Relational Modelling of Historical Data

Concepts and Challenges

Alexander Watzinger

Presented at the International Medieval Congress, Leeds, July 3th 2017

Digitising Patterns of Power, II: Borders, Power, and the Other (Session 339)

I - Introduction

For the past few years I have been working on the OpenAtlas project, an open source application for archeological, historical and geospatial data, http://openatlas.eu.

The principal projects supporting and using the software are:
DPP - Digitising Patterns of Power, led by Mihailo Popović, http://dpp.oeaw.ac.at and
MEDCON - Mapping Medieval Conflict, led by Johannes Preiser-Kapeller, https://oeaw.academia.edu/MappingMedievalConflict

I'm a software developer with special interest in database structures and information modelling and had the pleasure to work in close cooperation with project members like Stefan Eichert, the initiator of OpenAtlas and master mind of our model.

In this paper I would like to share some of my experiences and insights.

II - The model

Most information that historians work with comes from historical documents. Usually, the first step is to digitize them so that they are easier to work with. Often the next step is to identify bits of information like persons, events and places and map them so that you can ask questions like:

- Where was this person in the course of his life?
- Who took part at an event?

or do even more complex work like social network analysis.

For that you'll need a model. In OpenAtlas we based our model on the CIDOC CRM from the International Council of Museums (ICOM) which at its core consists of entities and relations between them.

Entities can be thought of as nouns like a person, an event, a place.

Relations can be thought of as verbs, linking the nouns e.g. a person participates at an event.

Since this is a very abstract topic I asked my daughter to provide an example with some pictures.

Person (Laura) → participates → Event (Snowman building)

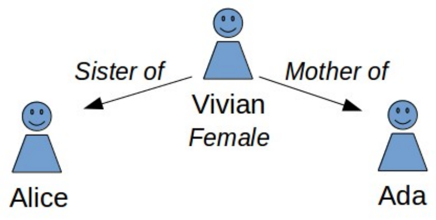
As you can see we have an entity with the class **Person** called **Laura**, which is connected via the relation **participates** to another entity with the class **Event** called **Snowman building**.

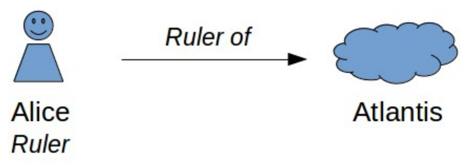
This is how the model works and what the examples are based on.

III - Data consistency

When designing a model there are some pitfalls which I want to show you. For example, in some applications you have persons who can have attributes like a name, day of birth, sex, profession and so on. When you want to connect these persons you are presented with a few options like mother, brother, enemy or king of.

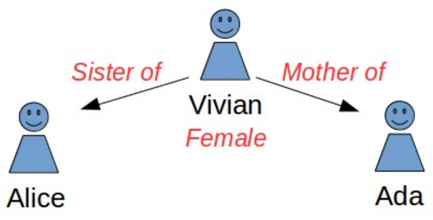
Lets assume you have a woman Vivian who is **female** and has the relation **sister of** to Alice and **mother of** to Ada. Assume further that Alice was the ruler of Atlantis. So you give Alice the profession of **ruler** and also the relation **ruler of** to Atlantis.

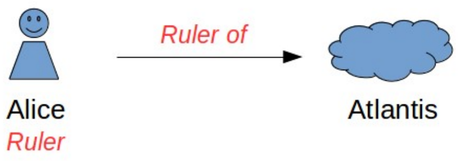




So whats wrong with that? Lets say someone finds out that Vivian was mistakenly thought to be female and but was in fact male. Now the sex and all relations have to be changed.
The same goes for the ruler example: what if there was a translation mistake and Alice was only an adviser, you would have to change it in multiple places.

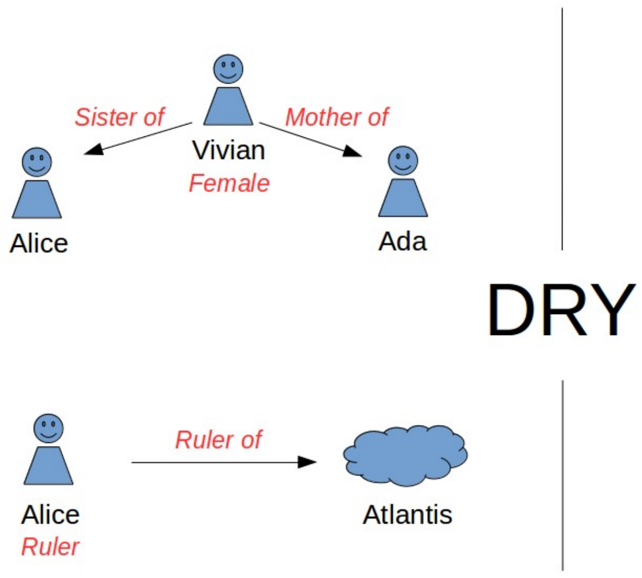
There is a risk that some gets overlooked and you have inconsistency in your data which is very bad and should be avoided at any cost.



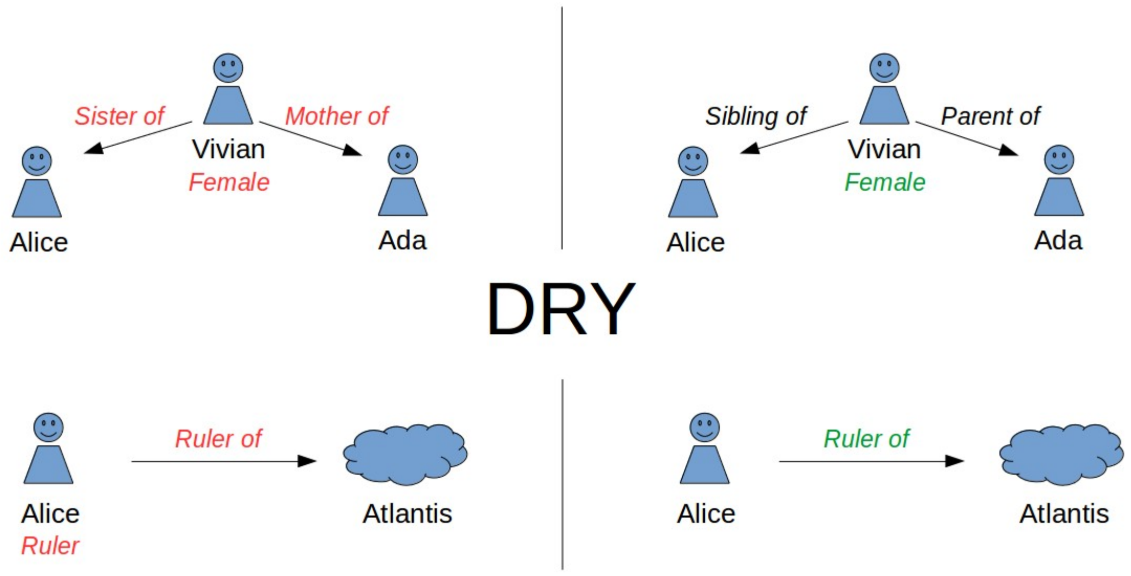


How can you spot and solve these problems? In programming there is a saying: DRY - Don't repeat yourself (Hunt and Thomas, 1999) which can also be applied to data modelling.

If we want to keep the information about the sex only at one place obviously the person itself would be a good choice. With a few changes you'll lose no information and are still able to ask e.g. for all sisters in a given time span and let the system do the math. But now there is only one place where you have to change the relevant information.



As for the ruler example, it is sufficient keep the information in the relation and remove it from the person. The main reason for this is that we would lose information if we kept it only as profession at the person, because then you would no longer know what Alice was ruler of.



So every time you notice that the same information is kept in multiple places, it would be a good idea to rethink your model.

IV - Data Quality

Imagine you want to enter persons from a historical document. The application you are using presents you with a form with the usual fields for name, date of birth and so on and also a required field for the sex with only the standard options female and male.

If you are unsure about the sex your only options would be to guess or don't enter data at all.
To avoid such situations I will give a few pointers.

- Don't force data entry, keep required fields to a minimum. In OpenAtlas for the most part the only required information of an entry is the name.
- Don't force to categorize. If you only have predefined categories, users will often have a hard time finding an appropriate option. We are using hierarchical types which users can expand and even restructure if needed.
- Leave room for fuzzyness. e.g. if you are unsure about a day of birth you can specify a time-span, or if you are unsure about the location of an old church, you can specify multiple locations.

All these things help you improve your data quality because you don't suggest certainty where there is none.

V - You Aren’t Going to Need It

In XP (extreme programming) there is a saying “You Aren’t Going to Need It” - meaning “don't implement things you assume will be used sometime in the future but not now”. Don't get me wrong, it's important to think and plan ahead but implement only things that are needed right now.

This saying applies to several areas.

- Avoid speculative data modelling. Only implement what you are using right now.
- Avoid or remove unneeded software functions. Fortunately there are good tools like unit tests with code coverage to find unused functions.
- Don't overthink categorizations.

In OpenAtlas we make a lot of use of types. This helps to keep the application flexible for different research topics and map information we couldn't have possible thought of in advance. At the beginning of project it's good to think about the general outline and give some example categories as pointers. But don't overdo it by specifying complicated categories that you think you might need at some point. The idea is to keep it open and manage it as it comes.

Often you realize later that you don't need subcategories at one place because there are only three entries in there anyway. And sometimes you look at something and think it's getting a little crowded and half of the entries have a special criteria which would be suitable to make a new sub category.

And by the way, if you are going over your categories and you find one that nobody used, delete it. Because if nobody used it until then there is a good chance that **You Aren’t Going to Need It**.

VI - References

Hunt, A. and Thomas D., 1999, *The Pragmatic Programmer*, The Pragmatic Bookshelf, 320p.

Images shown with kind permission of my daughter Laura Kremser and her cousin Malina Kremser.