International Journal of Business Analytics and Intelligence
7 (1) 2019, 03-07
http://publishingindia.com/ijbai/

Submitted: 15 December, 2018
Accepted: 21 January, 2019

Ontology, Graphs and Data Science

Favio Vázquez*

Abstract

The present and future of data science depends on us being alert of the trends and advances in the scientific and technological community. In this article the fundamentals of ontology and graphs are introduce with their relations with data science, also presenting the concepts of knowledge-graphs and data fabric.

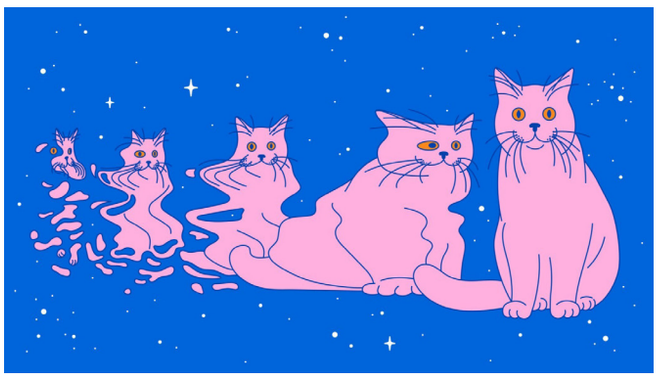
Ontology

If you are new to the word ontology don't worry, I'm going to give a primer on what it is, and then why it matters for the data world. I'll be explicit in the difference between philosophical ontology and the ontology related to information and data in computer science.

Ontology (the philosophical part)

In simple words, one can say that ontology is the study of what there is. But there is another part to that definition that will help us in the following sections, and that is ontology is usually also taken to encompass problems about the most general features and relations of the entities which do exist.

Ontology open new doors for what there is too. Let me give you an example.



Quantum mechanics opened a new view of reality and what "exists" in nature. For some physicists in the 1900s there was simply no reality expressed in the quantum formalism. At the other extreme, there were many quantum physicists who took the diametrically opposite view: that the unitarily evolving quantum state completely describes actual reality, with the alarming implication that practically all quantum alternatives must always continue to coexist (in superposition). And thus opening the whole world to a new view and understanding of the "things" that "exists" in nature.

But let's come back to the relation of entities part of the definition. Sometimes when we talk about entities and their relation, ontology is referred as formal ontology. These are theories that attempt to give precise mathematical formulations of the properties and relations of certain entities. Such theories usually propose axioms about these entities in question, spelled out in some formal language based on some system of formal logic.

And this will allow us to do a quantum jump to next part of the article.

Ontology (the information and computational part)

If we bring back the definition of formal ontology from above, and then we think of data and information, it's possible to set up a framework to study data and its relation to other data. In this framework we represent information in an especially useful way. Information represented in a particular formal ontology can be more easily accessible to automated information processing, and how best to do this is an active area of research in computer science like data science. The use of the formal ontology here is representational. It is a framework to represent information, and as such it can be representationally successful whether or not the formal theory used in fact truly describes a domain of entities.

* Data Science Course Instructor, Business Science University, Pennsylvania Area.

Now it's a good moment to see how ontology can help us in the data science world.

An important concept we need to introduce right now is the one of linked data. The goal of linked data is to publish structured data in such a way that it can be easily consumed and combined with other Linked Data. That allow us to talk about the concept of the knowledge graph which consists in integrated collections of data and information that also contains huge numbers of links between different data.

Ontology is important here because it's the way we can connect entities and understand their relationships. With ontology one can enable such a description, but first we need to formally specify components such as individuals (instances of objects), classes, attributes and relations as well as restrictions, rules and axioms.

Databases Modeling and Ontologies

Currently, most of the technologies that employ data modeling languages (like SQL) are designed using a rigid "Build the Model, then Use the Model" mindset.

For example, suppose you want to change a property in a relational database. You had previously thought that the property was single-valued, but now it needs to be multi-valued. For almost all modern relational databases, this change would require you to delete the entire column for that property and then create an entirely new table that holds all of those property values plus a foreign key reference.

This is not only a lot of work, but it will also invalidate any índices that deal with the original table. It will also invalidate any related queries that your users have written. In short, making that one change can be very difficult and complicated. Often, such changes are so troublesome that they are simply never made.

By contrast, all data modeling statements (along with everything else) in ontological languages for data are incremental, by their very nature. Enhancing or modifying a data model after the fact can be easily accomplished by modifying the concept.

Ontological languages, linked data and all of that exist in the realm of graph databases. So it's time to discuss some ideas and concepts of graph databases, what they are, what are their advantages and how they can help us in our daily tasks.

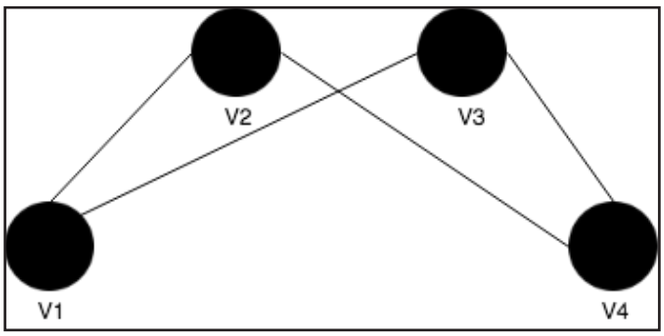
What is a Graph?

I'm going to give two definitions of a graph. First the mathematical one, and then a more simplistic one.

A graph G is a finite, non-empty set V together with a (possibly empty) set E (disjoint from V) of two-element subsets of (distinct) elements of V. Each element of V is referred to as a vertex and V itself as the vertex set of G; the members of the edge set E are called edges. By an element of a graph we shall mean a vertex or an edge.

One of the most appealing features of graph theory lies in the geometric or pictorial aspect of the subject. Given a graph, it is often useful to express it diagrammatically, where by each element of the set is represented by a point in the plane and each edge by a line segment.

It is convenient to refer to such a diagram of G as G itself, since the sets V and E are easily discernible. In the figure bellow, a graph G is shown with vertex set V = {V1, V2, V3, V4} and edge set E = {V1V2, V1V3, V2V4, V3V4}



As you can see the set V contains the number of vertex or points in the graph and E the relationships between them (read V1V2 like V1 is connected to V2).

So in simple words, a graph is a mathematical representation of objects (or entities or nodes) and their relationships (or edges). Each one of those points can represent different things depending on what you want. By the way, here nodes and vertices mean the same, we'll use them interchangeably.

What is a Database?

A database (DB), in the most general sense, is an organized collection of data. More specifically, a database is an electronic system that allows data to be easily accessed, manipulated and updated.

In other words, a database is used by an organization as a method of storing, managing and retrieving information. Modern databases are managed using a database management system (DBMS).

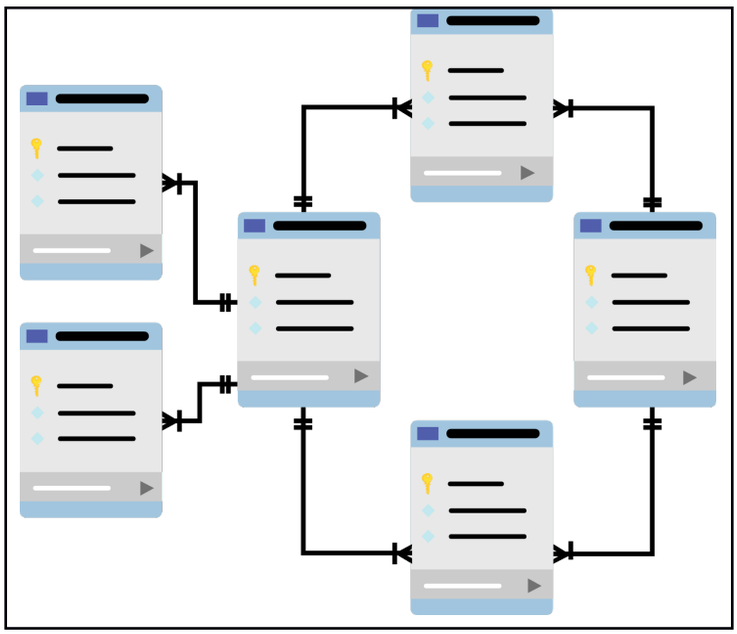
Do you want to know the truth? From my experience most databases are:

- Not organized
- Not easily accessed
- Not easily manipulated
- Not easily updated

When we talk about doing data science, in older years it was easier to maintain a database because the data was simple, smaller and slower. Nowadays we can save almost whatever we want in a "database", and that definition I think is stuck with another concept, the relational database.

In a relational database we have a set of "formally" described tables from which data can be accessed or reassembled in many different ways without having to reorganize the database tables. Basically we have schemas in where we can store different tables, and inside of those tables we have a set of columns and rows, and inside of an specific position (row and column) we have an observation.

We also have a relationship between those tables. But they're not the most important thing, the data they contain is the most important thing. Normally they are pictured like this:



What is a Graph Database?

Based upon the concept of a mathematical graph, a graph database contains a collection of nodes and edges. A node represents an object, and an edge represents the connection or relationship between two objects. Each node in a graph database is identified by a unique identifier that expresses key value pairs. Additionally, each edge is defined by a unique identifier that details a starting or ending node, along with a set of properties.

A graph database stores the same sort of data, but is also able to store linkages between the things. I don't have to run JOINs to understand how I should market to each individual customer. I can see the relationships in the data without having to make a hypothesis and test it.

Whereas relational databases store highly-structured data in tables with predetermined columns and rows, graph databases can map multiple types of relational and complex data. Thus, graph databases are not rigid in their

organization and structure, as relational databases are. All relationships are natively stored within the vertices of the edges, meaning that the vertices and edges can each have properties associated with them. This structure allows for a database that can depict complex relationships between unrelated data sets.

Did you know that 2018 was touted as "The Year of the Graph"?, as more and more organizations both large and small have recently begun to invest in graph database technology. So we aren't on a crazy path here.

I'm not saying that everything we know from relational databases, and SQL will not work anymore. I'm saying that there are some cases (surprisingly a lot of them) where you are better using a graph database than a relational database.

I'm going to give you right now an idea on when you should be using a graph database instead of something else:

- You have highly related data.
- You need a flexible schema.
- You want to have a structure and build queries that are more similar to way people think.

Instead if you have a highly structured data, you want to do a lot of grouping calculations and you don't have that many relationships between your tables, then you may be better with a relational database.

A graph database has another, not obvious advantage. It allows you to build a knowledge-graph. Because they are graphs, knowledge-graphs are more intuitive. People don't think in tables, but they do immediately understand graphs. When you draw the structure of a knowledge graph on a whiteboard, it is obvious what it means to most people.

And then you can start thinking on building a data fabric, which then can allow you to re-think the way you do machine learning and data science as a whole. But that's material for a next article.

From RDBS to the Knowledge Graph and the Data Fabric

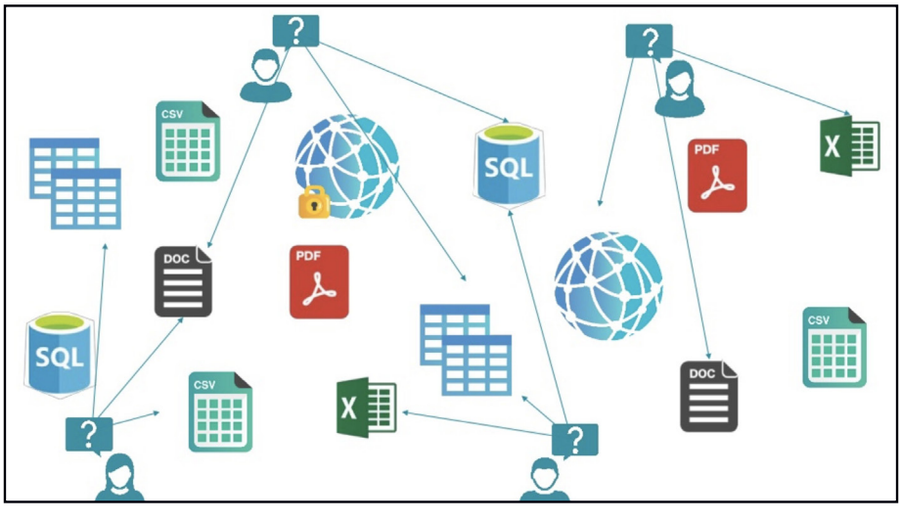
The Data Fabric is the platform that supports all the data in the company. How it's managed, described, combined and universally accessed. This platform is formed from an Enterprise Knowledge Graph to create an uniform and unified data environment.

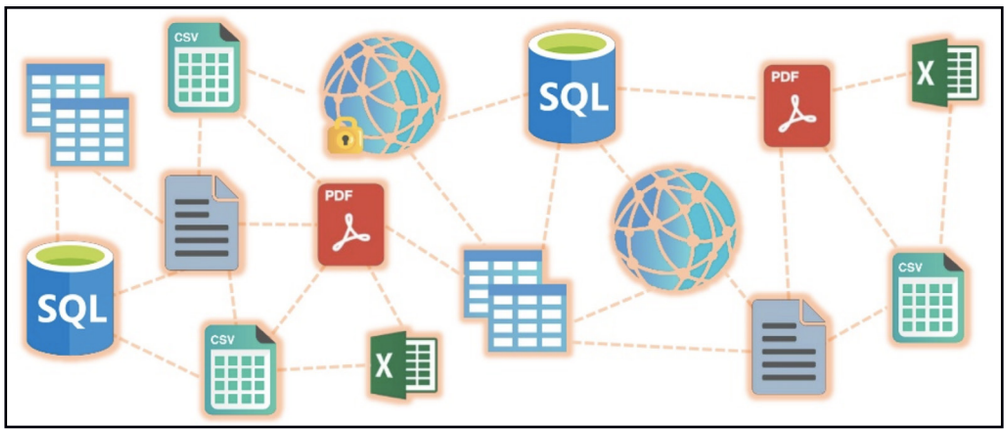
The formation of this data fabric first need to create ontologies between the data you have. This transition can also be thought of as going from traditional databases to graph databases + semantics.

There are three reasons why graph databases are useful:

- Graph Database offerings are showing maturity in capability and diversity
- Graph is being used beyond classical graph problems
- Digital Transformation of complex data requires graph

So if you tie this benefits with a semantic layer, built on ontologies, you can go from having your data like this:





Where you have a human-readable representation of data that uniquely identifies and connects data with common business terms. This layer helps end users access data autonomously, securely and confidently.

And then you can create complex models for discovering patterns in your different databases.

There are lots of advances in automation regarding machine learning, deep learning and deployment. But data is an important asset (maybe the most important one) for companies right now. So before you can apply machine learning or deep learning, at all, you need to have it, know what you have, understand it, govern it, clean it, analyze it, standardize it (maybe more) and then you can think of using it.

We need automation for data storage, data munging, data exploration, data cleansing and all the things we actually spend a lot time doing. That's why my bet it's that semantic technologies are the way to go here. With them you have automatic query generation and using them against the complex graph makes extracting features easy and eventually fully automated.

Conclusions

Graph databases provide an excellent infrastructure to link diverse data. With easy expression of entities and relationships between data, graph databases make it easier for programmers, users and machines to understand the data and find insights. This deeper level of understanding is vital for successful machine learning initiatives, where context-based machine learning is becoming important for feature engineering, machine-based reasoning and inferencing.

For me the key trends for 2019 and 2020 are:

- AutoX: We will see more companies developing and including into their stack technologies and libraries for automatic Machine and Deep Learning. The X here means that this auto-tools will be extended to data ingestion, data integration, data cleansing, exploration and deployment. Automation is here to stay.
- Semantic technologies: On the most interesting discoveries for me this year was the connection between DS and semantics. It's not a new field in the data-world but I see more people getting an interest in the field of semantics, ontologies, knowledge-graphs and its connection to DS and ML.
- Programming less: This is a hard thing to say, but with automation in almost every step of the DS process we will program less and less everyday. We will have tools for creating code and that will understand what we want with NLP and then transform that into queries, sentences and full programs. I think [programming] it's still a very important thing to learn, but it will be more easy soon.

This is one of the reasons why I'm creating this article, trying to follow what's happening across the industry, and you should be aware of this. We will program less, and will use semantics technologies more in the near future. It's closer to the way we think. I mean do you think in relational databases? I'm not saying we think in graphs, but it's much easier to pass information between our heads and a knowledge graph than creating weird database models.