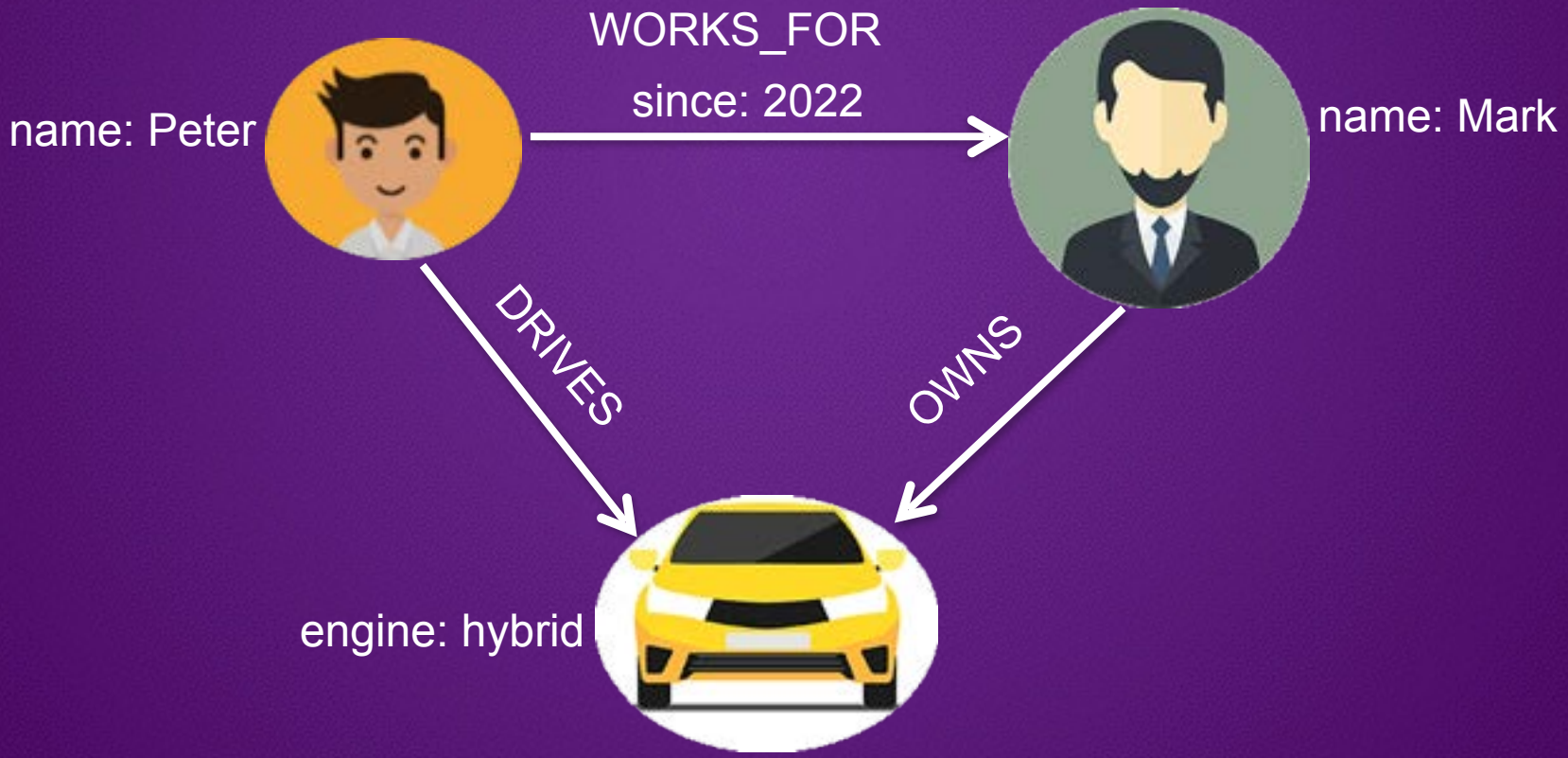
Agility = A Naturally Adaptive Model +
A Query Language Design for Connectedness

How easily and quickly your code adapt of the changing business.

MANCHESTER 1824
The University of Manchester

Detailed Property Graph

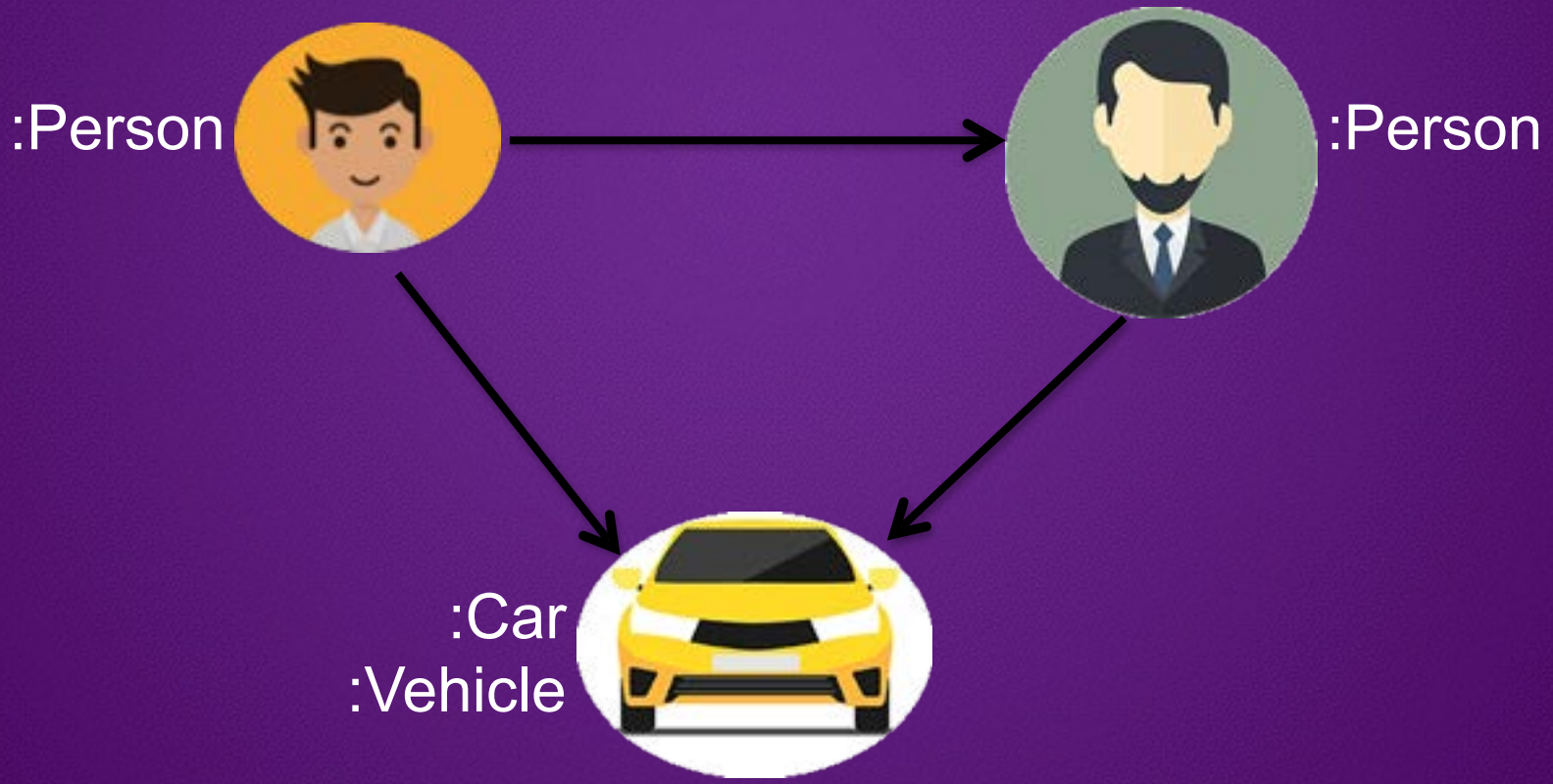
Let's start with a Detailed Property Graph



MANCHESTER 1824
The University of Manchester

Detailed Property Graph

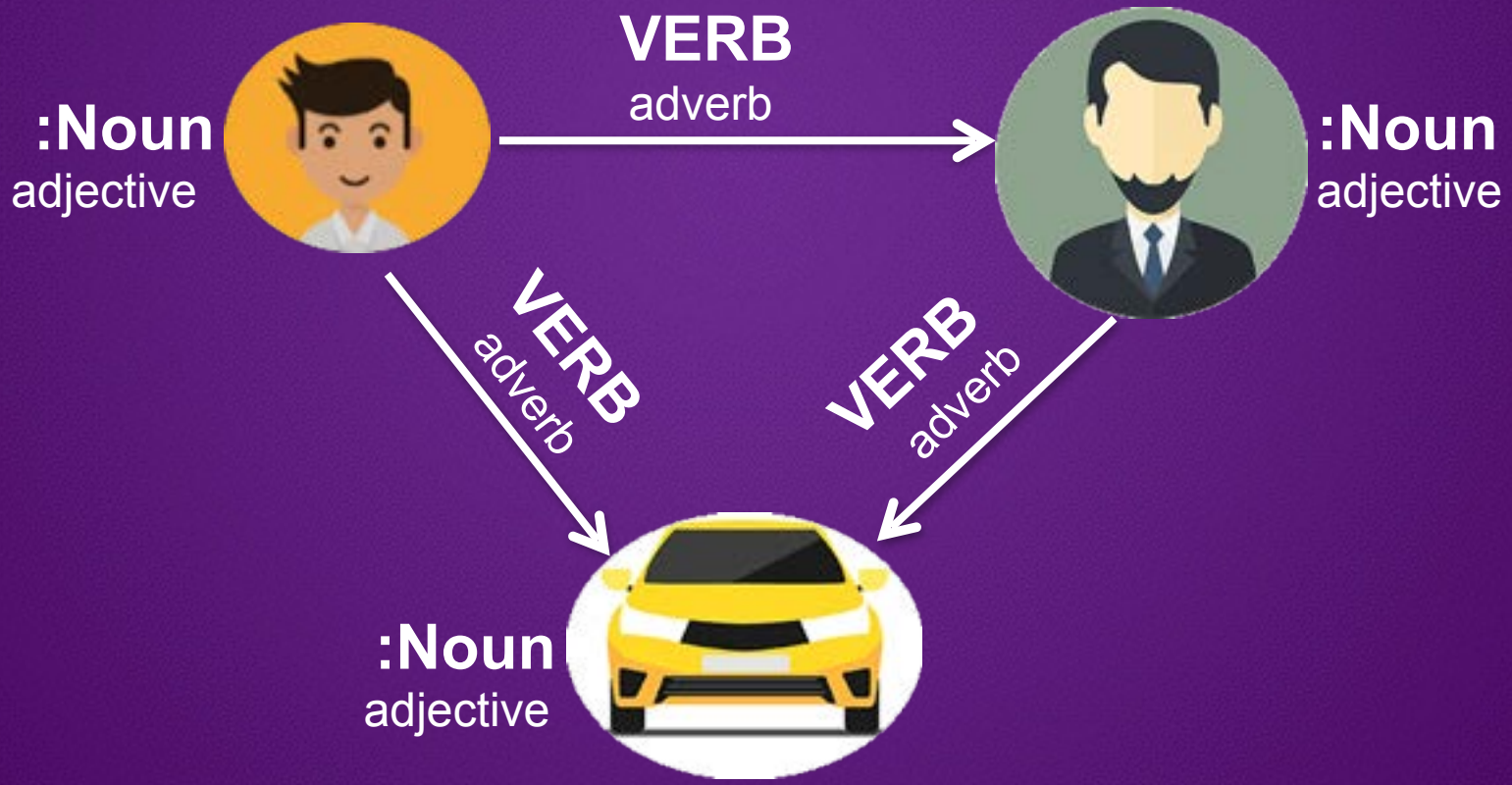
A Detailed Property Graph
is also called a Labeled Property Graph



MANCHESTER 1824
The University of Manchester

Mapping to Languages

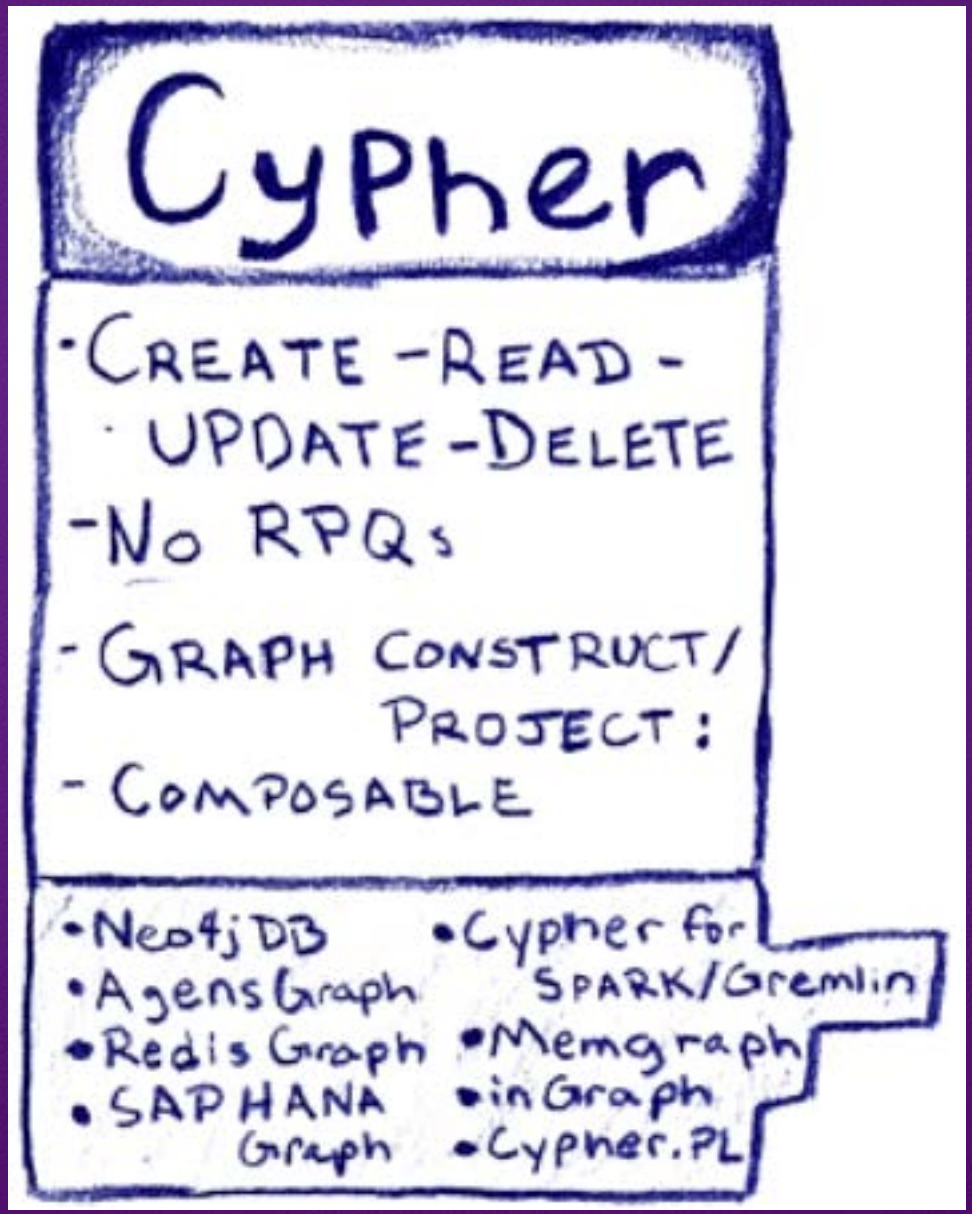
How to map a Detailed Property Graph to the English language?



20s

MANCHESTER 1824
The University of Manchester

Declarative query language



MANCHESTER 1824
The University of Manchester

Cypher versus SQL

Typical Complex SQL Join

```
(SELECT T.directReportees AS directReportees, sum(T.count) AS count
FROM (
SELECT manager.pid AS directReportees, 0 AS count
  FROM person_reportee manager
  WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
UNION
  SELECT manager.pid AS directReportees, count(manager.directly_manages) AS count
FROM person_reportee manager
WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
GROUP BY directReportees
UNION
SELECT manager.pid AS directReportees, count(reportee.directly_manages) AS count
FROM person_reportee manager
JOIN person_reportee reportee
ON manager.directly_manages = reportee.pid
WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
GROUP BY directReportees
UNION
SELECT manager.pid AS directReportees, count(L2Reportees.directly_manages) AS count
FROM person_reportee manager
JOIN person_reportee L1Reportees
ON manager.directly_manages = L1Reportees.pid
JOIN person_reportee L2Reportees
ON L1Reportees.directly_manages = L2Reportees.pid
WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
GROUP BY directReportees
) AS T
GROUP BY directReportees)
UNION
(SELECT T.directReportees AS directReportees, sum(T.count) AS count
FROM (
SELECT manager.directly_manages AS directReportees, 0 AS count
FROM person_reportee manager
WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
UNION
SELECT reportee.pid AS directReportees, count(reportee.directly_manages) AS count
FROM person_reportee manager
JOIN person_reportee reportee
ON manager.directly_manages = reportee.pid
WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
GROUP BY directReportees
UNION
SELECT depth1Reportees.pid AS directReportees,
count(depth2Reportees.directly_manages) AS count
FROM person_reportee manager
JOIN person_reportee L1Reportees
ON manager.directly_manages = L1Reportees.pid
JOIN person_reportee L2Reportees
ON L1Reportees.directly_manages = L2Reportees.pid
WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
GROUP BY directReportees
) AS T
GROUP BY directReportees)
UNION
(SELECT T.directReportees AS directReportees, sum(T.count) AS count
  FROM(
  SELECT reportee.directly_manages AS directReportees, 0 AS count
FROM person_reportee manager
JOIN person_reportee reportee
ON manager.directly_manages = reportee.pid
WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
GROUP BY directReportees
UNION
SELECT L2Reportees.pid AS directReportees, count(L2Reportees.directly_manages)
AS count
FROM person_reportee manager
JOIN person_reportee L1Reportees
ON manager.directly_manages = L1Reportees.pid
JOIN person_reportee L2Reportees
ON L1Reportees.directly_manages = L2Reportees.pid
WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
GROUP BY directReportees
) AS T
GROUP BY directReportees)
UNION
(SELECT L2Reportees.directly_manages AS directReportees, 0 AS count
FROM person_reportee manager
JOIN person_reportee L1Reportees
ON manager.directly_manages = L1Reportees.pid
JOIN person_reportee L2Reportees
ON L1Reportees.directly_manages = L2Reportees.pid
WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
)
```

The same query using Cypher

```
MATCH (boss)-[:MANAGES*0..3]->(sub),
      (sub)-[:MANAGES*1..3]->(report)
WHERE boss.name = "John Doe"
RETURN sub.name AS Subordinate,
  count(report) AS Total
```

22s

MANCHESTER 1824
The University of Manchester

Cypher: declarative query language

Cypher is designed to be easily read and understood by developers.

Cypher in action: creating data

```
CREATE (a:Person { name:"Tom Hanks",
  born:1956 })-[r:ACTED_IN { roles: ["Forrest"]}]->(m:Movie { title:"Forrest Gump",released:1994 })
CREATE (d:Person { name:"Robert Zemeckis", born:1951 })-[:DIRECTED]->(m)
RETURN a,d,r,m
```

Cypher in action: finding interesting connections

```
MATCH (p:Person { name:"Tom Hanks" })-[r:ACTED_IN]->(m:Movie)
RETURN m.title, r.roles
```

MANCHESTER 1824
The University of Manchester

Cypher for creating data

ASCII art representation in Cypher


```
CREATE (:Person {name:"Peter"} - [:WORKS_FOR] -> (:Person {name:"Mark"})
```

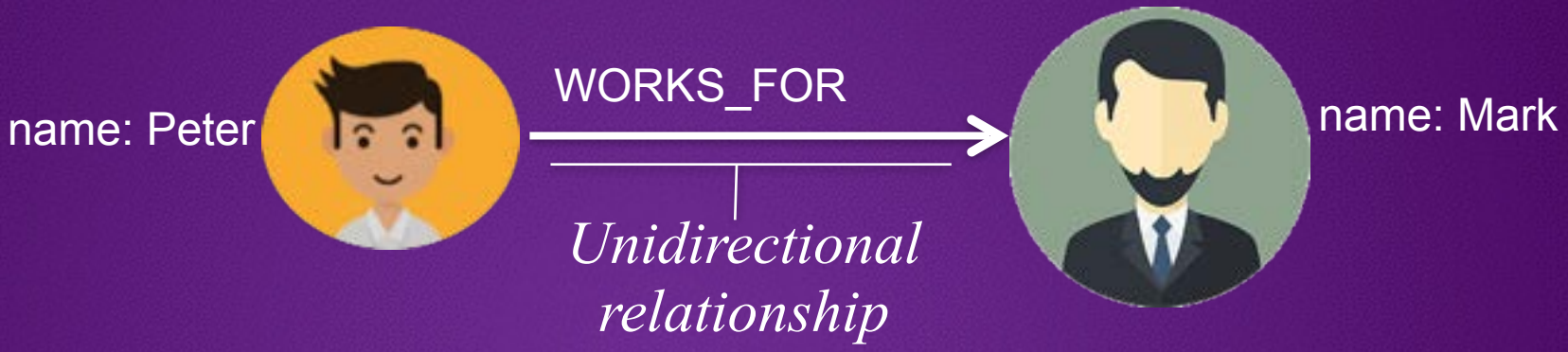
label property

Drawing the pattern of the graph in your query through ASCII characters ->

label property

MANCHESTER 1824
The University of Manchester

Cypher for representing Bi-Directionality




```
MATCH (:Person {name:"Peter"} - [:WORKS_FOR] -> (p:Person)
```

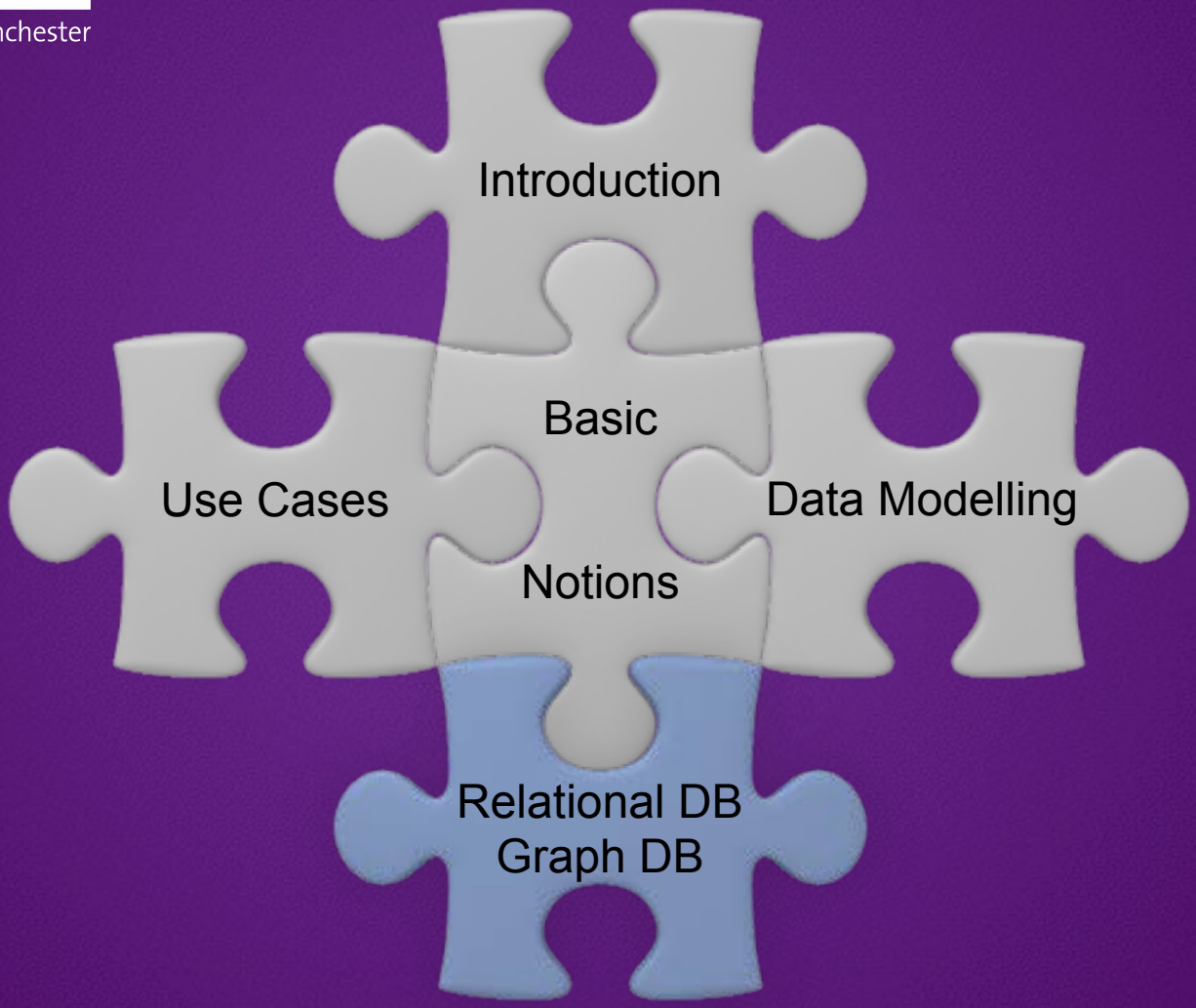
label property *Unidirectional relationship* label

```
MATCH (:Person {name:"Peter"} - [:WORKS_WITH] - (p:Person)
```

label property *Bidirectional relationship* label

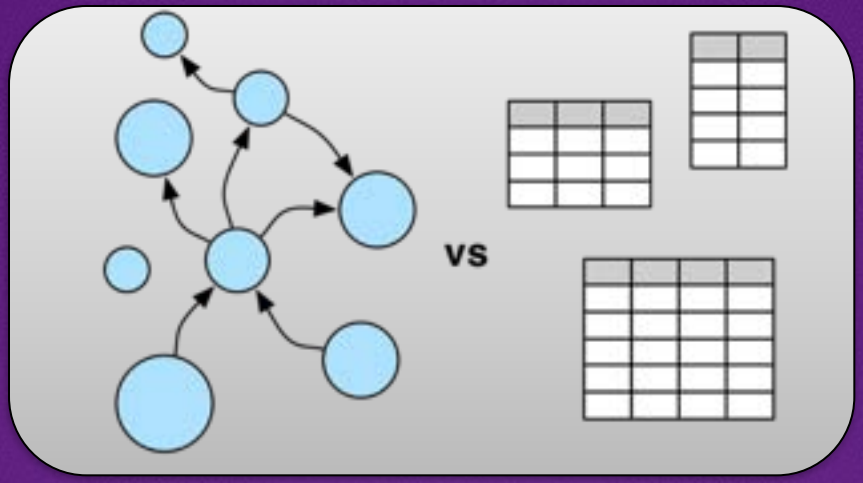
25s

MANCHESTER 1824
The University of Manchester



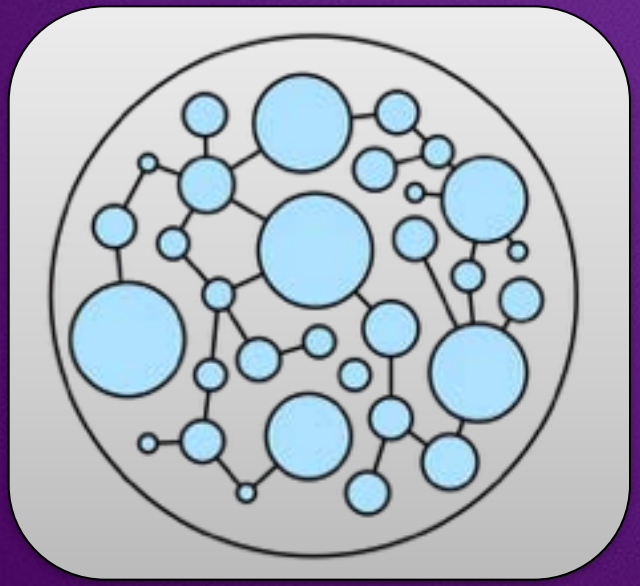
MANCHESTER 1824
The University of Manchester

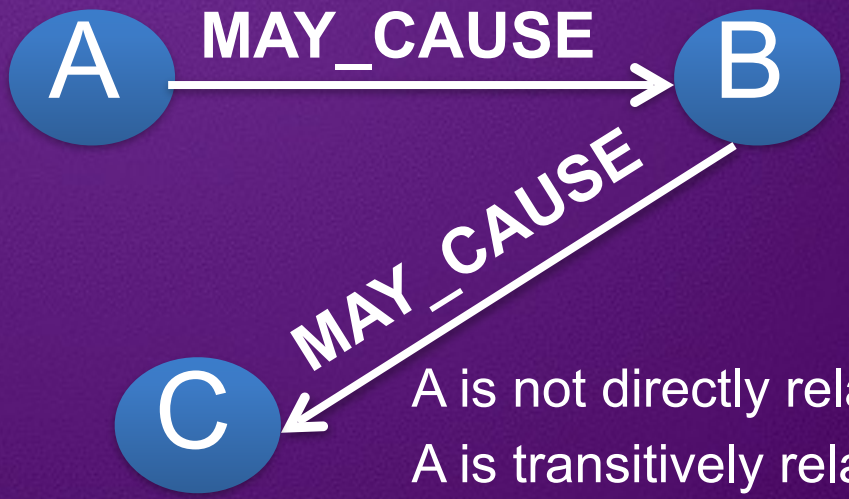
Relational DB vs Graph DB



Tabular data may be best stored in a relational DB

Interconnected data





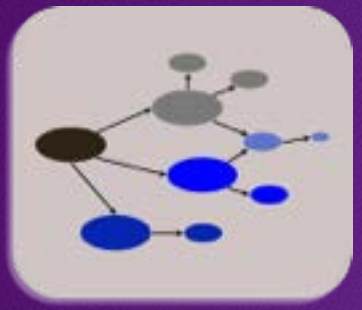
A is not directly related to C

A is transitively related to C

MANCHESTER 1824
The University of Manchester

How to use Neo4j

There are 3 basic steps to using Neo4j.



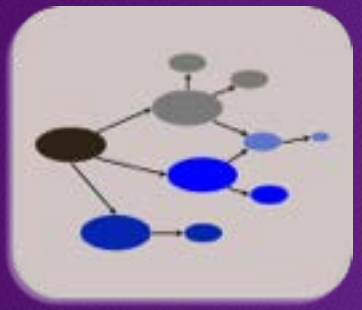
Step 1: creating a model

Creating in advance
labels; nodes; relationships; and properties

MANCHESTER 1824
The University of Manchester

How to use Neo4j

There are 3 basic steps to using Neo4j.



Step 1: creating a model

Creating in advance
labels; nodes; relationships; and properties

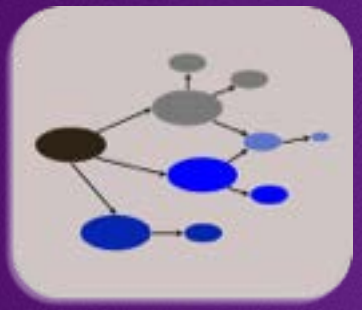
Step 2: loading data

The easiest way to load data from a relational DB in Neo4j is by exporting the data as a CSV file

MANCHESTER 1824
The University of Manchester

How to use Neo4j

There are 3 basic steps to using Neo4j.



Step 1: creating a model

Creating in advance
labels; nodes; relationships; and properties

Step 2: loading data

The easiest way to load data from a relational DB in Neo4j is by exporting the data as a CSV file

Step 3: querying data

Neo4j has a built-in web application for querying data, so you can explore your data

30s

MANCHESTER 1824
The University of Manchester

Moving from Relational DB to Graph DB

Not all applications are the same!

Depending on data interconnections, you may consider

Option 1: migrate all data

Option 2: migrate a subset of data

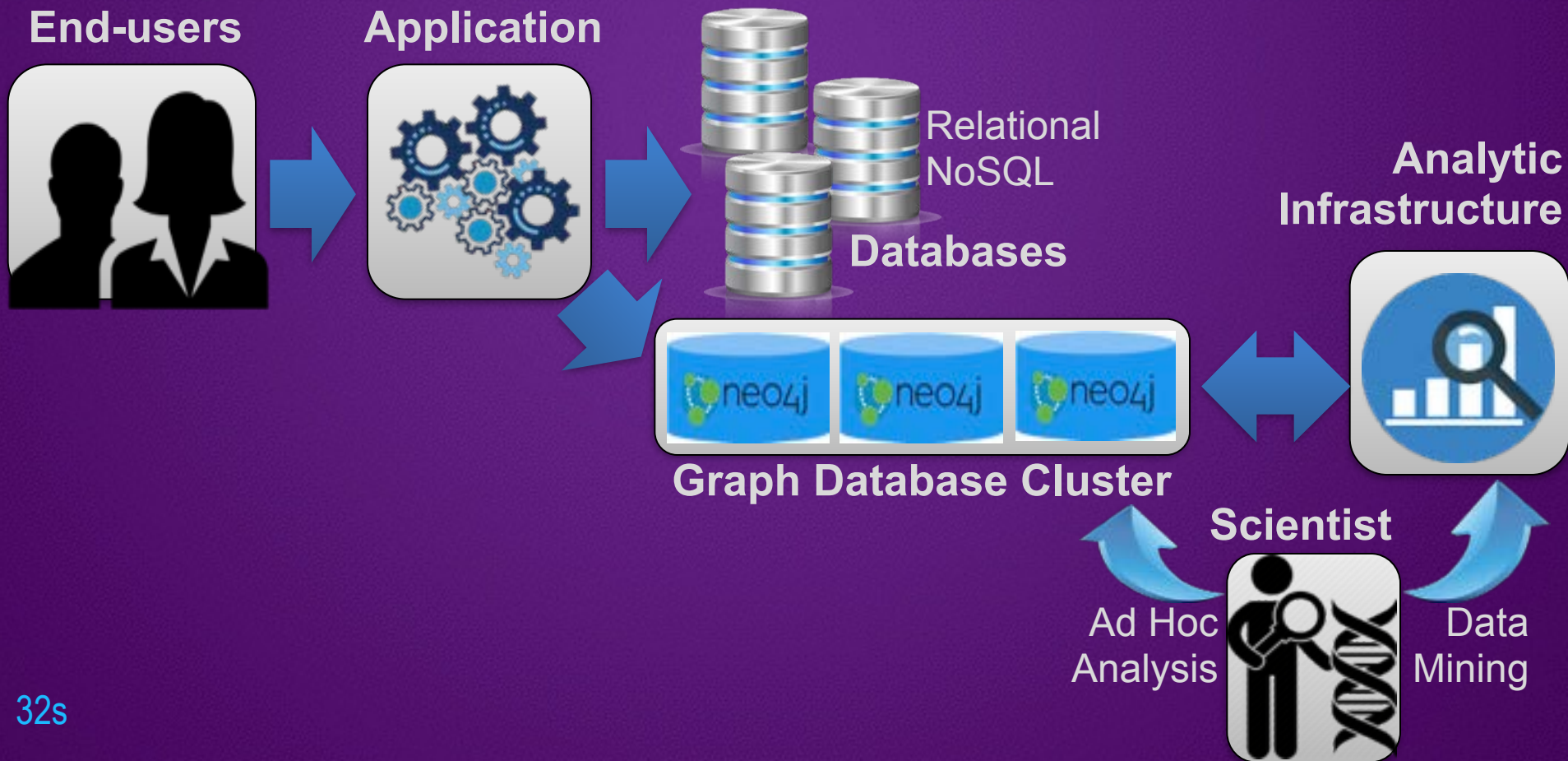
Option 3: duplicate a subset of data

MANCHESTER 1824
The University of Manchester

Moving from Relational DB to Graph DB

Option 2: migrate a subset of data

Architectural overview



MANCHESTER 1824
The University of Manchester

Relational DB vs Graph DB

PersonID	Person [Name]	Nationality [Country]	UK University	University location [UK country]
1024	*Luca*	*Italy*	*Queen's University Belfast*	*Northern Ireland*
12	*Lisa*	*UK*	*University College London*	*England*
2048	*Basil*	*UK*	*The University of Manchester*	*England*
24	*Alice*	*USA*	*Cardiff University*	*Wales*
212	*Maria*	*Spain*	*The University of Edinburgh*	*Scotland*

33s

Relational DB vs Graph DB

MANCHESTER 1824
The University of Manchester

Name	Country	University	UK country
Luca	*84*	*32*	*1*
Lisa	*186*	*3*	*2*
Basil	*186*	*6*	*2*
Alice	*187*	*25*	*3*
Maria	*165*	*4*	*4*

ID	Country name
84	*Italy*
186	*UK*
187	*USA*
165	*Spain*

ID	University name
32	*Queen's University Belfast*
3	*University College London*
6	*The University of*
25	*Cardiff University*
4	*The University of Edinburgh*

ID	UK Country name
1	*Northern Ireland*
2	*England*
3	*Wales*
4	*Scotland*

MANCHESTER 1824
The University of Manchester

Relational DB vs Graph DB

Name	Country	University	UK country
Luca	*84*	*32*	*1*
Lisa	*186*	*3*	*2*
Basil	*186*	*6*	*2*
Alice	*187*	*25*	*3*
Maria	*165*	*4*	*4*

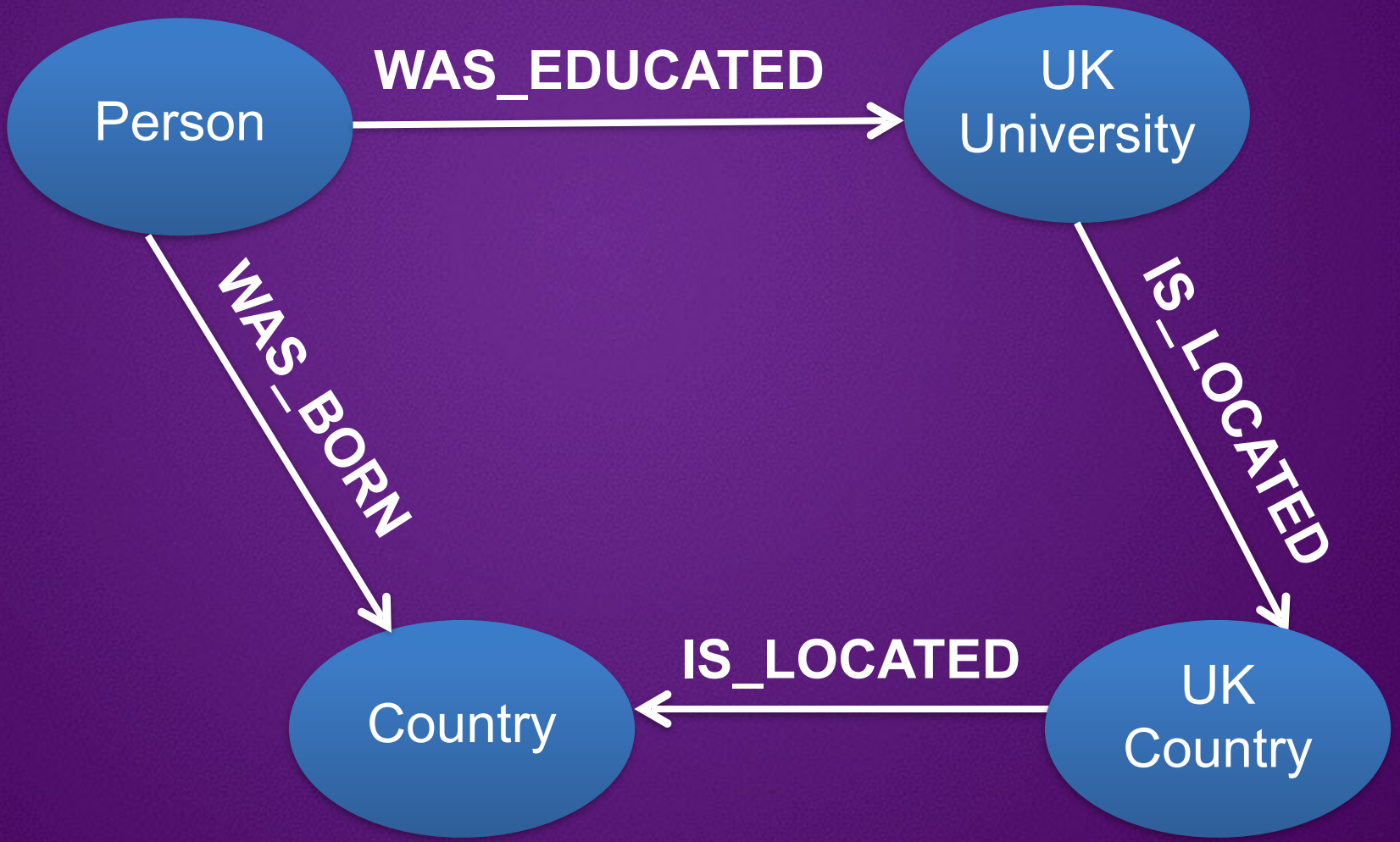
```
1 SELECT
2       p.name, c.name as country, u.name as university
3 FROM
4       person p
5       LEFT JOIN country c ON c.ID = p.country
6       LEFT JOIN university u ON p.university = u.ID
7 WHERE
8       u.name = 'The University of Manchester'
```

name	country	university
Basil	UK	The University of Manchester

35s

MANCHESTER 1824
The University of Manchester

Relational DB vs Graph DB



MANCHESTER 1824
The University of Manchester

Cypher: declarative query language

```
MATCH
(p1)-[r1:WAS_EDUCATED]-(u),
(p2)-[r2:WAS_BORN]→(c)
WHERE u.name = 'The University of Manchester'
RETURN p1.name as name, c.name as country, u.name as university
```

name	country	university
"Basil"	"UK"	"The University of Manchester"

37s

MANCHESTER 1824
The University of Manchester

Cypher: declarative query language

- Performance
- Write queries
- Maintenance
- Model and store relationships

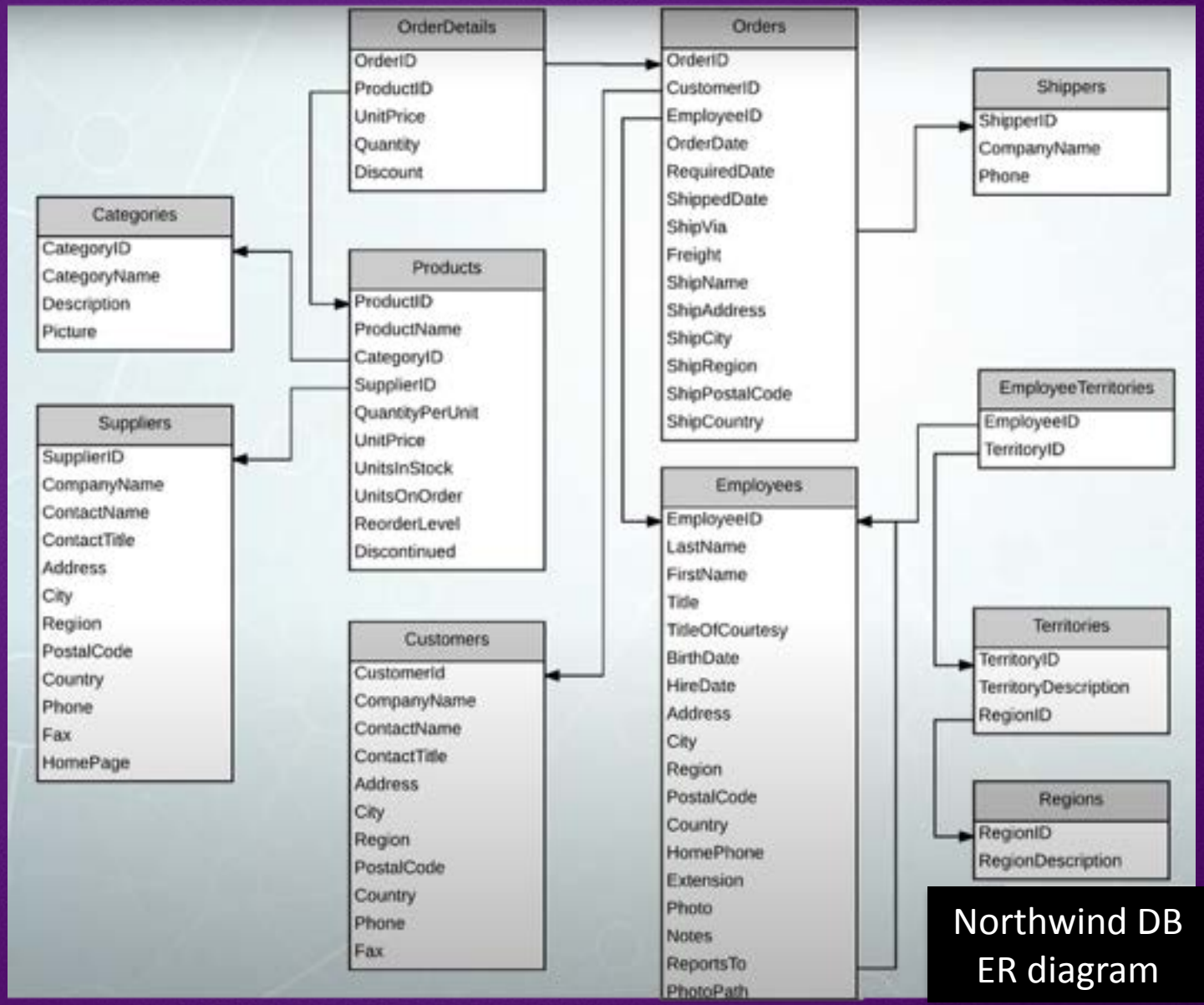
Converting Northwind DB to Graph DB

MANCHESTER 1824
The University of Manchester

Let's see if I understand this ER diagram…

…primary keys…

foreign keys…



Northwind DB ER diagram

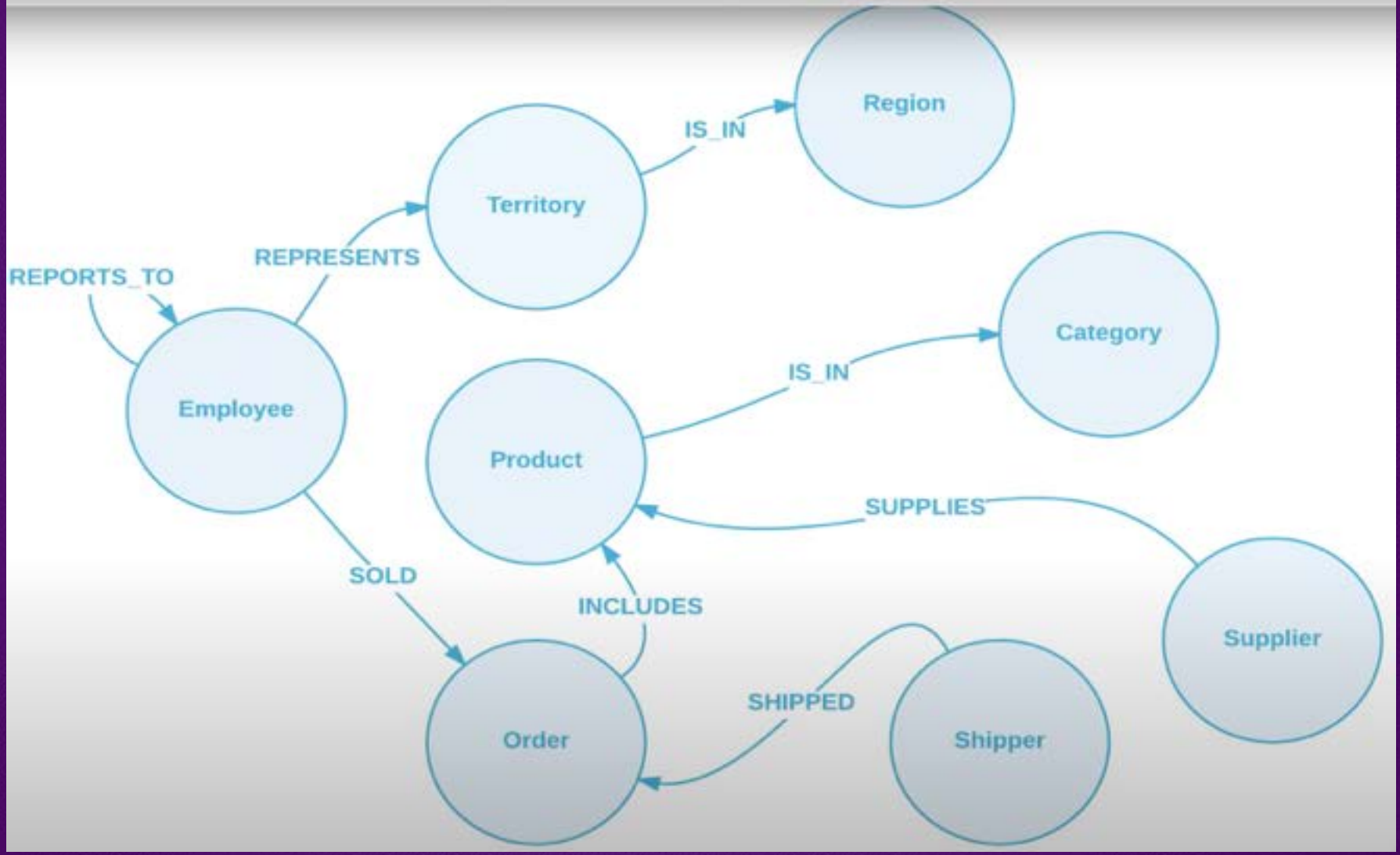
39s

https://www.youtube.com/watch?v=NO3C-CWykkY&list=PL9Hl4pk2FsvWM9GWaguRhlCQ-pa-ERd4U&index=6

MANCHESTER 1824
The University of Manchester

Converting Northwind DB to Graph DB

The graph model for Northwind dataset



40s

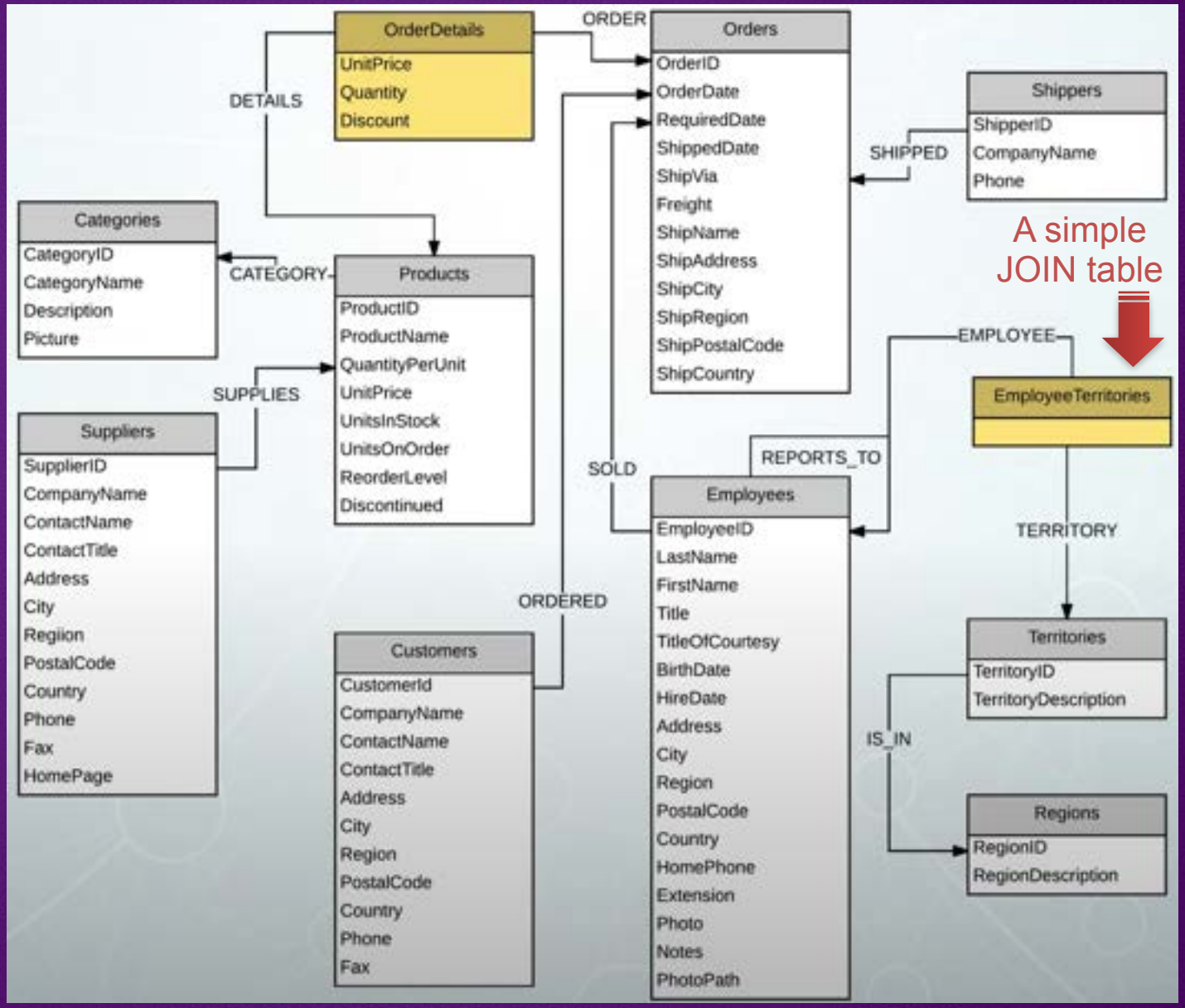
https://www.youtube.com/watch?v=NO3C-CWykkY&list=PL9HI4pk2FsvWM9GWaguRhICQ-pa-ERd4U&index=6

Converting Northwind DB to Graph DB

MANCHESTER 1824
The University of Manchester

Step 1: locate foreign keys, delete them, and replace them with relationships

Step 2: locate JOIN tables, and ….



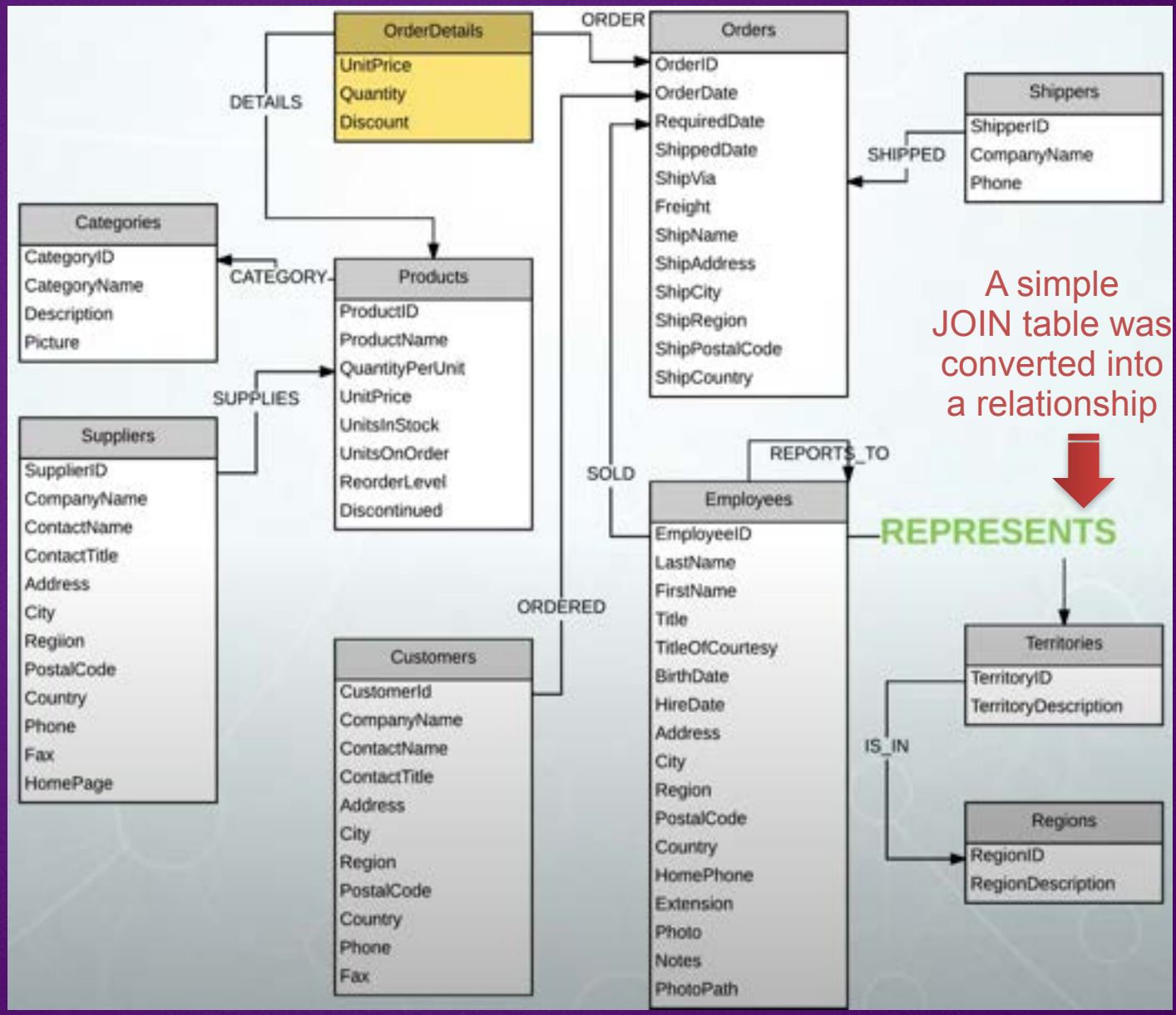
https://www.youtube.com/watch?v=NO3C-CWykkY&list=PL9Hl4pk2FsvWM9GWaguRhlCQ-pa-ERd4U&index=6

Converting Northwind DB to Graph DB

MANCHESTER 1824
The University of Manchester

Step 1: locate foreign keys, delete them, and replace them with relationships

Step 2: locate JOIN tables, and ….



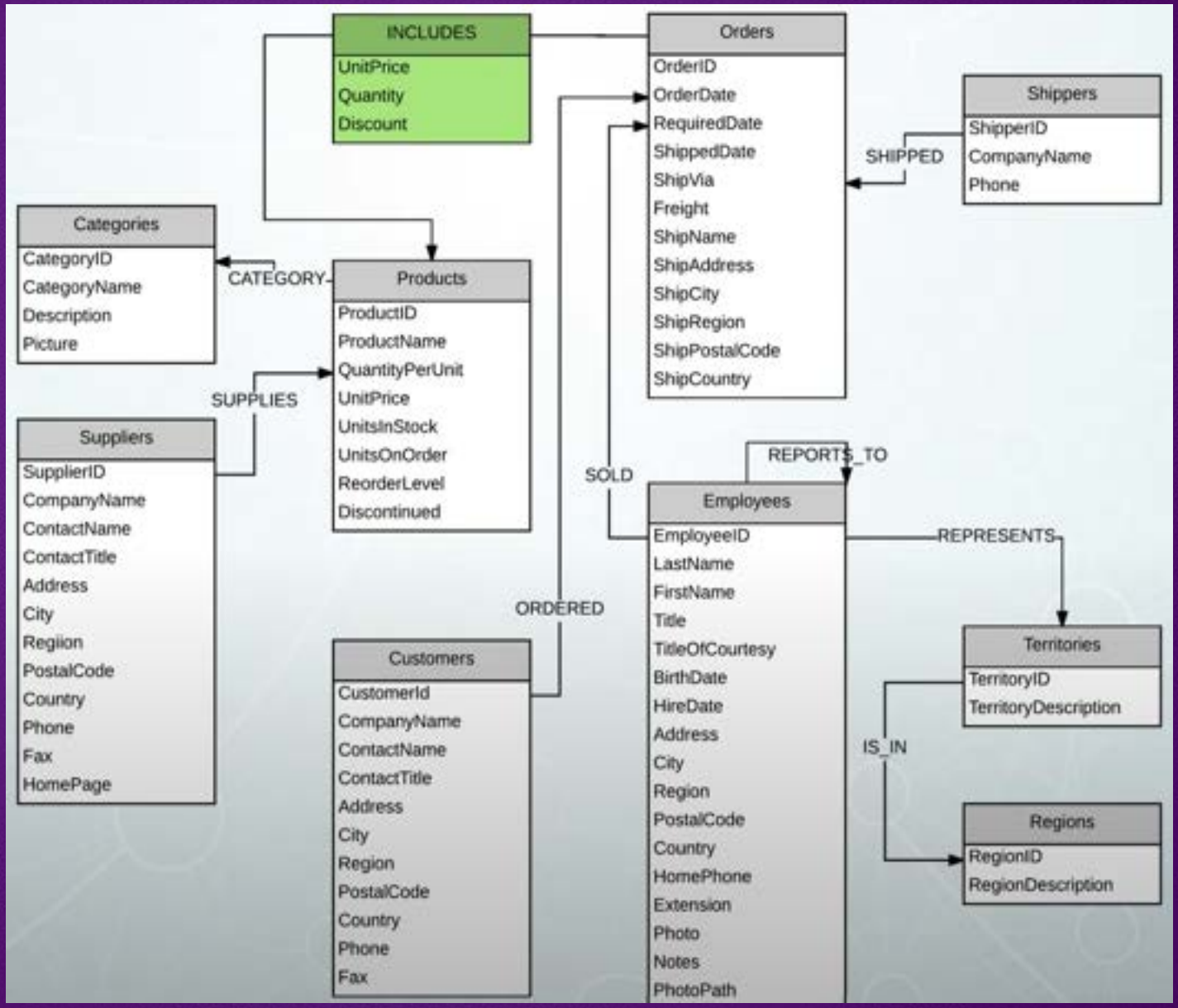
https://www.youtube.com/watch?v=NO3C-CWykkY&list=PL9Hl4pk2FsvWM9GWaguRhlCQ-pa-ERd4U&index=6

Converting Northwind DB to Graph DB

MANCHESTER 1824
The University of Manchester

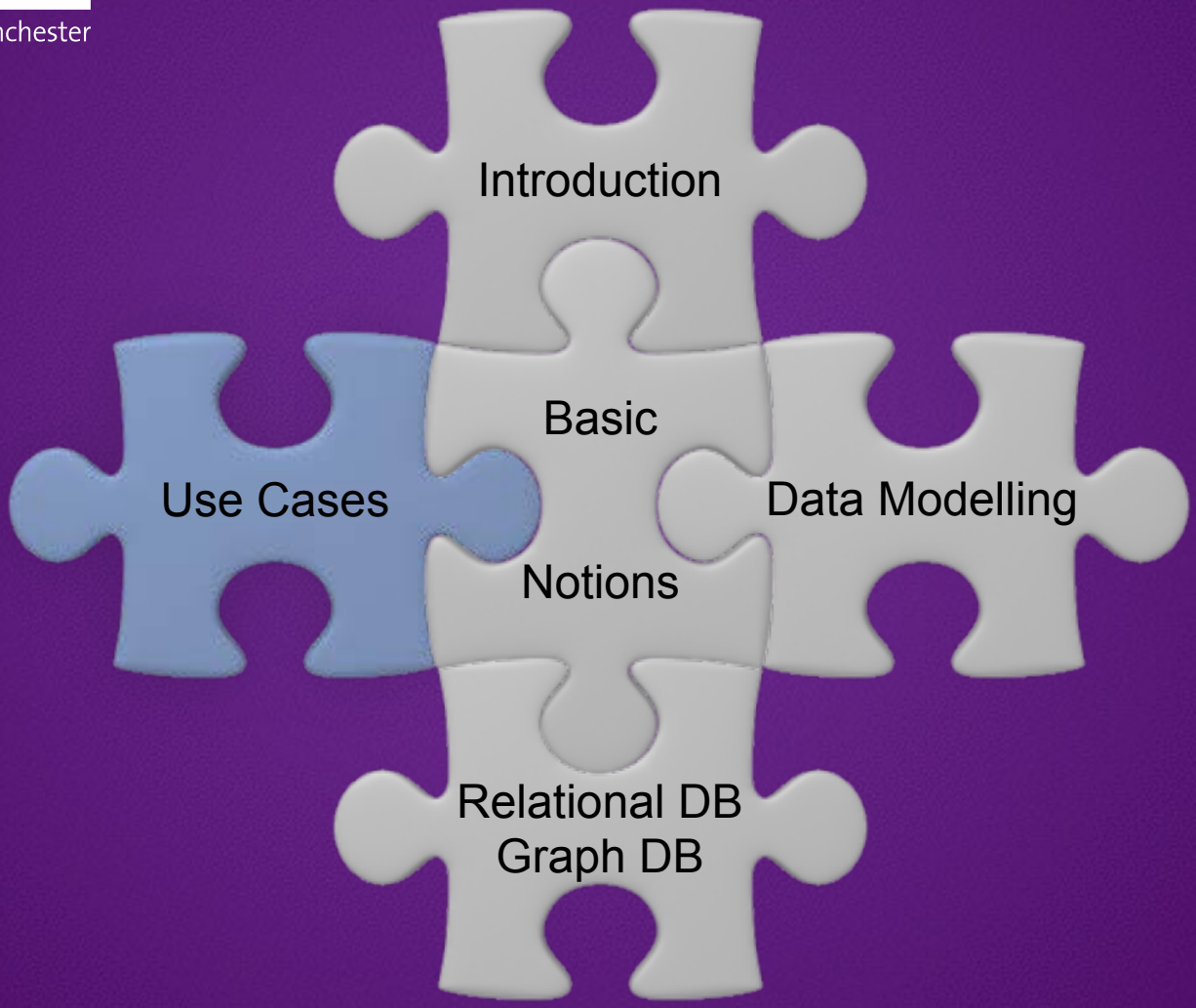
Step 1: locate foreign keys, delete them, and replace them with relationships

Step 2: locate JOIN tables, and convert them into relationships



https://www.youtube.com/watch?v=NO3C-CWykkY&list=PL9Hl4pk2FsvWM9GWaguRhlCQ-pa-ERd4U&index=6

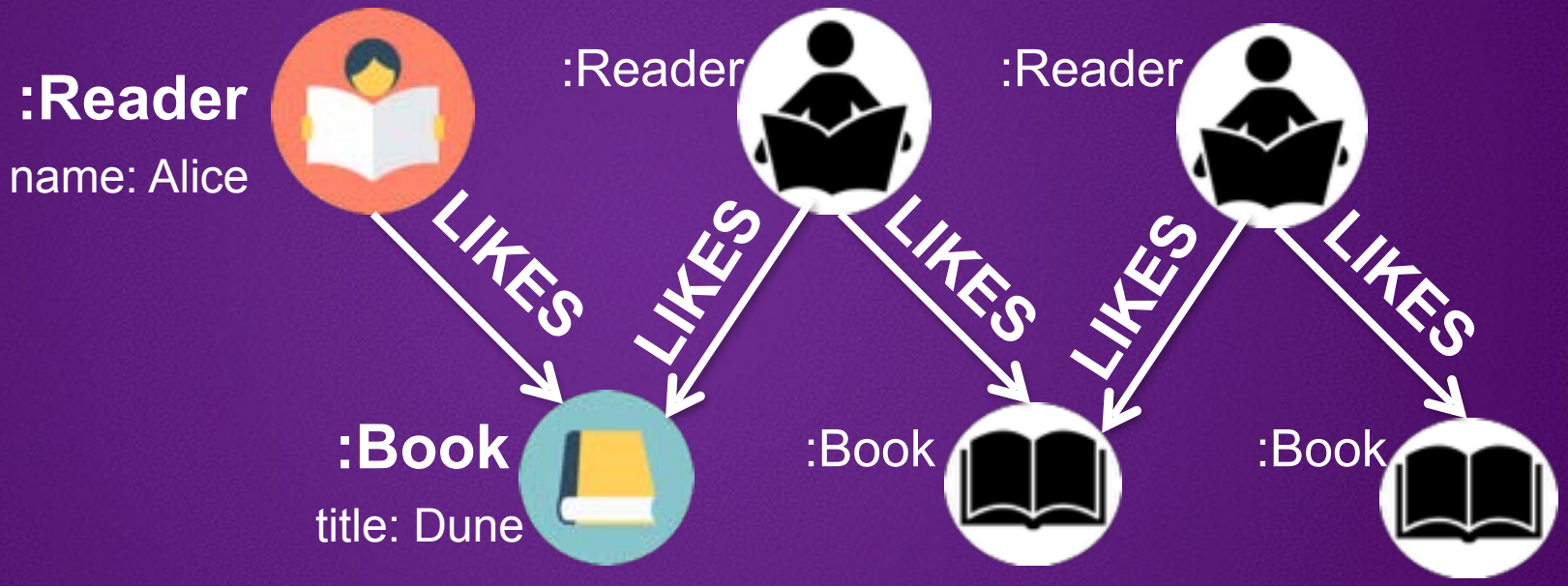
MANCHESTER 1824
The University of Manchester



MANCHESTER 1824
The University of Manchester

Book review data model

Simple data model



… finding books for Alice according to other readers …

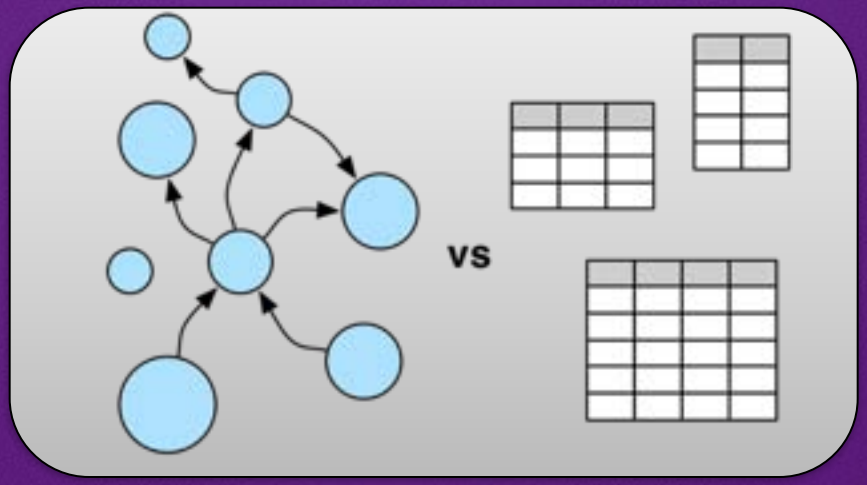
```
MATCH (:Reader {name:'Alice'})-[:LIKES]->(:Book {title:'Dune'})
      <-[:LIKES]-(:Reader)-[:LIKES]->(books:Book)
RETURN books.title
```

2015, Graph Databases by Ian Robinson, Jim Webber, and Emil Eifrem

MANCHESTER 1824
The University of Manchester

Common graph technology use cases: Fraud Detection

Banks and insurance companies lose billions of dollars every year to fraud.



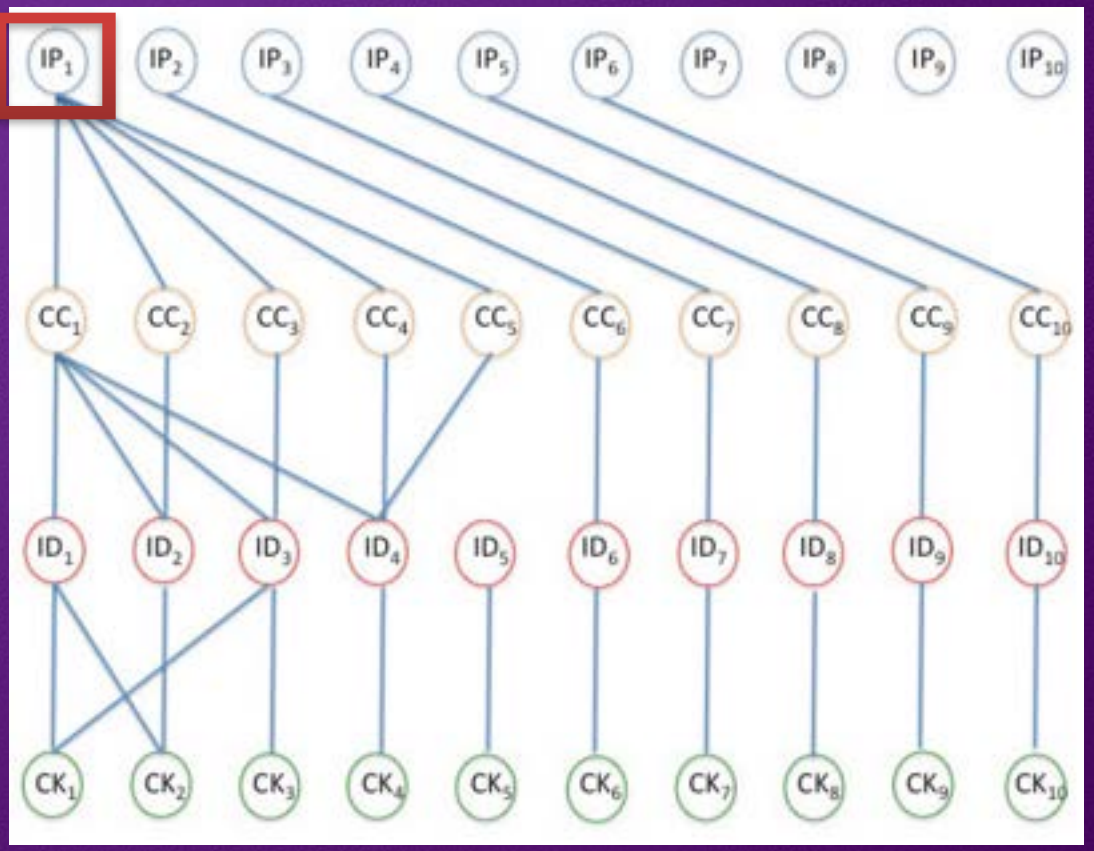
Graph databases offer new methods of uncovering fraud by looking at the connections that link individual data points

MANCHESTER 1824
The University of Manchester

Common graph technology use cases: Fraud Detection

Example of fraud-detection with link analysis

Look for multiple transactions from the same IP with different credit cards.

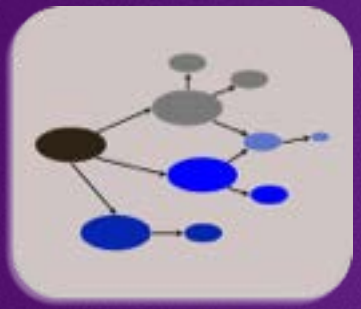


https://go.neo4j.com/rs/710-RRC-335/images/Neo4j_Top5_UseCases_Graph%20Databases.pdf

MANCHESTER 1824
The University of Manchester

Google: Knowledge Graph entities

Google Knowledge Graph has millions of entries that describe real-world entities, such as people and places.



Some types of entities found in the **Google Knowledge Graph**

Book
BookSeries
EducationalOrganization
Event
GovernmentOrganization
LocalBusiness
Movie
MovieSeries
MusicAlbum
MusicGroup
MusicRecording
Organization
Periodical
Person
Place
SportsTeam
TVEpisode
TVSeries
VideoGame
VideoGameSeries
WebSite

MANCHESTER 1824
The University of Manchester

Google: Knowledge Graph entities

Movie is a type of entity in the **Google Knowledge Graph**

Movie

A Schema.org Type

Thing > CreativeWork > Movie

A movie. [more...]

Property	Expected Type	Description
Properties from Movie		
actor	Person	An actor, e.g. in TV, radio, movie, video games etc., or in an event. Actors can be associated with individual items or with a series, episode, clip. Supersedes actors.
countryOfOrigin	Country	The country of origin of something, including products as well as creative works such as movie and TV content. In the case of TV and movie, this would be the country of the principle offices of the production company or individual responsible for the movie. For other kinds of CreativeWork it is difficult to provide fully general guidance, and properties such as contentLocation and locationCreated may be more applicable. In the case of products, the country of origin of the product. The exact interpretation of this may vary by context and product type, and cannot be fully enumerated here.
director	Person	A director of e.g. TV, radio, movie, video gaming etc. content, or of an event. Directors can be associated with individual items or with a series, episode, clip. Supersedes directors.
duration	Duration	The duration of the item (movie, audio recording, event, etc.) in ISO 8601 date format.
musicBy	MusicGroup or Person	The composer of the soundtrack.

https://schema.org/Movie

MANCHESTER 1824
The University of Manchester

Common graph technology use cases: Knowledge Graph

Organisations are using knowledge graphs to improve the reasoning skills

Knowledge graphs are a type of graph.

Knowledge graphs can reason (e.g. using a description logic reasoner) about the underlying data.

A key differentiator:
Resource Description Framework (RDF)

SPARQL is a RDF query language

Neo4j supports SPARQL

Amazon Neptune is compatible with SPARQL 1.1.

50s

MANCHESTER 1824
The University of Manchester

Common graph technology use cases: Knowledge Graph

The results of SPARQL queries can be RDF graphs.

An RDF graph is a set of RDF triples.

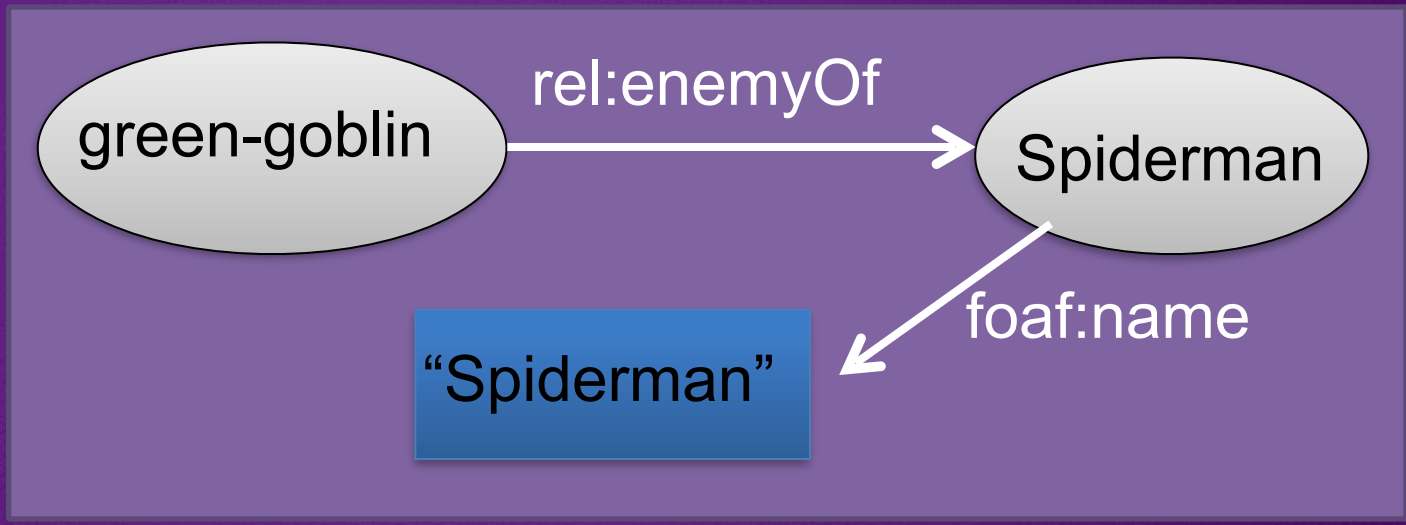
An RDF triple

MANCHESTER 1824
The University of Manchester

Common graph technology use cases: Knowledge Graph

Example of an RDF Graph

https://www.w3.org/TR/turtle/



RDF Graph uses
Universal Resource Identifiers (URIs)

```
<http://example.org/#green-goblin>
<http://www.perceive.net/schemas/relationship/enemyOf>
<http://example.org/#spiderman> .

<http://example.org/#spiderman>
 <http://xmlns.com/foaf/0.1/name>
 "Spiderman" .
```

52s

MANCHESTER 1824
The University of Manchester

Using Amazon Neptune to build an Enterprise Knowledge Graph

https://aws.amazon.com/neptune/knowledge-graphs-on-aws/

amazon alexa

"Amazon Neptune is a key part of the toolkit we use to continually expand ***Alexa's knowledge graph*** *for our tens of millions of Alexa customers"*

David Hardcastle,
Director of Amazon Alexa.

Why do you need a knowledge graph?

Insights with Machine Learning: use machine learning with knowledge graphs for better decision making and knowledge discovery.

Build virtual assistants, chatbots or question-answering systems: build context-aware systems that can derive at an answer based on queries and a vast knowledge base.

53s

MANCHESTER 1824
The University of Manchester

Images shown may have a copyright

Questions

54s