A Little Structure In Your Codes Will Make Your Research A Lot Easier

by Nicolas Woolf, Ph.D., Director of Woolf Consulting

Nick Woolf is the founder of an independent training company devoted exclusively to ATLAS.ti and has provided training and consulting in ATLAS.ti since 1997. His workshops (available throughout North America or in refresher web conferencing sessions) emphasize learning ATLAS.ti in the context of designing and conducting a data analysis. Nick conducts his own qualitative research programs and also provides on-going assistance and project management to individuals and research projects. More information is available at www.learnatlas.com.

In our Best Practices section you find detailed descriptions and explanations on how to get the most out of working with ATLAS.ti. Get the inside scoop here and learn the latest information and tips that haven't been included in the hand book yet.

How To Use ATLAS.ti's Tools To Bring Structure To A List Of Codes (Part 1)

Microsoft Word is not a writing program. There is no button for "short story" or "non-fiction" or "essay". Word is basically a character display program with a lot of bells and whistles. Similarly, ATLAS.ti is not a data analysis program. You do not give it qualitative data and press an "analyze" button. ATLAS.ti is fundamentally a concept database. You create and enter names of concepts, or "codes", to be used for conceptualizing chunks of data, and the program lets you organize and relate these concepts in ways that support your analysis. The great strength of ATLAS.ti is that it does not provide pre-determined ways to do this, so every analysis must be designed uniquely according to its needs. Unfortunately, this leaves many people not having any idea what to do with the program once they have created a long list of codes.

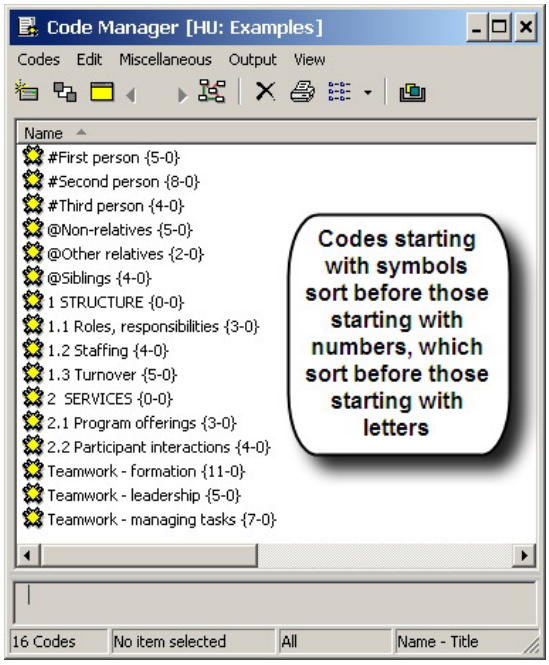
I recommend you bring some structure to your ATLAS.ti code list, and there are four ways to do this. These are not four alternative ways to do an analysis, but generic building blocks to use and combine as needed. The key piece of magic to unlocking the power of the program is recognizing that you do not first figure out the program's tools and then base the analysis on how the tools work. Instead, you decide what you want to accomplish at each step, and then figure out how the program's building blocks can be combined to pull that off. For purposes of bringing structure to the codes there are four building blocks: code prefixes, families, networks, and supercodes.

All four serve to bring together groups of related codes. The key to selecting and using these building blocks is to understand the concept of "related codes". From the perspective of the analysis itself, rather than the program, codes are related based on their evolving meanings in relation to the research question. But for the secondary perspective of using ATLAS.ti powerfully, codes are related based on the different purposes of relating them. These can be (a) for organizing or "housekeeping" purposes, or (b) for analytical purposes, to directly move the analysis forward. Housekeeping refers to mechanical or administrative or tidying up tasks, like putting codes into a grouping such as an ATLAS.ti family, so you can select them with one click as group, rather than finding them one at a time. An analytical purpose would be linking codes together with named relationships because they form a sequence of logic, or part of a developing conceptual framework. This way of thinking keeps you from going down blind alleys when you try to use housekeeping building blocks for analytical purposes. There are always exceptions, but code prefixes and families generally serve housekeeping purposes, supercodes are used for both housekeeping and analytical purposes, and networks are most often focused analytically on moving your thinking and your project forward.

In this issue's article, I will introduce the first to ways of structuring your codes, using code prefixes and families. In the next issue you will read about structuring your codes using supercodes and networks.

Code Prefixes

The most basic organizing technique is to give some groups of codes a common prefix or root, using either letters, numbers, or symbols.



ATLAS.ti does not recognize the prefixes of codes other than for sorting them. There are two housekeeping purposes for using prefixes. First, codes with the same prefix sort together as a convenient cluster rather than being distributed alphabetically among unrelated codes, and they are thus easier to find. Second, a code's prefix reminds you what kind of code it is, which may not be obvious from its name. Here are some examples from the accompanying Code Manager.

USING LETTERS

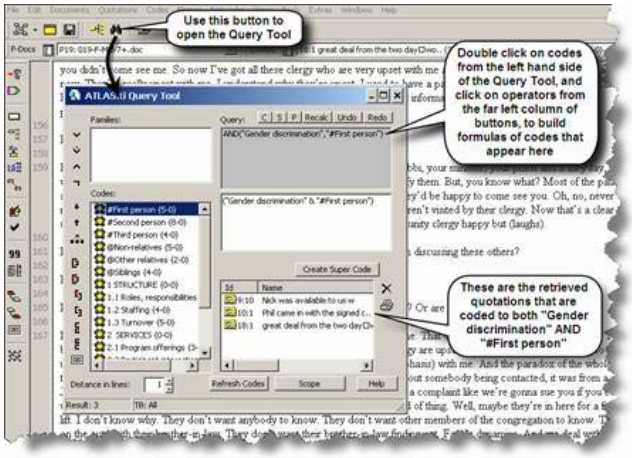
The three codes beginning with the word “Teamwork" distinguish three “flavors" of the concept, and are easily found together in the ‘T's without remembering the names of each flavor. I use prefixes in this way when there is really a single idea or concept with several fine differences of meaning that have not yet emerged as separate main ideas. However, I do not use such prefixes to represent layers of a hierarchy in long code name, like:Organization issues/Structure/Staffing/Gender balanceOrganization issues/Structure/Staffing/Gender relationsCode names and hierarchical organization are generally in flux throughout a project, so this style of naming codes is generally too cumbersome and inflexible. Other ways may be better.

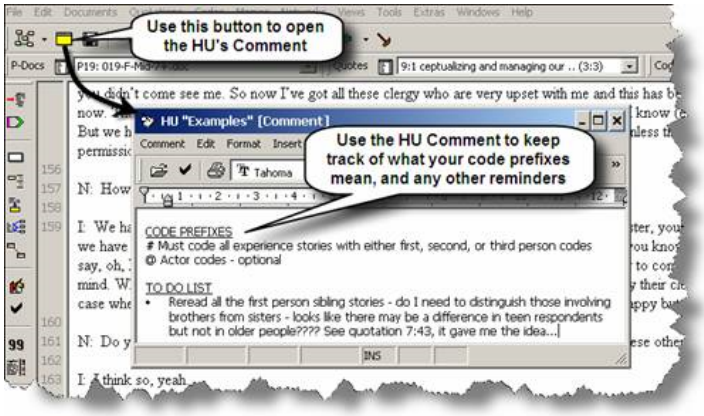
USING NUMBERS

The codes beginning with numbers sort in an orderly hierarchy, but are also inflexible, requiring renaming of many codes whenever we want to adjust the hierarchy. I only use numbered codes in very structured projects where there will likely be no further development of codes and concepts, or sometimes when the analysis is complete and it is convenient to rename codes with numbers so they sort in the order that the writing will follow.

USING SYMBOLS

While I use letters and numbers infrequently, I use symbols in every project to distinguish special groups of codes. In the example Code Manager, I used the three codes beginning with “#"in one study to distinguish experiences described by interviewees that they themselves had experienced, from those that their friends had had, and from those they were merely speculating about. In each case I coded their responses using the same set of conceptual codes, but I additionally coded each quotation with one of these special codes. Then later, using the Query Tool, I could retrieve all quotations coded to say “gender discrimination", but only for quotations also coded to “#First person", or to “#Second person", depending on what I was looking for (see example Query Tool.) I used a second group of codes beginning with “@"to identify actors involved in the reported experiences. In this case their use was optional, depending on whether other actors were involved in the stories. But in the case of the “#" codes they were required for every quotation, to ensure that the Query Tool results produced a complete set of first, second, or third person experiences. I keep track of the purpose and usage of these different groups of codes in the “HU Comment", along with other reminder information, which I pop up and read before each working session (see example HU Comment.)





Families

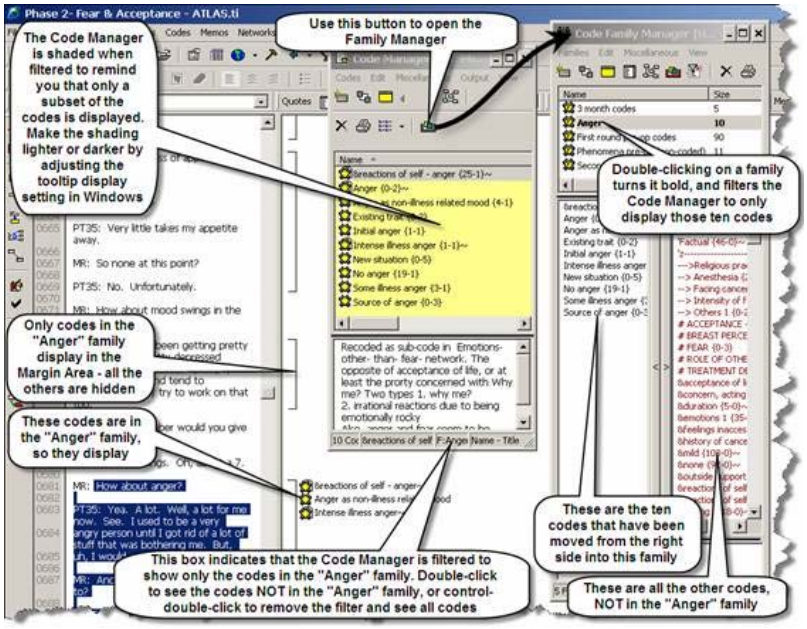
Families are the second way that code lists can be structured or organized. A family sounds like a concept, a grouping of things that are alike or related in some way, and because working with concepts is the main activity in qualitative research, many people assume that ATLAS.ti families are an important analytical tool. But because of the way ATLAS.ti families works it is more useful to think of families as a housekeeping or support tool. It is helpful to think of families as the "filtering tool".

When codes are grouped together into a code family, two things are possible: (a) the family can be used as a filter, so that only the codes in the family are visible, with all other codes hidden; and (b) the family can be used as a short cut, allowing you to operate on a group of codes all at once. It is generally best not to think of a family as the way to represent a new, bigger concept in its own right, as this tends to lead down blind alleys - new, bigger concepts built out of other codes are usually better represented by supercodes or in networks, as I will discuss in the second article on structuring codes in the next newsletter.

Here are just a few examples of housekeeping purposes for families.

FILTERING THE CODES MANAGER

Use code families to filter the codes, so that only a subset is visible, rather than having to scroll through the whole list in order to find the codes you are currently interested in. Sometimes a project naturally falls into two parts, and you can add half the codes to one family and half to another, and then flip back and forth between seeing each subset of codes in order to focus one or other aspect of the project. Or, create an ad hoc family to see a group of related codes for a particular purpose, as in the illustrated example in which the HU is filtered to show only the ten codes that have something to do with one particular emotion. Then delete the family when that task is completed.

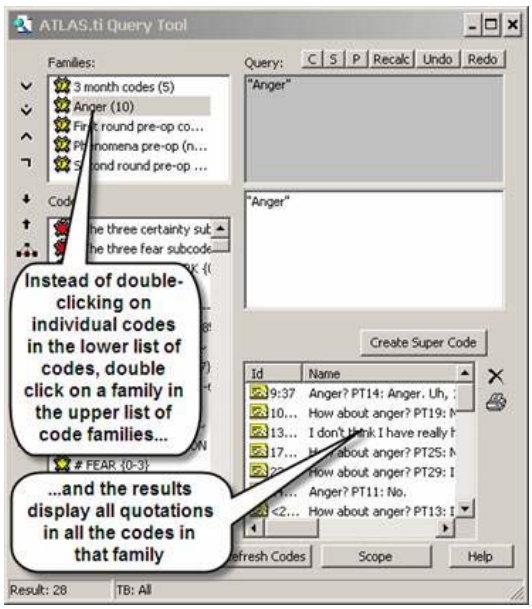


FILTERING CODES AND PD's

Families are useful if you use the Codes-Primary Document table to output coding frequencies for each document to Excel. Each column in the table is a code, and each row a document. Create a code family for the codes that you wish to have columns in the table, and filter by this family. You can also create a family of PD's in a similar way, in order to limit - i.e. filter - the number of documents for which you wish to have rows in the table. The outputted table will then only have those rows and columns.

SHORT CUT

Code families offer a short cut by allowing you to select a group of codes all at once in the Query Tool instead of having to select all its individual codes. A useful technique in the Query Tool is to make a supercode of a family of codes. As new codes are added to or removed from the family, the supercode automatically reflects the updated set of codes that are members of the family.



Be sure not to miss the second part of this article in next issue of INSIDE ATLAS.ti - Your Quarterly Newsletter. You will find out how to structure your code lists using networks and supercodes.