Creating and using databases in medieval history: historiography, concepts and practice

Octave Julien

Creating and using databases in medieval history: historiography, concepts and practice

Octave Julien
LaMOP (UMR 8589) / Pireh - Université Paris 1 Panthéon-Sorbonne

Seminários do GIHM – Universidade do Porto
April 17, 2024

I. Introduction: Storing vs. producing knowledge

A database allows data to be recorded in a structured manner, and so it helps researchers organise their work and analyse their sources. When, as historians, we regularly encounter the same kind of informations in the documents we study, we tend to record them in a structured format, in a text document or a spreadsheet. This way we can more easily search, compare and reuse those informations. And as soon as they are stored and used in this manner, they can be described as data.

If data are well recorded, in a consistent manner, they also become easier to share. This opportunity cannot be overstated, as it is very easy today to share data on the Web. In fact, the academic community as well as governing bodies promote the publication of open data, that is data that meet the requirements summed up in the FAIR acronym : they must be findable, accessible, interoperable and reusable. Publishing one's data makes research more transparent, reproducible, and ultimately more solid from a methodological point of view. This is also a good way for you and your work to gain recognition from your peers. Having a database or even a simple dataset online is a good way to achieve this.

So, the first use of a database is to store and retrieve data, whether for personal use, for a group of researchers or even for the general public. In this case, users search the database to find a piece of information that is already there. The second use is to generate new information, new knowledge, based on the data stored in the base. Analysing aggregated data using basic or more advanced statistical tools, can provide us a comprehensive view of our research subject. It enables us to track changes over time, or find correlations between different phenomena. In this case, we don't only use a database to retrieve a piece of information, but to shed light on some underlying structures within the data.

Most online databases are designed as catalogues and they do not allow for such quantitative analysis. But the other way around, if you build a database with this kind of objective in mind, it can also be used as a catalogue. Therefore, I would recommend aiming for the top and considering in advance the statistics that can be obtained from your data. And we will see that it usually improves the design of a database. Moreover, the statistical analysis I mention can be fairly simple. Counting things, calculating averages, or percentages, already is a huge improvement compared to works that only rely on vague adverbs like 'rarely', 'frequently', and so on[1]. Naturally, there are many more advanced statistical tools one can use to classify thing, or to prove or disprove hypothesis more rigorously.

One last, and less obvious, benefit of designing a database for historical research is that it gives a different perspective on the subject and the sources under study. The process of designing a database to represent things from the real world is called modeling, and this is the most important, and usually the most challenging task when building a database. The model of a database is basically its blueprint. It must follow a few principles I will present this morning, it depends on the intended use I have juste mentioned (to build a catalogue or a tool for further investigation), but first and foremost it must accurately represent the historical subject one studies (people, documents, institutions, anything).

Since this presentation is just an introduction, I will focus today on modelling, but there is already much to say on this topic. First, I want to introduce the concept of relational databases and explain why this type of database is useful in historical research. Then, I will go into more details through the concepts of relational databases and the process of modelling. In the third section, I want to shift focus to historiography and give an overview of how historians, particularly medievalists, approached and conceptualized database modeling. Last, I will briefly mention a few tools for those who want to create their own relational database.

II. What is a relational database, and why use one?

1. A database with multiple tables linked together

You may have already thought about the different points I've raised if you have been using a spreadsheet (with Excel for example) to record the data extracted from your sources. An Excel spreadsheet is indeed a database, to some extent.

Marc Humbert gives the following definition of a database: it is a 'structured set of data linked together and stored in a consistent manner (in a computer or on paper cards), without unnecessary redundancy'.[2]

[1] See for example Antoine Prost, Douze leçons sur l'histoire, Paris, Éditions du Seuil, 1996, p. 198-201.

[2] Marc Humbert, *Les bases de données* (Hermes, 1991).

Typically, when using a spreadsheet, we fill it with data row by row, organising specific information into different columns. This utilisation of rows and columns helps us to structure our data and to record them consistently: whatever row we consider, we expect some information to be in a specific column. It makes the data easier to navigate, search or sort.

But Marc Humbert's definition goes beyond this. First, as you have noticed, he says a database can be made of paper cards. We will get back later to this intriguing idea. For the moment, I would like to emphasize the idea of data being linked together and the importance of avoiding redundancy (that is, having some information repeated over and over again).

These characteristics point to the concept of relational databases, which is the type of database I will talk about today. A relational database is basically a database made of several spreadsheets linked together. Instead of using just one, as we do with Excel, we can use as many spreadsheets as we want in order to organise our data. They are called tables, and they can be linked to each other with a numbering system.

Let's consider the information recorded in the prosopographical database *Studium Parisiense*, which describes the career of the masters of the University of Paris in the Middle Ages.[3] In a singular table, as in Excel, it would make sense to use one row for each individual, and each column for their first name, their last name, and their degrees.

Figure 1: Example of prosopographical information organised in a single table

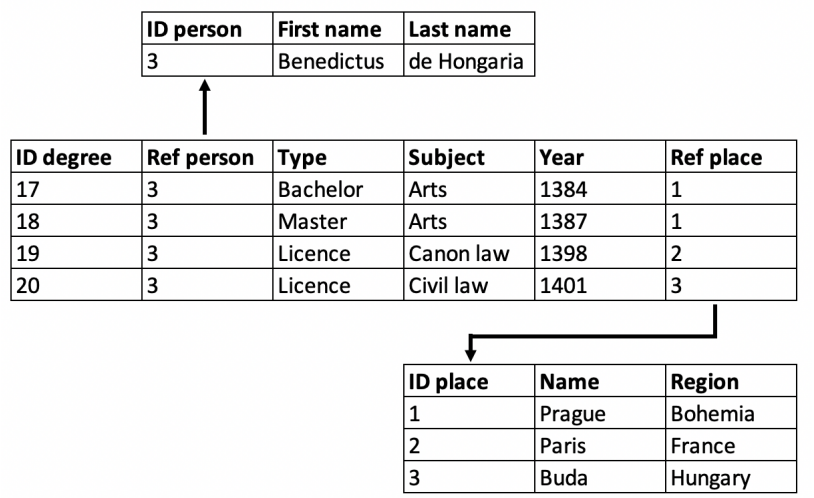
First name	Last name	Degrees
Benedictus	de Hongaria	Bachelor of Arts (Prague, Bohemia, 1384) ; master of Arts (Prague, Bohemia, 1387) ; license in canon law (Paris, France, 1398) ; licence of civil law (Buda, Hungary, 1401)

This method of recording information is clear and precise, but it would lead to multiple issues, mainly because a lot of information is condensed in a single cell. What if we want to find other people who, like Benedictus de Hongaria, got a degree in canon law in Paris and a degree in civil law in Buda ? What if we want to calculate the average duration of the curriculum of these scholars ? The information is there, it is just difficult to search for and utilize effectively.

In a relational database, the same information would be structured like in figure 2, in three different tables. The basic information about the individuals is in the first table. The second table details their degrees, one in each row, with different columns to separate precisely the information we have about them: their type, their subject, the date they were obtained. The last table is used

[3] Jean-Philippe Genet, 'Studium Parisiense, un répertoire informatisé des écoles et de l'université de Paris', *Annali di storia dell'università italiane*, 21.1 (2017), p. 25-74, online: `http://studium.univ-paris1.fr/`

Figure 2: The same information organised in three different tables



ID person	First name	Last name
3	Benedictus	de Hongaria

ID degree	Ref person	Type	Subject	Year	Ref place
17	3	Bachelor	Arts	1384	1
18	3	Master	Arts	1387	1
19	3	Licence	Canon law	1398	2
20	3	Licence	Civil law	1401	3

ID place	Name	Region
1	Prague	Bohemia
2	Paris	France
3	Buda	Hungary

to record information about geographic locations: it tells that Prague is in Bohemia.

As you may have understood, the records in those different tables are linked together by a system of numbers. The individual described in the first table, Benedictus de Hongaria, is assigned the number three. The degrees are also numbered but, more importantly, they point to Benedictus via the second column, labeled **Ref person**. The number three in this column indicates that these degrees belong to Benedictus. Likewise, the numbers in the **Ref place** column serve as references to the places table. They mean that the first two degrees were obtained in place number 1, which corresponds to Prague, that the third one was obtained in Paris, and so forth.

2. Principles and benefits of relational databases

a. Modelling complexity

Of course, this is a simplified example featuring only a small portion of the available data concerning the members of the University of Paris. But it illustrates the advantages of using several interconnected tables. First, it enables the modeling of complex objects, or the representation of different kinds of objects together. In this case, we can consider that we are describing individuals, degrees, and locations. One may argue that all of this constitutes the curriculum of university members, that ultimately we are merely describing people. But it would be possible in a relational database to include additional elements, such as the works they wrote and the manuscripts they were copied in, as it is the case in the *Studium* database.

b. No redundancy

Secondly, by storing data in different tables, one can avoid redundancy, the repetition of the same information over and over again. This is why I included a table for geographic locations. While it would be possible to put the city and the country of the university in the degree table alongside the year, doing so would necessitate repeating that Prague is in Bohemia and Paris is in France. With a separate table, this information can be stored once and for all in one place.

c. Atomicity of data

Using several tables is also necessary to have atomic data. The concept of data atomicity is essential: it means that the data are divided into elementary pieces of information, in as many columns or rows as necessary. This is crucial as it facilitates the search and analysis of data. In this example, it is quite obvious that there was a lot of information to unpack in the column degree of the single table database. With a dedicated table, it would be straightforward to look up a specific degree, to count them by universities, to do calculations on their chronology, and so forth.

Sometimes, the issue of atomicity is not as obvious. To illustrate this point, I would like to move forward in time and look at a census register from 1906 (figure 3). This source already looks like a table —each row describes one inhabitant of the city of Saint-Florent– and it would be tempting to replicate its content in a spreadsheet, with the same columns. But there is this column labeled 'Situation par rapport au chef de menage' or, in English, 'relation to the head of family'. In this column, expressions like 'head', 'wife', 'mother', 'sister', 'son', are listed. Their meaning is unambiguous, but in fact they imply a lot of different pieces of information that must be separated into a more atomic form to be fully utilized.

Indeed, each of these words encompasses at least four pieces of information: the individual's gender, their marital status, their age or the generation they belong to, and the authority they have over the household. For instance, a wife is a woman who is married, who is not a child, and who doesn't have authority. Similarly, a son is a male single child under the authority of his parents, and so on. We must separate these different pieces of informations into distinct columns to analyse them effectively (figure 4). Otherwise, how would it be possible to determine the number of men and women in the city, or the average number of children per family? We know which words designate a man, a woman or a child, but the softwares we use do not.

Figure 3: Excerpt from the census register of Saint-Florent (1906)

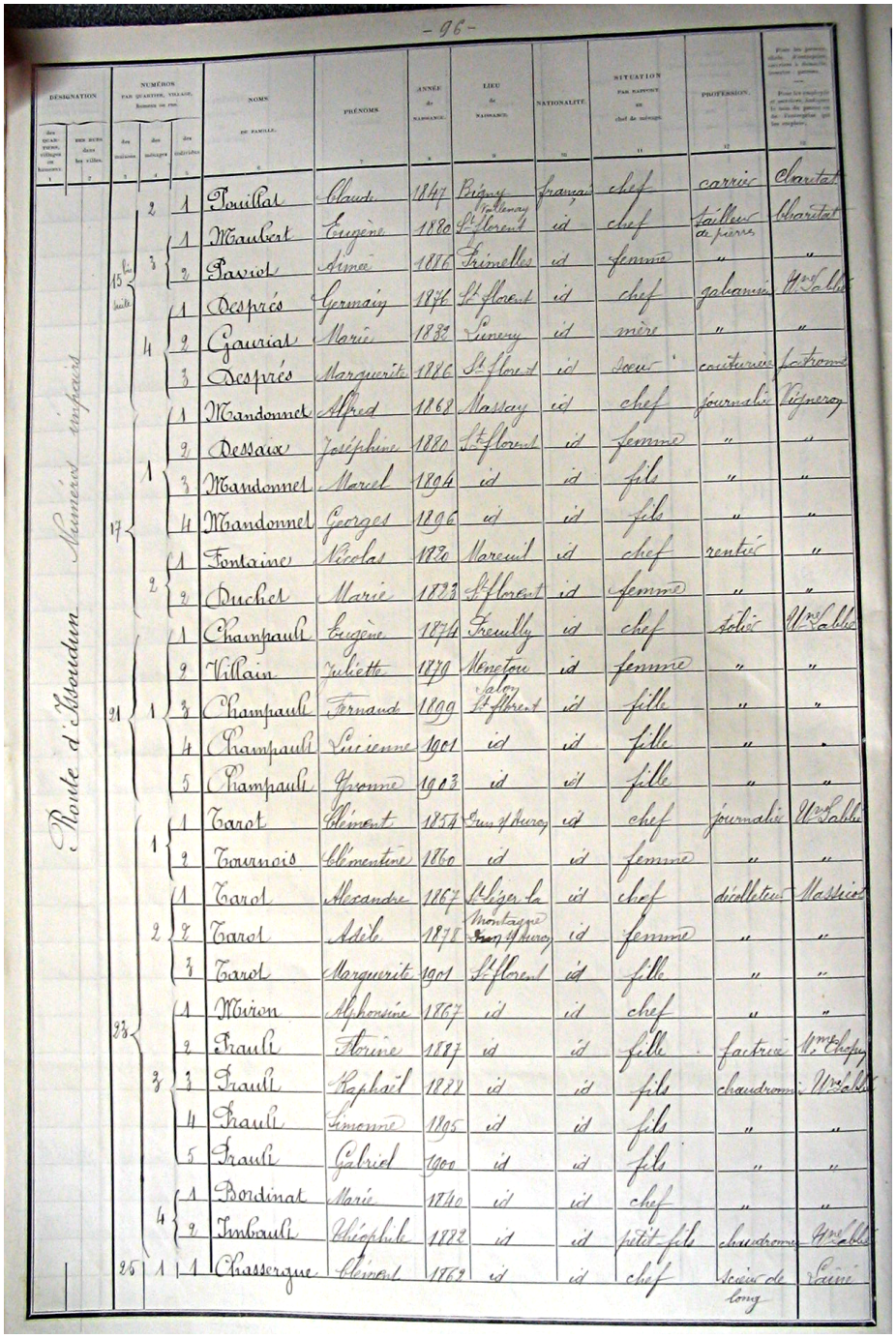


Figure 4: Atomisation of the information regarding the relation to the head of family

Position
chef
femme
mère
fils

Gender	Marital status	Authority	Generation
M	Married	Yes	Parent
F	Married	No	Parent
F	?	No	Grand parent
M	Single	No	Child

d. The importance of using categories

A last principe of modeling I want to introduce is the importance of using categories. A category is a broad class encompassing things that have some common characteristics. In the previous example, the fact that Prague is in Bohemia is recorded although it does not appear in the sources. It is recorded in the database to better exploit the data. Here the region serves as a spatial category for places. Likewise, one could use social categories for people, genres for texts, and so forth. Using categories is essential because it gives a better grasp on the data. It permits the data to be searched more easily: if I want to study French university masters, it is simpler to search for the 'France' region than to look for each possible place of birth that happens to be in France. Moreover, for statistical purposes, it is indispensable to be able to work with a few large, homogeneous subsets of a bigger population. Categories enable this by reducing diversity and dividing the things we study into large groups.

III. The process of modeling

Now you understand the basic ideas behind relational databases. The difficulty lies in the process of modeling, that is the designing of the database blueprint.

After introducing the general concept of relational databases and the principles we need to follow (atomicity, data non-redundancy, utilisation of categories), we can now see the key concepts in more depth, and introduce some terminology.

1. Structure of a table

As we have seen, a relational database is made of tables linked together. Please note that the term 'relational' has nothing to do with the idea of relations between tables. I will discuss its origins and meaning later on. The tables are made of rows and columns. Rows may also be referred to as 'records', and columns can be called 'fields'. Keep in mind that these terms are synonymous. When I talk about 'records', I mean rows, and when I use the term 'fields', I am talking about the columns of a table.

2. Entities and attributes

It is useful to think in a rather abstract way about the things to be described in a database, because it will help designing the database correctly. The things we describe are called *entities*. An entity is a general abstract concept, such as a master of the University of Paris. Not a specific master, who has a name and who existed in history, but the general concept of a university master. Usually, but not always, an entity is represented by a table, with each row in this table describing a specific instance of the entity, in this case a real individual.

Attributes are the characteristics that describe entities: a university master can be described by his first name, his last name, his date of birth, and so on. These pieces of information are attributes of the entity 'university master'. Very often, as you may have guessed, attributes correspond to the columns of the table.

However, in some cases, what seems to be an attribute is actually an entity in its own right. Let's consider the degrees these masters hold. One could think that they are part of the description of a master, since an individual is identified by the degrees he has. But in the process of detailing the information we want to record, it appears that degrees themselves are described by other attributes. Each degree can be described by its type (bachelor, licence, doctorate), its subject (arts, law, medicine, etc.), the year it was obtained and the university it was awarded by. Whenever something can be described by various piece of informations, it can and it should be treated as an entity. In consequence, it should be described in its own table, with its own attributes, like in this example (figure 2).

3. Primary and foreign keys

Not all columns are attributes with an historical significance however. In each table, one column (or field) must be designated to hold a unique number for each record. This field is referred to as the 'primary key' of the table. Because no two rows within a given table can have the same value as a primary key, they will always be distinguishable, even if their contents are identical (consider two persons born the same day with the same name).[4] Primary keys are used to distinguish records and to link records from different tables. A column in a table can contain a value that is a reference to the primary key of another row in a different table. Such a column is called a *foreign key*. We have seen this system in the example of the masters of the University of Paris. Here, **ID person**, **ID degree** and **ID place** serve as primary keys of their respective tables, while **Ref person** and **Ref place** are foreign keys that point from one table to the others.

As it is the case here, several rows in a given table can have the same value as a foreign keys. In fact, this is what makes it useful. This enables multiple rows

[4] In reality, a primary key can be made of strings of characters, letters, words, or any type of data. It is also possible to use a combination of two or more fields as a primary key, provided this combination is unique for each row.

in one table to be linked to a single row in another. By convention, primary keys are called *ID something* (*ID* standing for 'identifier'). The system of calling foreign keys with the prefix 'Ref' (as in 'Reference') is a personal preference. In my opinion it helps to understand the structure of the database.

4. Cardinality

Lastly, I want to address a fundamental question when designing a database: the cardinality of relationships between tables. Cardinality indicates the number of entities that can be linked to each other. This is very important because it determines how tables can relate to each other with foreign keys.

Suppose we are building a database to describe the study the production of manuscripts in medieval scriptoria. Our sources are the manuscripts that have survived until today. After considering the information we want to record, we identify three entities: scriptoria, manuscripts, and texts. Scriptoria produce manuscripts, which contain texts.

The first relationship to consider is the one between scriptoria and manuscripts. How many scriptoria and manuscripts are involved in this relation? We need to establish a minimal and a maximal number.

The best way to answer this question is to adopt each point of view. Let's consider one scriptorium : how many manuscripts did it produce? It produced at least one manuscript, otherwise it wouldn't be called a scriptorium, and it may have produced many. So the minimum is 1 and the maximum is many, represented as 'n'. Now, let's take the point of view of a manuscript : in how many scriptoria was one manuscript produced? At least one (since we are only considering professional production of codices) and at most one as well, because a manuscript cannot have been copied in several scriptoria at the same time (in fact it can, but we will keep things simple). The cardinality we have just determined can be written this way:

$$\text{Scriptorium } [1,n] \longleftrightarrow [1,1] \text{ Manuscript}$$

It means that a scriptorium can be related to one or more manuscripts, and each manuscript is related to one single scriptorium.

Now, what is the cardinality of the relationship between manuscripts and texts? A manuscript can contain one or more texts. And a given text can appear in one or more manuscripts. So, on both sides of the relationship, the cardinality is [1,n]: at least one, possibly more.

$$\text{Manuscript } [1,n] \longleftrightarrow [1,n] \text{ Text}$$

Finally, for the sake of completeness, let's consider the scenario where we need to record information about the digitised version of a manuscript, such as its web address, the institution that made it, whether it is in color or black and white, or other relevant information. It makes sense to consider a manuscript

and its digital artefact as two distinct entities. One digitised version is link to one and only one manuscript, and each manuscript is linked to zero or one digitised version. In this case, the cardinality would be:

$$\text{Digitisation } [1,1] \longleftrightarrow [0,1] \text{ Manuscript}$$

In fact, for the purpose of designing the database, what is important are the maximum values of the cardinality, represented by the second number between brackets. If we consider only these, there are three possibilities: 1:1 (one to one), 1:n (one to many) or n:n (many to many). The cardinalities of the relations between entities determine what tables are needed and in which tables the foreign keys should be.

5. Data types

The last concept we need in order to understand how relational databases work is data type. Contrary to a spreadsheet software like Excel, every field of a database must have a specific type, which puts restrictions on what can be recorded in it. The main ones are strings of characters (any combination of letters, numbers, punctuation marks, and so forth), integer numbers, decimal numbers (non-integer numbers, with a decimal part), dates and times (although times are rarely uses in medieval history), and booleans (a type with only two values, True and False, akin to a checkbox). The database system requires these types to properly handle the data, and it limits what you can do with them: For instance, it won't be possible to perform calculations with numbers recorded as strings for example. Or you won't be able to append a question mark after a number stored as an integer to indicate that the value is doubtful, because a question mark is not a number and it does not match the integer data type.

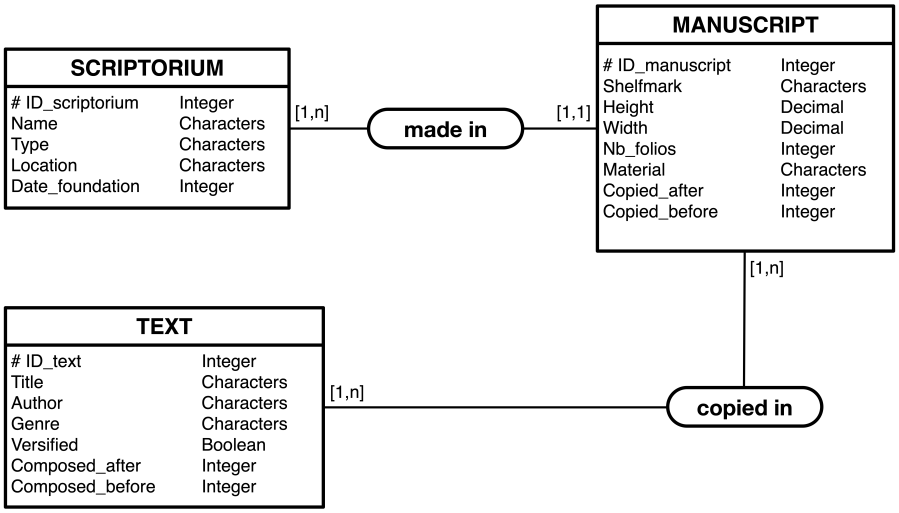
At this point, we have seen all the key concepts needed to model a database. We can now determine which tables compose our model and how they articulate. This takes the form of a conceptual data model.

6. The conceptual data model

A conceptual data model is the detailed blueprint of a relational database. It is usually represented as a diagram, as depicted in figure 5. In these diagrams, tables are denoted by square boxes, with their name. The relationships between them are illustrated as lines, with their cardinality and their meaning written in a rounded box.

In each table, its fields and their data types are listed. Here, I have included a selection of fields that would be relevant for describing scriptoria, manuscripts and texts. By convention, primary keys are indicated by a little symbol, such as a sharp sign or a little key. However, the foreign keys are have not been included. Our blueprint does not yet illustrate how the tables are linked together. It depends on the cardinality of their relations.

Figure 5: Sketch of the conceptual data model of the manuscripts database



a. Linking two tables: Case 1:n

When you have a one to many (1:n) relationship, the foreign key must be in the table of the multiple entity. Accordingly, in this example, a foreign key is required in the Manuscripts table, pointing to the Scriptorium table (since multiple manuscripts can be associated with one scriptorium). You can remember this as a rule, but it also makes logical sense when considering the issue of data atomicity. The foreign key could not be placed in the Scriptorium table, because in some cases it would be necessary to store multiple values (the ID of several manuscripts) within a single cell.

b. Linking two tables: Case 1:1

If the cardinality of their relationship is 1:1, the foreign key can be placed in either table, pointing to the other one[5]. In fact, in this case, any row in a table is linked to no more than one row in the other. Then, it is possible to simply merge the two tables into a single one, even if they describe two distinct entities. This is not mandatory but it makes things easier. In the example above, we would simply incorporate the information about the digitised version into new columns of the manuscripts table. If a manuscript has not been digitised, these columns can simply be left empty.

[5] It is also possible not to add any foreign key, and to use the same values for the primary keys in the corresponding records of each table.

c. Linking two tables: Case n:n

Relations with a n:n cardinality, like the one between manuscripts and texts, are the tricky ones. In this case, a foreign key cannot work in either table. We cannot use a foreign key column in the Manuscripts table because it would need to hold as many values as there are texts in a given manuscript. Likewise, we cannot rely on a foreign key in the Texts table, in case we need to use several IDs for the different manuscripts in which a given text appears.

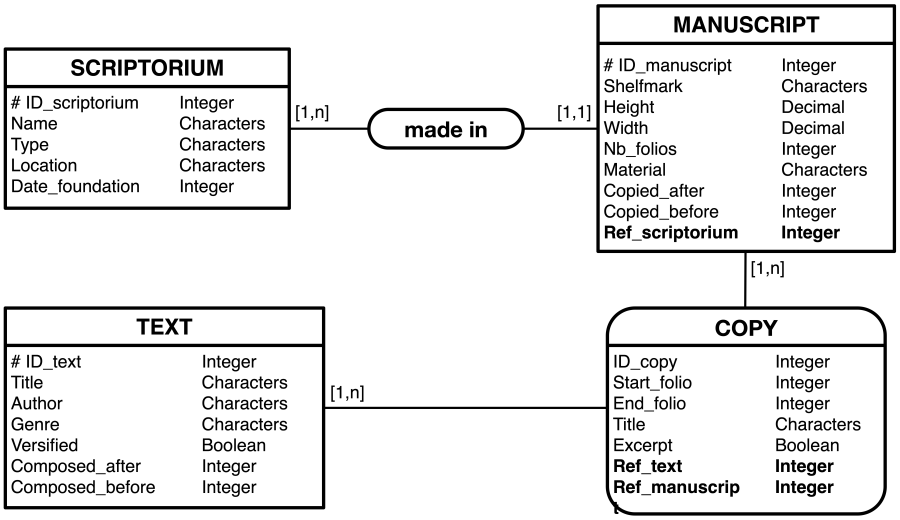
The solution is to add a third table, known as a 'junction table', in order to represent each relation between the first two (figure 6). A junction table relies on three columns: a primary key (as in any table), a first foreign key pointing to one table, and a second foreign key pointing to the other[6]. Each row of the junction table indicates that one specific text appears in one specific manuscript.

This represents the minimal structure of a junction table. But very often, it appears it is the ideal place to store information about the association between two entities. Here for example, we can use the junction table to our advantage to record the foliation of the text in a manuscript (where it begins and ends). We can also indicate the title given to the text in this manuscript, or whether it is an excerpt or the complete text. All such information depends on both the text and the manuscript under consideration. It cannot be recorded in either table, but it can in the junction table. We can give a meaningful name to this junction table, such as 'Copies', to reflect its purpose.

Now the definition of Marc Humbert makes more sense : a database is a 'structured set of data linked together and stored in a consistent manner (in a computer or on paper cards), without useless redundancy'. It is a structured set of data because we use the rows and columns of tables to store data in a organised way. These data can be linked together thanks to the relational system. And, as we have seen, it can eliminate redundancy. But why does Marc Humbert talk about paper cards? Actually, it makes sense if we consider how data were handled by historians since the 19^{th} century.

[6] In fact, it is possible to define a composite primary key based on the two foreign keys only, as long as their combination is unique in each record, which is not necessary the case in this example.

Figure 6: Final conceptual data model of the manuscripts database (with foreign keys and the junction table)



IV. The uses of databases in historiography

1. Files and paper cards: databases without computers?

In this section I will briefly cover the use of data by historians, with an emphasis on medieval history and French historiography, given my familiarity with it.

It is worth noting that the reflection on the right way to handle data is consubstantial to the birth of modern history. In France this moment is often dated to the publication of Charles-Victor Langlois and Charles Seignobos *Introduction aux études historiques* in 1898[7]. Although they did not explicitly speak of data, they already emphasized the importance of well-structured information. In their book, they explain how historians can properly record the information taken from sources[8]. They advocate for a system based on paper cards, where each source is transcribed or described on a card. Additional cards can be used to record several pieces of information separately, with a systematic reference to the source. They discuss the categories that can be written down and used for sorting : dates, geographic origin, type of document, etc. In short, they describe a system of loosely structured information linked to each other, with a reference system, similar to foreign keys, in order to avoid redundancy. The benefit of this system is to make information easier to filter and sort. For this reason, they dismiss their fellow historians who continue to write in notebooks. This

[7] Charles-Victor Langlois, Charles Seignobos, *Introduction aux études historiques* (Kimé, 1992, 1st ed. 1898).

[8] *Ibid.*, p. 93-100, 126 et 163.

use of paper cards was already common at the time they wrote. Historians may have been influenced by librarians, who had already devised such card-based catalogues in large wooden file boxes with three separate directories for books, authors and subjects[9]. Cards were widely used among historians well into the 20th century, at least until the 1980s. For instance, George Duby writes in his autobiography about the tens of thousands of paper cards he wrote down for his PhD, organised in thematic files. He continued to use cards until at least 1982-1983[10].

Obviously, paper cards are much more limited than a proper computer database. However, they helped historians handle large amounts of information required for broad works typical of the *Annales* school. And it certainly made it easier for them to transition to computer databases in the 1980s.

2. Mechanography and navigational databases

Historians who wanted to conduct statistical analysis before that period used other technologies : mechanography and punched cards. Punched cards are paper-cards where holes are made to encode data. They can be processed with mechanical machines designed to punch the holes, read them, sort the cards and perform calculations on them. This system was invented in the United States for the 1890 census. For a long time, it was the main activity of the firm IBM (which stands for 'Industrial Business Machines')[11]. Modern computers, based on electronics, were invented during the Second World War and they gradually replaced mechanography afterwards. Even though, punched cards were still used for data storage until the 1980s.

The standard punched card is 18.7 cm long and 8.2 cm wide. It is organised in 80 columns and 12 rows. With the appropriate combination of holes, each column can encode one character (figure 7).

In France, historians of the early modern period began using punched cards as early as the 1960s, mainly for recording and analysing demographic and economic data. [12] In medieval history, a famous example is the analysis published in 1978 by Christiane Klapish-Zuber and David Herlihy of the *Catasto*, a fiscal census conducted in Florence between 1427 and 1430[13]. Using punched cards,

[9] See Françoise Waquet, *L'ordre matériel du savoir. Comment les savants travaillent, XVIe-XXIe siècles* (CNRS Éditions, 2015), p. 74-89.

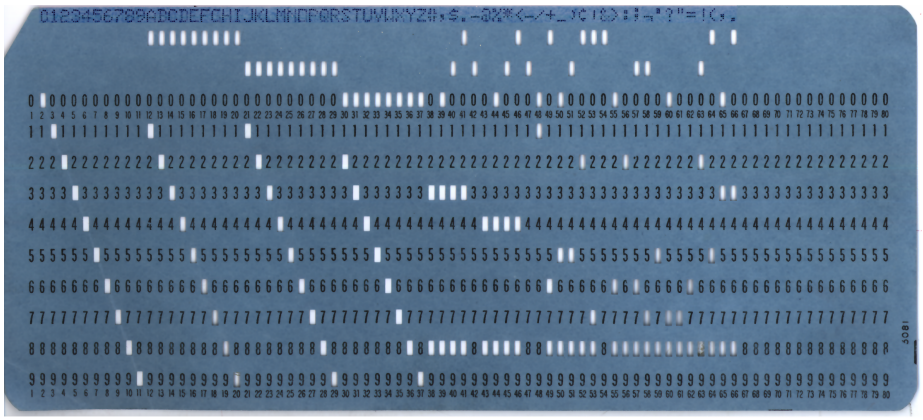
[10] *Ibid.*, p. 88.

[11] Delphine Gardey, *Écrire, calculer, classer*, p. 263-267

[12] Adeline Daumard, François Furet, 'Méthodes de l'histoire sociale : les archives notariales et la mécanographie', *Annales. Economies, sociétés, civilisations*, 14-4 (1959), p. 676-693 ; Jean Delumeau, 'Méthode mécanographique et trafic maritime : les terre-neuviers malouins à la fin du XVIIe siècle', *Annales. Economies, sociétés, civilisations*, 16-4 (1961), p. 665-685 ; M. Couturier, 'Démographie historique et mécanographie électronique', *Annales de démographie historique* (1966), p. 57-78 ; J.-C. Perrot, J. Sutter, M. Couturier, 'Nouveau débat sur démographie historique et mécanographie électronique', *Annales de démographie historique* (1967), p. 29-61.

[13] Christiane Klapisch-Zuber, David Herlihy, *Les Toscans et leur famille. Une étude du Catasto florentin de 1427* (Presses de la Fondation nationale des sciences politiques, 1978).

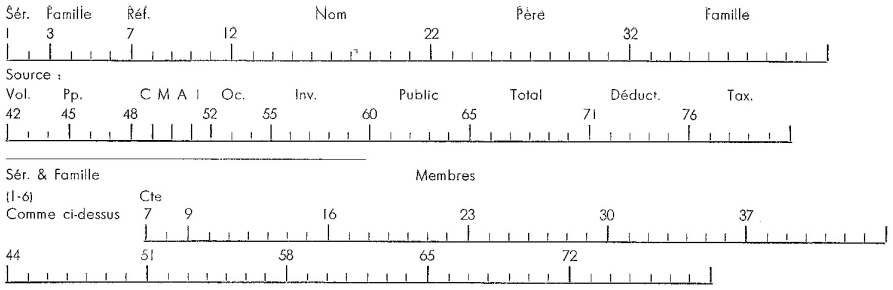
Figure 7: An 80-columns punchcard. Source: https://commons.wikimedia.org/wiki/File:Blue-punch-card-front.png



the two historians and their team were able to encode data on approximately 60,000 taxpayers and their family, and conduct a comprehensive analysis of the demography and economy of Florence.

But it had some limitations, which Christiane Klapish-Zuber discussed in a paper written in 2016[14]. The information was first written down on forms like the one below (figure 8), one for each individual.

Figure 8: Form used to manualy record data from the Catasto before the punching of cards.



The numbers indicated there correspond to the column system of the punched cards. For example, columns 12 to 42 were designated for recording the names of the head of a household, of his father and his family. Columns 60 to 80 were

[14] Christiane Klapisch-Zuber, Béatrice Marin, Nicolas Veysset, 'Le catasto florentin de 1427-1430 : présentation générale de l'enquête et du code', *L'Atelier du Centre de recherches historiques*, 2016, doi: 10.4000/acrh.7458 (accessed April 10, 2024).

used to record numbers about their wealth. Additional cards could be utilised to detail the members of one family (see the lower half of the form). So, while the information was highly structured, it raised some issues: The available space was sometimes insufficient, limiting the recording to no more than 10 members for a given family. It lacked flexibility, since they could not add other different informations as they went through the source. In some cases, they had to put several pieces of information together into a single entry on the form, which contradicted the principle of atomicity.

In order to understand the use of databases by historians at that time, one must consider not only the data they used and the type of research they did, but also the whole environment they worked in. Indeed, the practical limitations of punched cards was not the only issues faced by historians who wanted to use a database in their research. Firstly, it is worth remembering that in the 1960s and 1970s, the only available computers were mainframes, large computers shared by many users, each for a limited amount of time. Secondly, before the introduction of relational databases, people used navigational databases that were limited or difficult to use. They were also based on the concept of fields and records, and their logic also aimed to maximize atomicity and reduce redundancy. But the most common ones could not handle n:n (many to many) relations between records. Moreover, the main problem was that the data, the conceptual model, and the inner workings of the computer were inextricably intertwined. To query the database, users had to understand how the memory of the computer worked. Practically speaking, this meant that historians (or any user, for that matter) were very dependant on computer technicians and programmers to build and use their database.

In fact, at that time, using a database was a collective enterprise. Researchers relied on computer technicians, as I have just mentionned. For the largest or the most innovative projects, they frequently collaborated with computer companies that provided them with the necessary equipment and assistance. When cards had to be punched, they also relied on employees devoted to this task. This idea of a collective enterprise involving many different people is illustrated by the organisation of Roberto Busa's work. I won't delve too much into details because it is not strictly a database. Suffice it to say that Roberto Busa was a theologian and historian who wanted to create an index of the monumental work of Thomas Aquinas. To achieve this, Roberto Busa made a partnership with IBM in the 1950s. But he also had a large team of mostly female workers who read Thomas Aquinas works and copied the text onto punched cards (figure 9). This gendered division of labor, that has long been ignored, including by Busa himself, is the subject of a recent book by Julianne Nyhan.[15]

When one reads testimonies by historians from that period about their use of computers, one often catches a glimpse, between the lines, of all those anonymous people who contributed to the task: technicians, typists, temporary work-

[15] Julianne Nyhan, *Hidden and Devalued Feminized Labour in the Digital Humanities On the Index Thomisticus Project 1954-67* (Routledge, 2022).

Figure 9: R. Busa's employees punching cards for the *Index Thomisticus* in Gallarate (Italy, 1967).

ers or unpaid students, and so forth. For the better, this environment also fostered interdisciplinarity. As I mentioned earlier, historians had to use mainframe computers for some time, which were shared by many researchers. Practically speaking, this meant they had to wait for the computer to run their program with their data. Jean-Philippe Genet, my PhD supervisor, and one of the most fervent advocates of computing in history, recounts that while he was waiting for the computer in the town of Orsay, south of Paris, in the 1970s, he had a lot of discussions with mathematicians, linguists or physicists, which was very beneficial.

3. Relational databases

The introduction of relational databases, along with personal computers[16], solved most of these problems. In the first part of this presentation, I explained how they allow data to be well-structured. Now, I would like to discuss how they changed the way historians worked with databases and approached modeling.

The concept of relational databases was presented in 1970 by a British mathematician and computer scientist named Edgar Frank Codd. At that time he was working at IBM in the United States. Codd's ambition was to solve the problems of previous database systems. To do that, he developed a new branch of mathematics called relational algebra. This is where the word 'relational' comes from. A relation is a mathematical concept that describes a set of infor-

[16] The diffusion of personal computers can be dated back to 1977, with the release of three popular models, the Apple II, PET 2001 and TRS-80.

mations organised in different fields. It has nothing to do with the fact that we establish relations between the tables of a database. So Codd founded a new theory of databases on firm mathematical basis. The paper he published on this topic is probably one of the most cited and the least read in the history of science because it is very abstract mathematics. [17]

However, the benefit of this mathematical foundation is that the rules for designing and querying a database are entirely abstract. This is a good thing because it means that, unlike older models, a database's logic (its conceptual model) is completely separated from the inner workings of the computer (its physical model). A given conceptual model, if it adheres to the principles of a relational database like those we have seen in the first two sections, can be implemented on any computer, with any software. It could even be simulated with piles of paper cards.

This is also true of the computer language Edgar Codd and his colleagues devised to query a relational database, named SQL (for 'structured query language'). Likewise, SQL can be used on any relational database system[18]. It is based on English words and basic notions of logic. With the development of personal computers and relational databases, any person could now create and use their own database.

4. The uses of relational databases in medieval history

This greater accessibility of databases explains why historians have frequently used them in the 1980s, despite the decline in interest in quantitative history. While many have employed databases during this period, few actually wrote about the subject. Conceptual data models are rarely published, which makes any kind of historiographical survey difficult to do.[19] However, two theories of modeling have had a lasting influence, and I wish to discuss them and highlight their implications in the context of prosopographical databases.

[17] Edgar F. Codd, 'A relational model of data for large shared data banks', *Communications of the ACM*, 13-6 (1970), p. 377–387.

[18] SQL is actually a family of languages, with some differences regarding their syntax and their implementation.

[19] A few exceptions in French medieval history are Muriel Gougerot, Jacques Moureton, « L'adaptation d'une base de données : MEDIUM », *Le médiéviste et l'ordinateur*, n° 31-32, 1995, p. 28-32, online: `http://www.persee.fr/doc/medio_0223-3843_1995_num_31_1_1422` (accessed April 10, 2024) ; Caroline Bourlet, Agnès Guillaumont, « Test d'une méthode de conception de projet sur un corpus médiéval : les artisans parisiens sous Philippe le Bel », *Le médiéviste et l'ordinateur*, n° 23, 1991, p. 5-11, online: `https://www.persee.fr/doc/medio_0223-3843_1991_num_23_1_1308` (accessed April 10, 2024) ; Olivier Matteoni, « Une base de données informatisée pour l'étude prosopographique du personnel politique de la principauté bourbonnaise a la fin du Moyen Âge : présentation et exploitation », *Medieval Prosopography*, vol. 19, 1998, p. 99-109, online: `https://www.jstor.org/stable/44946284` (accessed April 10, 2024).

a. The concept of metasource

The first one is based on the concept of 'métasource' introduced in 1986 by Jean-Philippe Genet[20]. A metasource is a computer system, a tool, used by historians to formalize the information taken from historical documents. It is not necessarily a database, even though the concept is highly relevant in that context. It can be a system of maps or a corpus created for lexical statistics, for example. Genet argues that a metasource ultimately serves the purpose of quantifying historical phenomena, counting and measuring things, in order to properly describe and explain the past. To achieve this, historians reconstruct the historical past within a computer system. Of course, this process is not neutral; what they create is an artefact. According to Genet, it is determined by two theories. What he calls a global theory is basically the specific question underlying a research and the intellectual paradigm one adopts (cultural anthropology, marxism, etc.). The local theory is more directly related to the use of computers. It consists of the variables and categories used to structure data, and it depends on how we conceive and describe our objects.

Indeed, the benefit of using a database is not only to manipulate large amounts of data. Before that, the way we design the database and record information raises many questions that force us to think through our object. My PhD was a study of manuscript miscellanies copied in France and England at the end of the Middle Ages. Because I used a database to describe the form and contents of such manuscripts, I had to ponder many questions that I could have ignored if my analysis was directly based on the manuscripts: What is a text? Should I divide texts that form a collection within a manuscript? Should I consider small notes added by the owner of a manuscript as a text? Is it important to record the presence of blank pages in a book (it is)? Categories used to classify things are also a central issue: for example, what classification should be used to describe the social hierarchy of the society under investigation? Should we adhere to the classification used by the people we study (if it exists), should we follow one defined by another historian, or define one ourselves?

All the choices we make regarding those questions define a local theory. Jean-Philippe Genet sums up the idea by saying that the historian who creates a metasource is a 'démiurge', some kind of god, who designs a closed world with its contents and its internal laws.

b. Factoids

Contrary to the closed world described by Jean-Philippe Genet, some historians have tried to design databases that describes sources as closely as possible.

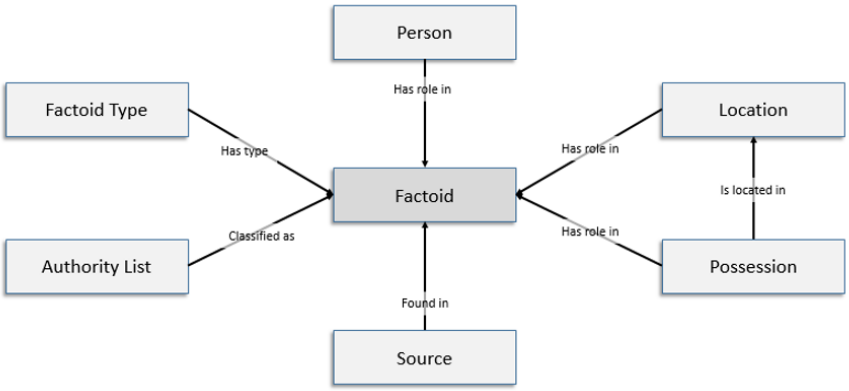
The factoid model was first designed to create a prosopographical database of the Byzantine Empire, and later for a a database of Anglo-Saxon England.[21]

[20] Jean-Philippe Genet, 'Histoire, informatique, mesure', *Histoire et mesure*, 1.1 (1986), p. 7-18.

[21] Michele Pasin, John Bradley, 'Factoid-based prosopography and computer ontologies: towards an integrated approach', *Digital Scholarship in the Humanities*, 30-1 (2015), p. 86–97, online preprint ; John Bradley, 'What is Factoid Prosopography all about?', *King's College*

The factoid model puts facts, or rather factoids, at the center of the conceptual model. As the name suggests, a factoid is something that looks like a fact, given in a source. It can be a true or a false statement, and it can be implicit or explicit. The name actually comes from the American journalist Norman Mailer, in his biography of Marylin Monroe. With some irony, he defines factoids as 'facts which have no existence before appearing in a magazine or newspaper'. Because it is centered on factoids, this model is also centered on the sources they appear in. The simplified conceptual model of a factoid database looks like figure 10.

Figure 10: Simplified conceptual model of a factoid database.



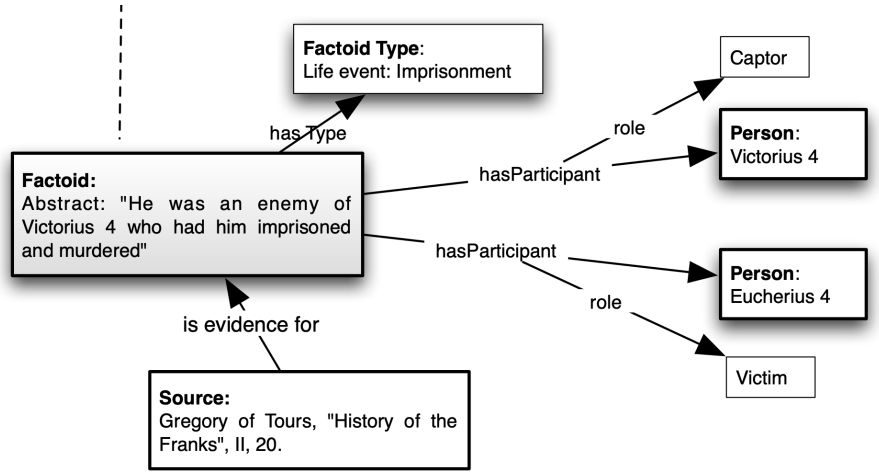
Factoids are found in sources, and, because they are used to build prosopographical databases, they relate to one or several persons. Through a factoid, it is possible to record that a person is related to a location (the place of their birth, where they lives, or where a specific event happened, etc.). Individuals can also have possessions, and since this model was used for databases covering the Middle Ages, these possessions often include pieces of land, hence the link between possession and location.

To illustrate how this model is used, Michele Pasin and John Bradley give the following example (figure 11), based on an extract from Gregory of Tours' *Historia Francorum*[22]. This extract mentions that a nobleman named Eucherius 4 was imprisoned by Victorius 4. So, the factoid can be classified as a life event, and more precisely as an imprisonement. It is found in book 2, chapter 20 of Gregory of Tours, and this event involves two persons : Eucherius 4, who is the victim, and Victorius 4, who is the captor.

London (n.d.), online: `https://www.kcl.ac.uk/factoid-prosopography/about` (Accessed April 10, 2024).

[22] Michele Pasin, John Bradley, 'Factoid-based Prosopography and Computer Ontologies', p. 3-4.

Figure 11: Detailed example of a factoid



So, you have to imagine that in a prosopographical database based on factoids, all information is coded this way. Many more types of factoids can be used. In the online *Prosopography of Anglo Saxon England*[23], it is possible to find factoids classified as adultery, drunkenness, insult, envy, etc.

The previous examples illustrate the difference between two types of prosopography and two types of databases, even if both follow the relational model[24]. Old-style prosopography aimed at compiling as much information as possible on a large population from periods of history where the sources are limited, such as the late Roman Empire, Byzantium, or Anglo-Saxon England. These prosopographies did not intend to answer specific research questions; they were rather used as a reference for other historians seeking information about an individual. Factoid databases follow this logic: they are designed as a reference tool, a database to store and search informations from various sources, which other historians can use as they wish. This explains why, as Michele Pasin and Paul Bradley put it, 'the factoid approach prioritizes the sources, rather than [the] historians' reading of them'. The goal is to maintain a neutral stance towards the source, even if the selection of facts, their categorisation, cannot be completely neutral.

On the contrary, more recent prosopographical studies, since the 1970s, have focused on smaller, more homogeneous social groups to answer specific questions. This approach is more common for the late Middle Ages or the modern era, where sources are much more numerous and rich. It justifies the design and

[23]https://pase.ac.uk

[24] On this distinction, see Claire Lemercier, Emmanuelle Picard, 'Quelle approche prosopographique ?', in Laurent Rollet, Philippe Nabonnaud (ed.), *Les uns et les autres. Biographies et prosopographies en histoire des sciences* (Presses Universitaires de Nancy, 2012), p. 605-630, online : `https://shs.hal.science/halshs-00521512v2/document` (accessed April 11, 2024).

use of a database as a metasource, a description of a small part of the past construed with a local theory in order to focus on specific issues. This dichotomy is reflected in the use of quantitative analysis. Large prosopographies, such as the *Prosopography of Anglo Saxon England*, is not designed to produce statistics, whereas this is central in the concept of metasource defined by Jean-Philippe Genet. Of course, some databases can fall between the two types.

V. Some common modeling problems in medieval history

In this last section, I want to discuss three common problems historians of the Middle Ages face when they design a database, and what solutions can be implemented in a conceptual data model.

1. Describing social networks

A social group can be studied through its social network, that is to say the relationships existing between its members. These relations can be family ties, friendships, relations between employers and employees , and so on. In fact, it is also possible to model with a database a network of things other than human beings: they can be cities or ports linked by trade routes[25], companies doing business with each other, books that have a text in common[26], and more. However, since I am focusing on prosopographical databases, it is worth focusing on human social networks. The following example (figure 12) is based on the conceptual model of a database of Parisian artisans.[27] One obvious entity, with its own table in the database, is the individual. In order to describe the relationships between two individuals, another table is needed. It resembles a junction table, with two foreign keys, but both point to the same table. Each record in this table indicates that one individual, called *ego*, is linked to another individual, called *alter*. It is possible to add other information in this table, such as the type of relation, if multiples types are possible. One can also add dates indicating when the relation existed, or a reference to a source. It is also necessary to take into account the fact that some relationships have a direction. Friendship, for instance, can be considered bidirectional: if A is a friend of B,

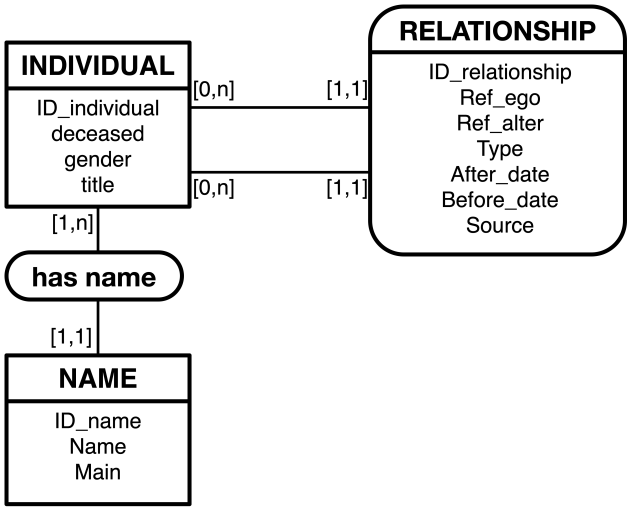
[25] One of the first application of social network analysis to medieval sources was a description of the trade network of medieval Russia : Forrest R. Pitts, 'The medieval river trade network of Russia revisited', *Social Networks*, 1.3 (1978), p. 285-292, doi: hrefhttps://doi.org/10.1016/0378-8733(78)90025-410.1016/0378-8733(78)90025-4, (accessed April 16, 2024).

[26] See Octave Julien, 'Délier, lire et relier. L'utilisation de l'analyse réseau pour construire une typologie de recueils manuscrits de la fin du Moyen Âge', *Hypothèses*, 19 (2016), p. 211-224, doi: 10.3917/hyp.151.0211 ; *Networks of Manuscripts, Network of Texts*, ed. by Evina Stein, Gustavo Fernández, special issue of the *Journal of Historical Network Research*, 36.2 (2023), doi: 10.25517/jhnr.v9i.

[27] Caroline Bourlet, Agnès Guillaumont, 'Test d'une méthode de conception de projet sur un corpus médiéval : les artisans parisiens sous Philippe le Bel'.

then B is a friend of A (although in fact, sociologists who have studied friendship networks know that the feeling is not always mutual). On the contrary, a relation of patronage is asymmetrical; one must be able to distinguish the patron and the client. In such cases, relations can be recorded twice, once for each direction, or one can define in advance which point of view one uses to describe a relationship. This is why I like to call the foreign keys *Ego* and *Alter*, to keep in mind that when I describe a relationship, I am adopting the standpoint of *Ego*.[28]

Figure 12: Detail of the conceptual data model of the database of Parisian artisans



2. Multiple attributes: the example of names

Another common problem with medieval sources is the identification of individuals. A person can be found with different names, in Latin or in vernacular, or with different spellings. It is common to use a distinct table to record those different names, with a n:1 relation to the individuals table (figure 12). In this case, as with any table used to record multiple attributes, it is wise to use a field to designate one name as the main one, or the official one, so that it is easy to associate one person with one name.

[28] This way of modeling relationships between two individuals, with two foreign keys pointing to the same table, poses a challenge when querying the database. For a given relation, a query cannot point to two records from the table of individuals simultaneously. The solution is to work with two distinct copies of the individuals table, which can be done with views or subqueries.

More generally, when an attribute can have multiple values for one thing, it is good practice to choose one main one arbitrarily. Otherwise, any calculation based on this attribute may result in one thing counted several times.

3. Modeling uncertainty

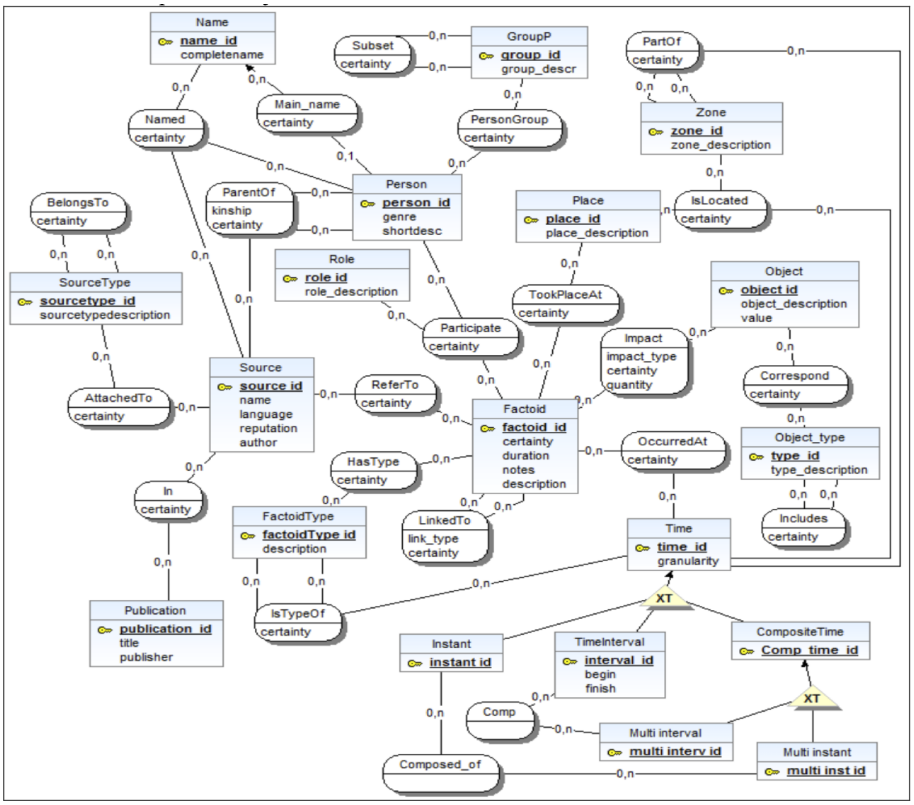
Very often, a piece of information cannot be recorded with plain confidence[29]. It is possible to systematically record the degree of certainty of an information in separate fields. My colleagues Stéphane Lamassé, Cédric du Mouza, Jacky Akoka et Isabelle Comyn-Wattiau have proposed a conceptual model based on factoids with a systematic record of the level of confidence: see the 'certainty' attribute present in all the relations between one factoid and other entities (figure 13).[30] This allows credible and doubtful informations to be sorted out, which is very important since, as I have said, the factoid model considers all facts given by sources, event the doubtful ones. They give the example of a query to find 14^{th} and 15^{th} century scholars who studied in Paris before getting the grade of doctor in Bologna, with a score of confidence for each information (figure 14).

Ultimately, their goal is to create algorithms that could discover automatically new information in a large prosopographical database, even if the data is fuzzy or contradictory.

[29] The issue is briefly addressed in Genet, 'Histoire, informatique, mesure', p. 10-11

[30] Jacky Akoka, Isabelle Comyn-Wattiau, Stéphane Lamassé, Cédric Du Mouza, 'Modeling historical social networks databases', *Hawaii International Conference on System Sciences* (2019), p. 2772-2781.

Figure 13: Documenting uncertainty in a conceptual data model



VI. Conclusion

This presentation is merely an introduction to relational databases and the historiography of the use of databases in medieval history. Every aspect of it deserves detailed developments. The rules of relational modeling have been formalised with the theory of normal forms, for instance. Other examples could illustrate many useful 'tricks' for modeling a database. New ways of modeling and opportunities exist today with graph databases or the availability of linked open data on the web. Last but not least, there is much to say about querying and analysing the data for historical purposes. The main goal of this presentation was to focus on modeling, because it determines what can be done with a database, and because it can actually change the way historians look at their subject.

Figure 14: Measures of score of confidence

Complete name	Birth date	Birth place	Confidence in the PhD place	Confidence in former studies	Reputation of the sources
Castellanus Nicolai de Bunarellis			0.5	0.8	0.6
Faustus andrelinus	1462	Forli	1.0	0.7	0.8
Bonaventura Badoer de Peraga	1332	Padoue	0.5	1.0	1.0
Laurentius de Bononia			0.8	1.0	0.6

VII. Addendum: Which software to use?

If the reader wants to create their own database, the first step is to design the conceptual data model. This requires no more than a pencil and a sheet (or more probably several sheets) of paper. Modifying the model once the base is implemented and the data have begun being recorded is possible, but it sometimes implies an extensive overhaul of the model, so it is best to think the model through beforehand. Various softwares are available to implement a relational database, each with their pros and cons.

The easiest solution is to use a standalone database management software such as *Microsoft Access*, *LibreOffice* or *OpenOffice Base*. They run on a personal computer, they provide a convenient graphical user interface, and the possibility to use forms to record and read data. The data is stored in a file, just like a text document or a spreadsheet. However, *Microsoft Access* only runs on Windows. *LibreOffice* and *OpenOffice* run on Mac and Linux as well, they are free and open-source, but they are not free of bugs and the interface lacks polishing, which can be frustrating sometimes.

The 'pure' canonical solution is to use a SQL server, such as MySQL. This can be done on one's own personal computer or a distant server. This can be the server of a university or an internet provider. This solution is very robust and powerful, but it is a little bit more complicated to set up and use for someone not familiar with computers. A client software like *MySQL Workbench* can ease things by providing a graphical interface to connect to a database. *LibreOffice Base* can also be used this way: it lets the user create and use forms with a distant database[31]. For more advanced users, or for those who can have the help

[31] See my video tutorial, in French: Octave Julien, 'Réaliser des formulaires de bases de données avec LibreOffice Base', *Médiathèque de l'université Paris 1 Panthéon-Sorbonne*, 2021, online: `https://mediatheque.univ-paris1.fr/video/2910-realiser-des-formulaires-de-bases-de-donnees-avec-libreoffice-base/`.

of someone knowledgeable, a SQL server lets users make the best use of their data: a website can be designed to access or even add data, it can be used in conjunction with a programming langage such as *Python* or *R*, or a geographical information software, to produce statistics and create maps [32].

Recently, two softwares, named *Heurist*[33] and *Nodegoat*[34], have been invented to help researchers create and publish a database. I have not personally tested them, but the feedback I have heard is very good. Both can be run on their own dedicated servers, or installed on a different one. Their web interface makes it possible to easily publish a database online, or to have a whole team of researchers working collaboratively on it. *Heurist* is completely free, while *Nodegoat* charges a fee for multiple user databases.

They implement a relational model, but their interface makes it easy to design one's database. It is also possible to use or adapt existing templates. *Nodegoat* and *Heurist* propose built-in solutions for chronological or spatial data, and they can reuse open linked data available elsewhere on the web. Such data can easily be represented with maps, timelines or social networks. The only downside is that you do not have the full control of the database and you depend on external infrastructures.

[32] My database of miscellanies manuscripts is used this way: it is accessible on www.octavejulien.fr (in a version that needs an upgrade) and it was used in conjunction with *R* to do statistics.

[33] `https://heuristnetwork.org/`

[34] Pim van Bree, Geert Kessels, nodegoat: a web-based data management, network analysis and visualisation environment, online: `nodegoat.net` (accessed April 12, 2024).

Contents