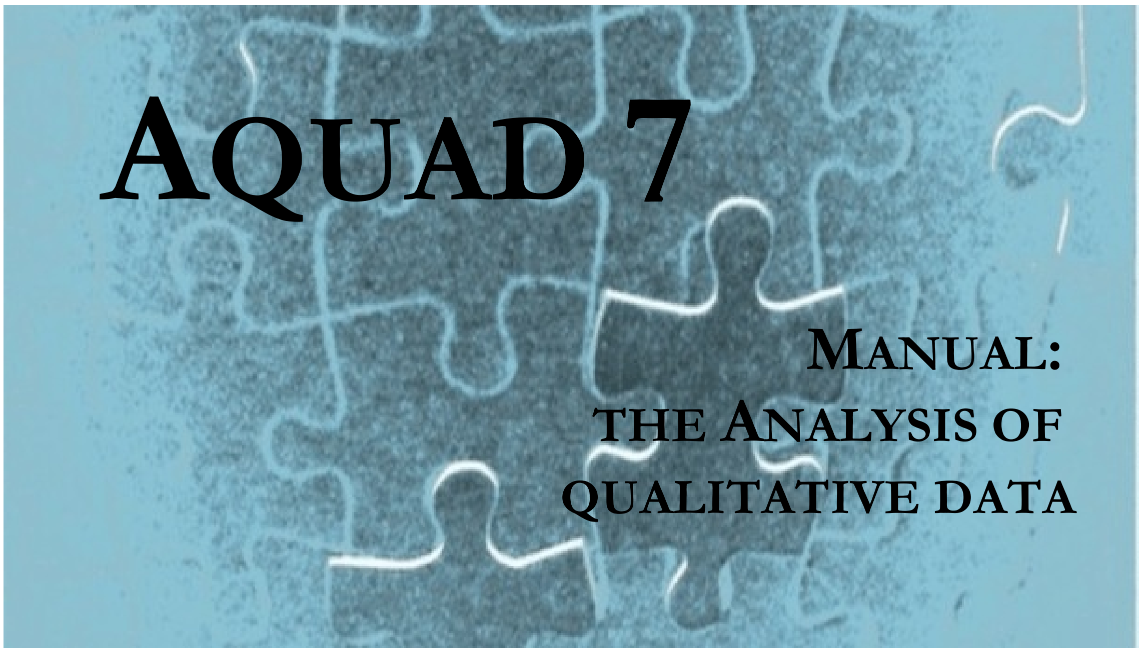


AQUAD 7

MANUAL: THE ANALYSIS OF QUALITATIVE DATA

Günter L. Huber
Leo Gürtler

Softwarevertrieb
Günter Huber

Copyright Notice:

Warranty Notice, GNU General Public License, paragraphs 15 and 16:

1. Edition 2013

Authors: Günter L. Huber and Leo Gürtler

Content

Introduction

The first version of AQUAD (**A**nalysis of **Qua**litative **D**ata) was developed in Germany in 1987 to compensate for a shortage in manpower in a research project. At this time several programs for qualitative analysis had been in existence for a number of years. Simpler types of software tools were using the search function of available word processors or database programs. Some were already designed for the special demands of qualitative analysis, although they did not offer more than simple counting and retrieval functions. A much higher level of functionality was realized just by one program, the American software package Qualog written by Anne Shelly and Ernest Sibert for a mainframe computer. This program made use of the rich potential of so called "logical programming", and it was the stimulus and model for AQUAD.

A particular goal in developing AQUAD was to support the reconstruction of linkages of meaning within the data base as described, for instance, by methodological approaches like "grounded theory" (Glaser & Strauss, 1967). In addition, AQUAD tries to incorporate methodological ideas for text analysis like Miles and Huberman's table or matrix analysis (Miles & Huberman, 1994) and Ragin's comparsion of configurations of meaning (Ragin, 1987). The more elementary functions of other programs like managing code entries and retrieving coded text segments are of course available in AQUAD, too.

The actual version 7 of AQUAD allows for the first time to analyze qualitative data without costly and time consuming transcription of the initial data base. Of course, AQUAD still offers rich possibilities to analyze written texts. Altogether, the following types of data files can be imported into the program and analyzed directly:

- Texts formatted as *.txt (plain ANSI format) or *.rtf;
- audio recordings as sound files formatted as *.wav or *mp3;
- video recordings formatted as *.avi (in various compressions);
- graphic files (pictures, drawings, photos, etc.) formatted as *.jpg.

The logic of data handling and internal processing of codes is almost the same for all of these data types. Therefore, users do not experience difficulties when switching from one data type to another in their projects. Above all, there are no fundamental differences of operating the program between version 6 and version 7. Available files from projects run under AQUAD 6 can be converted without any problems for further use in AQUAD 7.

In all qualitative analyses following the "coding paradigm" the main task is to reduce the usually wordy and redundant descriptions, explanations, justifications, field notes, protocols of observations, etc., that comprise the researcher's data texts to some kind of systematic description of the meaning of the data. The procedures used to accomplish that purpose have been very individualistic and, naturally, dependent on the ultimate purpose of the type of qualitative research at hand. However, one invariant can be found in all but the most artistic / phenomenological types of analysis: the classification or categorization of text passages (or scenes in videos). Categories can be thought of as "containers" for organizing data according to their meaning. Either the reduction of the data is carried out deductively with an *a priori* designed system of categories, usually developed from existing theories; or categories are established on the basis of central research questions, hypotheses, or important constructs that have been introduced earlier during the collection of data (see Glaser & Strauss, 1979). Inductively, they can be created in an initial perusal process with a sample of data texts, or as a reconstruction of the subjective "categories" used by the research subjects and in this way they faithfully mirror the subjects' view of the matter. AQUAD supports the deductive as well as the inductive process, and also a combination of both.

In AQUAD 7 the possibility of qualitative analysis as "sequence analysis" in accordance with the approach of objective hermeneutics (Oevermann, 1996; Oevermann, Allert, Konau, & Krambeck, 1979) is added in a specific module. Unlike many other approaches, a sequence analysis does not start from an overview on the complete text and subsequently searching and coding those data segments, which are significant under the perspective of the research question(s). Instead, in a first phase of generating hypotheses the researcher notes for each text segment all possible meanings s/he can imagine. Generating hypotheses is a strictly sequential process, data segment after data segment. Once the researcher is convinced that s/he has generated all possibly meaningful hypotheses by scrutinizing the text segments presented up to this point, the remaining text is used to confirm or falsify these hypotheses. During this second phase, the remaining text is used in a non-sequential manner to find reasons for keeping or rejecting the hypotheses generated during phase 1.

The special characteristic of the coding approach in AQUAD is its ability not only to categorize and then to assemble the data for each category, but to allow the researcher drawing conclusions by relating categories to each other, i.e., by exploring the occurrence of typical and repeated configurations of category representations in the data. Once such repetitions are surmised, a researcher may wish to confirm the hunch by systematically surveying the data or, with Miles & Huberman (1994), Shelly & Sibert (1985) and Shelly (1986) by "testing the hypotheses." A positive outcome (the claim that particular combinations of statements show up in the texts in a systematically combined form is found to be "true") would validate the conclusion.

The configurations mentioned above can take on various forms. For instance, during the development of AQUAD the program was used for research studies in which scholars explored sequences, clusters, correlational, and hierarchical or dimensional structures. (Examples will be provided later in this manual.) AQUAD contains special facilities that allow the user working with all of these forms of "hypotheses." In addition, the user may postulate other types of hypotheses just by clicking on codes and logical links in a list and use AQUAD to test them.

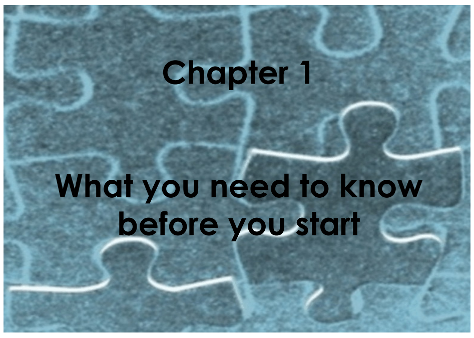
This is the point where the computer ceases to be merely a useful convenience and becomes an essential tool for qualitative analyses. It is extremely time-consuming to locate all combinations of occurrences manually, and it is impossible to guarantee accuracy and completeness. Only the computer can do the latter. The human mind would be vastly overextended by it. Consequently, researchers can now pursue avenues of analysis that were previously closed to them. The computer has helped to change the ways qualitative analysis can be done, and AQUAD is a pioneer in the process.

The place of strongest impact of the computer may be in postulating causal relations. Since causal hypotheses have been the main concern of experimental and statistical procedures, they have been shunned in qualitative research. Ragin (1987) reminds us, however, that qualitative causal analysis has a tradition that goes back to John Stuart Mills (Ragin, 1987, p. 36f.), and that its methods may be superior to quantitative analysis in certain respects, since generality is not given precedence over complexity (Ragin, 1987, p. 54). A detailed treatment of Ragin's argument is beyond the scope of this manual (the reader is referred to Ragin's book, 1987), but the method itself, called the "Boolean method of qualitative comparison" by Ragin, is explained later in chapter 13. This qualitative comparative method integrates – not simply combines – features of experimental and interpretational design by treating the existence of a certain "condition" in a set of data that represents one "case" (the occurrence of a code signifying a category) as a dichotomous categorical variable. The evidence or "condition" either exists or does not exist in a given piece of data. Causes are always seen as complex combinations of conditions that are associated with a particular "outcome." All data are searched for the presence and absence of all forms of possible combinations and the results are entered into a table, with each cell containing either a zero or a one, signifying the absence or presence of the condition. Using algebraic procedures developed by the mathematician Boole, called "combinatorial logic", "minimization," and the use of "implicants," conclusions are drawn from the table about the one or several combination(s) of conditions that result(s) in the occurrence of the outcome being investigated. AQUAD contains a separate program module that facilitates this Boolean method of qualitative comparison.

AQUAD, in summary, is a program for the generation of theory on the basis of qualitative data. Since theory-building and hypothesis-testing have been traditionally the domain of researchers who work with quantitative data, theoretical notions based on qualitative data are easily distrusted. Although we acknowledge (and have no desire to claim otherwise) that qualitatively developed statements do not achieve the same degree of generalization as statistically tested statements, it is important to make sure qualitatively developed conclusions are based on as rigorous a verification process as possible. Therefore, special emphasis is placed in AQUAD on objectivity, reliability, and validity. The researcher is encouraged to use procedures such as the repeated interpretation of the same text by the same analyst or by different analysts over time, or to pay attention to issues of internal validity such as whether the categories have been used consistently, whether their defined range of meaning has been maintained, whether the meanings represented by specific categories indeed correspond to the content of the text passages that are sorted into them, etc. (Huber, 1989). More about these issues will be presented later in this manual.

Finally, two more attributes of AQUAD deserve attention. Unlike many other qualitative analysis programs, AQUAD supports some versions of conventional content analysis or linguistic analysis by allowing the user not only searching for words and phrases that occur in the data text and examine their frequency, but the program can extract words together with their context (key word in context – KWIC – indexes, see Weber, 1985). Furthermore, AQUAD provides for the attachment of researcher memos to text segments.

Principally empirical research follows a path towards discovery that starts from descriptive or categorial analyses and leads via postulating or observing regularities to statements, which explain these connections at least tentatively. No matter whether an analysis results in a taxonomic, correlative or causal order of the phenomena under study, the process of research is focused on reduction, that is reducing concrete details and moving to higher levels of abstraction and generalization. We will be able to see the essentials, if we abstract away "... all coincidental elements of the real world" (Galtung, 1980, p. 98). AQUAD tries to contribute to this goal, but is at the same time able to keep open the way back to the manifold, concrete, and colorful details of the original data base.



Chapter 1

What you need to know before you start

1.1 AQUAD and your computer

All necessary technical information is provided in chapter 2 of this manual. How to prepare texts written in any text processor (for instance, Word or WordPerfect) for the analysis in AQUAD is described in chapter 3. Very detailed instructions how to prepare multimedia data (pictures, audio or video recordings) are compiled by Leo Gürtler in chapter 4. The most important things for you to know at this point have to do with whether or not your computer is able to accommodate AQUAD. Here are its basic requirements:

(1) Your computer must be equipped with Microsoft's Windows XP/Vista/Windows 7..

(2) Your computer must have a hard disk with approximately 90 MByte free disk space.

(3) The program comes on CD-ROM. Your computer must have the appropriate drive.

You must install AQUAD on your hard disk before you can use the program. How to install AQUAD is described in chapter 2.

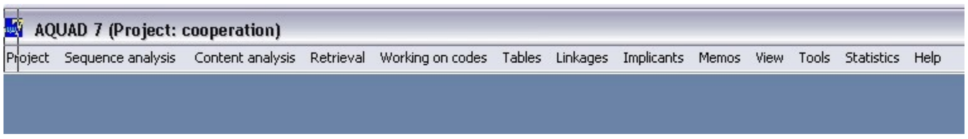
1.2 The structure of AQUAD

1.2.1 Modules and menus

AQUAD offers you menus that list the various things you could do, and you choose the one you want by clicking on it with your mouse or by pressing the Alt-key together with the underlined letter of the function you want.

There are two ways in which to navigate from one menu to another. These two have to do with the distinction of "main menu" and "sub-menu." The main menu corresponds to the major components of the program, called "program modules." They form the basic "architecture" of the program. Their names are present in a headline on the screen when you start AQUAD or after you terminate your work within one of its modules.

You proceed very easily from the main menu to its sub-menus: You place the mouse pointer, usually an arrow, on the choice of action you want the program to perform, and click the left mouse button or you press the Alt-key and the underlined letter together, and the appropriate sub-menu will appear. Here you see the main menu:


AQUAD 7 (Project: cooperation)
Project Sequence analysis Content analysis Retrieval Working on codes Tables Linkages Implicants Memos View Tools Statistics Help

The module "Project:"

The module *Project* gives you an opportunity to determine some "default" parameters for the project you are just working with, i.e., to "define" your project. The function "Define project" asks you to tell the program whether you are going to work with texts, pictures, audios or videos as data base. You click on the appropriate type of data files (see below) and select in the next window all files, which you want to analyze in your project.

The function "Open project" will be used only, if you work with more than one project at the same time and you want to chance the actual settings. Otherwise, AQUAD opens automatically during each start the project that was opened the last time.

With the function "END" you terminate your work and exit from AQUAD.

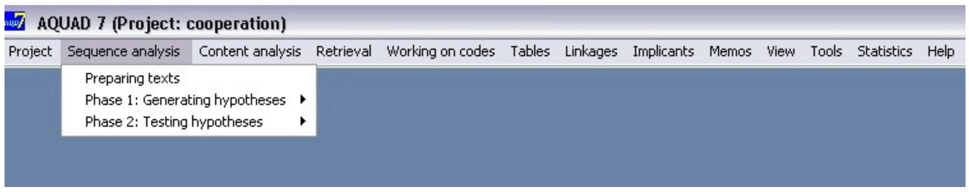
The modules "Sequence analysis" and "Content analysis:"

There are various approaches to the analysis of qualitative data, which can be grouped according to their main characteristics in two main forms, namely "sequence analysis" and "content analysis":

A *sequence analysis* is a specific way of analyzing qualitative data that does not start from an overview on the complete text, video, etc. and then looking for and coding data segments, which are significant for the research question. Instead, the researchers note during the first phase of *generating hypotheses* step by step for each data segment all possible meanings they con think of. Depending on the preparation of data, in the case of texts AQUAD presents as segments complete sentences, parts of sentences (each time until the next punctuation mark) or segments containing a determined number of words. Generating hypotheses is realized strictly in sequential order, on data segment after the other. Once a researcher is convinced that the segments s/he has interpreted up to this point have evoked all hypotheses significant in the context of the actual research question, s/he uses the remaining data to confirm or falsify these hypotheses. During this phase of *confirming hypotheses* the researcher is looking non-sequentially in the remaining text segments for reasons to keep or to reject hypotheses generated earlier.

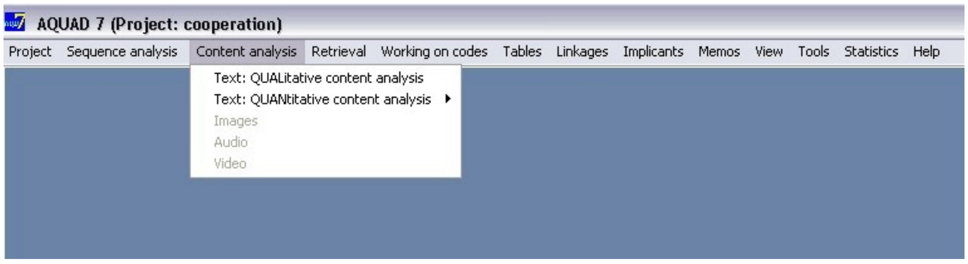
Since the beginnings of a broader reception of text analysis in the social sciences, content analyses are realized either as a quantitative or a qualitative method. In the same year, 1952, two trendsetting articles were published by Berelson and by Kracauer. While text analysis serves for Berelson to assess systematically, objectively, and quantitatively the *manifest* contents of communication, qualitative content analysis according to Kracauer tries to reveal the categories of meaning hidden or *latent* in the data.

The module "Sequence analysis:"



First, the imported data are structured into segments, for which then the researcher generates hypotheses about all possible meanings in strict sequential order. Finally these hypotheses are compared critically with further data segments and confirmed or rejected.

The module "Content analysis:"



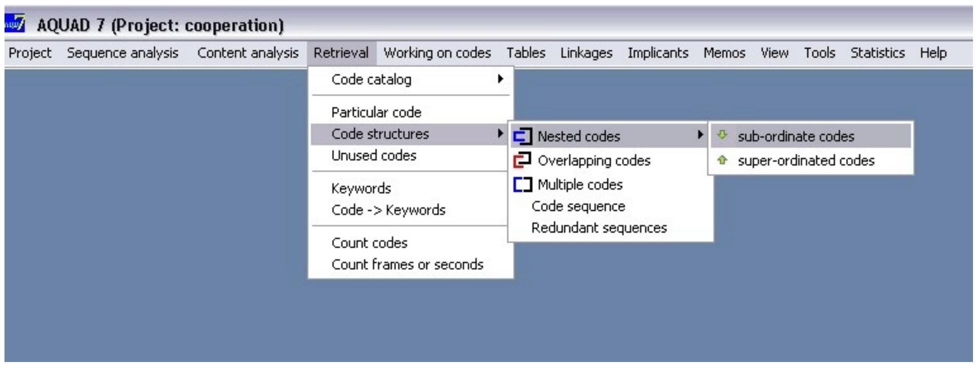
The content analysis of texts follows either a qualitative approach, that is an approach according to the coding paradigm, or a quantitative approach (see above).

Qualitative content analysis:
Following Miles and Huberman (1994) we differentiate in a qualitative content analysis two processes linked to each other, the processes of reduction and conclusion. Depending on the methodological orientation data *reduction* is a more or less strictly structured procedure. Common ground of these procedures is the classification or categorisation of data segments. According to the terminology of Miles and Huberman (1994) we will name the symbols used for characterizing these categories "codes", correspondingly the process of reduction is called "coding" of data. When drawing conclusions we will try to detect typical and/or repeated configurations among the codes. For this end we have to pay attention to patterns of sequence configurations, trying to find super- and subordinated codes, merging similar codes in super-ordinate categories ("meta-codes"), viewing codes under the perspective of opposite poles on a common dimension or generate and checking hypotheses about causal relations between coded data segments.

Quantitative content analysis:
Quantitative content analysis is focused on manifest characteristics of texts, that is particular keywords, idiomatic expressions, metaphors, etc. The researcher has to find and count these characteristics. Interesting are only directly accessible, openly given text elements. Of course, the limitation of quantitative content analysis to just counting manifest elements of content is only partially true. First the critical elements have to be determined –

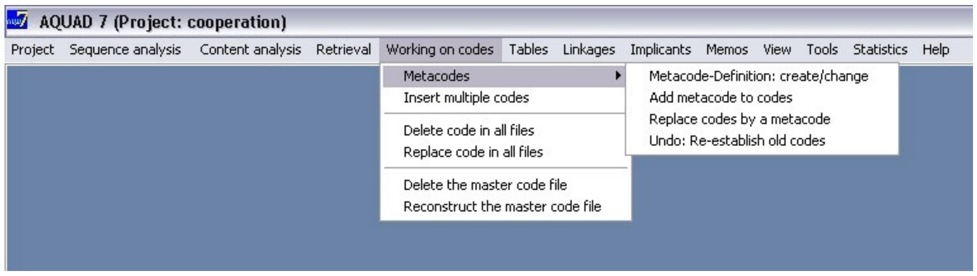
and this process depends necessarily on assumptions that exactly these directly accessible elements are related to more or less hidden meanings in the text. In other words, quantitative content analysis infers latent contents from manifest, observable content elements, which themselves are determined qualitatively at least in the beginning.

The module "Retrieval:"



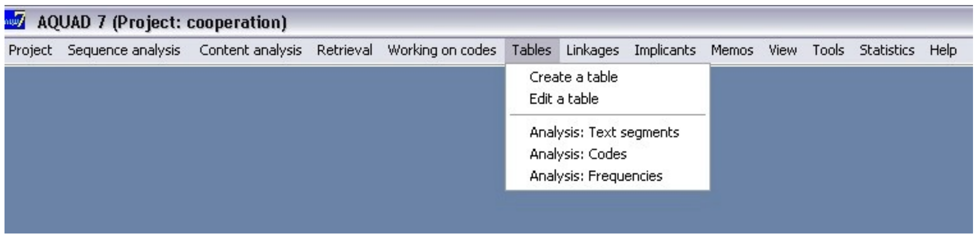
The *R̲etrieval* module allows you writing a list called a catalog of all those codes of particular interest in your data. AQUAD will use every single code in a catalog when you choose an adequate program function like retrieval of code sequences (for instance, code overlaps or hierarchical nestings) or counting of code frequencies. The functions of this module allow to retrieve particular codes and the associated data segments. We can look for determined code structures, that is whether a specific code appears su-ordinated or super-ordinated to other codes, whether its data segment is overlapping the segments of other codes, whether a data segment was coded multiply with more then one code, whether a code appears repeatedly in a particular sequence together with other codes an whether these sequences appear repeatedly. The data can also be analyzed looking for codes, which were *not* used for interpretation. By entering keywords we can find those data segments that contain probably a specific meaning – thus a sort of semi-automatic coding is possible. On the other hand, meaningful application of specific codes can be checked by looking for the use of these codes in data segments, which contain also critical keywords. Finally, the frequency of codes can be counted as well as the length of particular scenes (defined by codes) in video- or audio-recordings. The code catalogs mentioned above allow to apply all these functions to several codes at the same time.

The module "Working on codes:"


AQUAD 7 (Project: cooperation)
Project
Sequence analysis
Content analysis
Retrieval
Working on codes
Tables
Linkages
Implicants
Memos
View
Tools
Statistics
Help
Metacodes
Insert multiple codes
Delete code in all files
Replace code in all files
Delete the master code file
Reconstruct the master code file
Metacode-Definition: create/change
Add metacode to codes
Replace codes by a metacode
Undo: Re-establish old codes

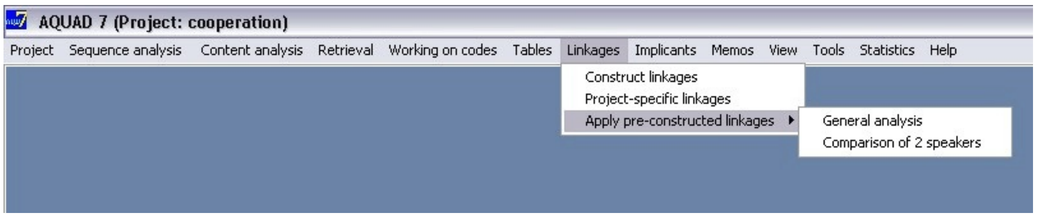
The code entries in code files can easily be corrected, added to, or modified in AQUAD. Furthermore, the organizing system itself can be restructured by combining particular categories, thus creating "meta-codes". Additional codes can be added later automatically to particular codes, for instance the control code "$do not count" could be added to a code "Interviewer"; thus, the text parts of the interviewer will be excluded from quantitative analysis. Poor or incorrect codes can be deleted from all code files or replaced by a more appropriate (or correctly typed) code. During coding AQUAD performs a sort of bookkeeping and lists in a master code file, which codes were applied how many times in a project. In case there appears an error in this list, for instance by externally working on code files, the master code file may be deletedand automatically reconstructed correctly from the available code files.

The module "Tables:"


AQUAD 7 (Project: cooperation)
Project Sequence analysis Content analysis Retrieval Working on codes Tables Linkages Implicants Memos View Tools Statistics Help
Create a table
Edit a table
Analysis: Text segments
Analysis: Codes
Analysis: Frequencies

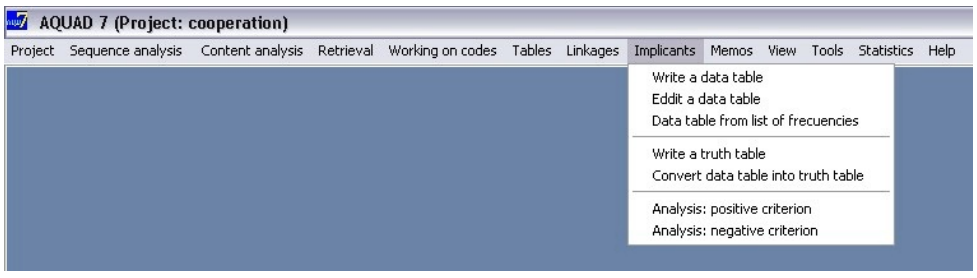
This module supports two-dimensional retrieval strategies. Sometimes you may want to find text segments of a particular code only in those texts for which another critical code is valid. For instance, if one of your codes is "male" and another is "female", a third is "vacation", and a fourth is "work", you can tell the program to make a matrix with two rows and two columns. The cells will be filled out with the segments of text from your data in which men talk about vacations, then those in which they talk about work, then with the text containing women's opinions about vacations, and then their opinions about work. The second analytical option (Analysis: Codes) will show only the positions of codes in your data, the third option (Analysis: Frequencies) shows – for a fast overview – only the frequencies of codes in the cells of the matrix.

The modul "Linkages:"


AQUAD 7 (Project: cooperation)
Project Sequence analysis Content analysis Retrieval Working on codes Tables Linkages Implicants Memos View Tools Statistics Help
Construct linkages
Project-specific linkages
Apply pre-constructed linkages
General analysis
Comparison of 2 speakers

This is one of the two modules most important for theory building in AQUAD. It allows you specifying certain meaningful interrelations of text segments by formulating linkages of more than just two codes in your data files. Then you let the program check whether these linkages occur in the database or not. That is you have to formulate a hypothesis about two or more codes appearing regularly together, for instance: "Whenever the interviewees mention topic A, they also begin to speak about B or C". This function will check, whether this linage of codes appears in your data. You can select pre-constructed linkage structures and enter your codes like "variables" or you may construct yourself linkages of up to five codes using the logical operants "and," "or" and "not."

The module "Implicants:"


AQUAD 7 (Project: cooperation)
Project Sequence analysis Content analysis Retrieval Working on codes Tables Linkages Implicants Memos View Tools Statistics Help
Write a data table
Eddit a data table
Data table from list of frecuencies
Write a truth table
Convert data table into truth table
Analysis: positive criterion
Analysis: negative criterion

The analysis of implicants was mentioned briefly in the introduction and will be explained later in detail in chapter 13. It applies the procedure of "logical minimalization" to a complex comparison of configurations of codes appearing in the database. One of these codes is assumed to be an "outcome" of a configuration of the codes – which in turn are assumed to represent potential "conditions" or "causes." The result makes sense only if the analysis can be based on a greater number of cases. The basic data table can be written and edited or created by the program from a list of code frequencies (see module "Working on codes"). This table must be transformed into a table of "truth values" (yes/no; true/false; 1/0). A truth table can be also written manually. In the process of analysis one of the hypothetically relevant codes is selected as positive or negative criterion and the program then investigates all cases, in which the criterion appears as "true" or "false" and shows the configurations of the other codes, which can be found under this condition.

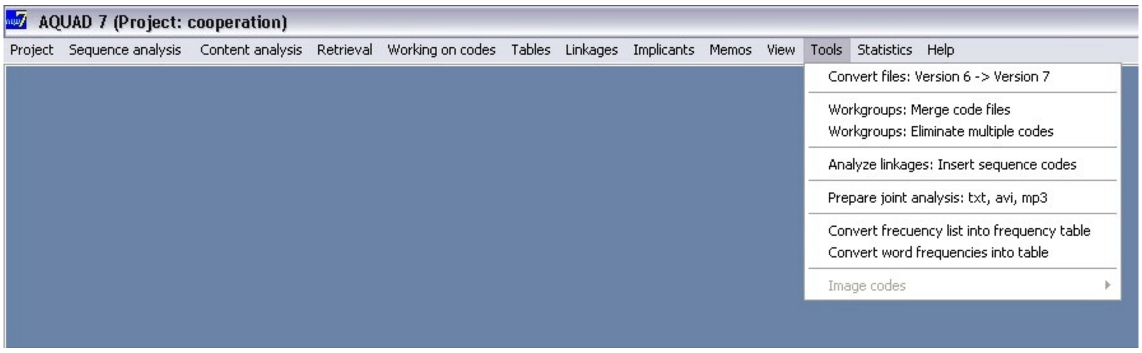
The module "Memos:"

This module supports the advice of Glaser (1978) to note immediately everything that crosses your mind while interpreting a text. Maybe you get a hunch about missing codes, about relations, contradictions, potential exceptions from your coding rules, etc., but you cannot elaborate on that particular idea at the moment. Writing memos is a good compromise if you are confronted with the alternatives either to forget some really important ideas or to lose the thread in your process of interpretation. In AQUAD you may relate your notes to critical characteristics of your data. Thus, retrieval of memos and the associated data segments is very reliable. When you need your memos again, you select an appropriate retrieval criterion. In case you should have forgotten all of a memo's critical markers, you may at least remember the one or the other characteristic word from the memo's text; then you can use the function for memo retrieval by key words.

The module "View:"

The module *View* gives – as may be expected – an overview on text files together with codes attached to the beginnings of text segments in an "outline" format, which you know, for example, from file lists in WINDOWS dialogues.

The module "Tools:"


AQUAD 7 (Project: cooperation)
Project Sequence analysis Content analysis Retrieval Working on codes Tables Linkages Implicants Memos View Tools Statistics Help
Convert files: Version 6 -> Version 7
Workgroups: Merge code files
Workgroups: Eliminate multiple codes
Analyze linkages: Insert sequence codes
Prepare joint analysis: txt, avi, mp3
Convert frecuency list into frequency table
Convert word frequencies into table
Image codes

This module is interesting for users of version 6 of AQUAD, for workgroups and for researchers, who want to export frequency lists for further quantitative analysis. If you want to continue in AQUAD 7 a project started under version 6, you can convert your available code files into the new format of version 7. Workgroups, whose members are analyzing the same files, will find a tool to merge their codes into one file and remove multiple codings (for instance, because each team member necessarily applied the same "speaker codes" to mark the same data segments).

The module "Statistics:"

For checking the statistical significance of the distribution of codes or keywords in frequency tables AQUAD 7 offers the computation of chi-square values. The program computes chi-square for tables with 2 * 2, 2 * m, k * 2 and k * m cells. There will appear a warning, if the number of cases and/or too many expectancy values are too low.

The module "Help:"

The module *Help* offers access to general information about AQUAD via a list of contents and via a list of key words. Principally, it functions like all help modules in software developed for WINDOWS. In addition, most AQUAD windows contain a specific help button in a button panel on the right end. This button opens a selection of help topics which may be useful in that particular situation of your work.

1.2.2 Individual files and file catalogues

When you start AQUAD, as a default it always loads the project setup and the "catalog" (list) of files you have last been working with. After the installation, the setup of example projects using interview texts ("interview"), audio recordings of the same texts ("a_inter"), video recordings of a playing baby ("baby") and photographs of orchids ("blossoms") are already available. Normally, you would work on only one project, and as one of the first steps of working with AQUAD, you will make a list of all data files you are analyzing in that project and give the list a name. This is done in the module *Project*. Of course, you may add new files to the catalog or do any other editing any time you need to (more on this in chapter 5). You also may make as many catalogs as you want (also chapter 5).

Usually, a researcher would make more than one catalog even when only one research project is being worked on at the same time. For coding, you pick a single file from the code catalog. But if you have reached a stage in coding where you want to run your first analyses, for instance, you want to retrieve nested codes, the program takes *all the files* listed in your code catalog and analyzes them one after another automatically. Now imagine, you have 100 text files, but at the moment you are interested in nested codes only in 10 of them. Or you just want to try an analytic heuristic on a sample of files and not waist much time by waiting for all 100 files to be analyzed. No problem. You create a separate file catalog in "*Define project*", enter a different project name and select from the list of file names those few ones which you need for your actual work. Later you can enlarge the selection using the same project name again. The results of your coding (your "code files") do not get lost, but are available in any project, for which the corresponding data files were selected.

1.2.3 Paper print-outs and result transferability

Since you will want to look at the results of your work with AQUAD at times other than when you actually are in the program, AQUAD provides two options for you. You can either have the results of your work printed out on paper, or you can have them stored as an ANSI text file, i.e., a file that can be imported into your word processor, or other programs that deal with text and have import-capability.

For *printing on paper* you push the right mouse button once the results of an analysis are visible on the screen. A small menu window will pop up, from which you then select the option *Print*. In case you want to change fonts, font size, the layout of printed pages (size, margins) or select a particular printer – if more than one is available from your computer – choose the appropriate option in the function bar on top of the screen by clicking on the corresponding icon.

For *saving results in ANSI files* the procedures are similar to those for paper copies. Of course, you always select the *Save* option from the pop-up menu first. This opens an additional window, where you enter a file name for saving this file (the path for result files was already determined during the installation of AQUAD, see above). After pushing the 'OK'-Button all the results present on your screen or a marked group of result lines are copied to the file you just defined.

Result files are saved in plain ANSI format. You can import them in any word processor, supposing it is capable to read ANSI files (we know of no word processor unable to do this). Here you can re-format the results, change the layout, etc. as you wish when publishing your study. If the results consist mainly of numerical data – as in frequency counts of codes – you can use result files directly in your statistical package (again supposing it is able to read plain text data). Alternatively, you may convert frequency lists in CSV (comma separated values) tables, which can be imported in most spread-sheet programs and also statistic programs.

1.2.4 How to get out of AQUAD

Find your way back to the main menu from wherever you are by either choosing 'OK' or by choosing 'Cancel' from the panel of buttons on the right side of the screen (if necessary several times). Then you choose *Project* from the menu headline, and from the sub-menu, which is now pulled down, you select 'END'. All you have to do is to confirm your choice by selecting 'END' again in a subsequent window – and AQUAD is terminated.

In case something very unusual happens, AQUAD will place a warning on the screen. Please, tell the author of the program about this error and where in the program it happened and what you intended to do within this module. The author's e-mail address is shown in the final window before terminating AQUAD. In several cases when AQUAD terminated its work in an unconventional way, we found that a user had decided later to copy the program modules or text files in other sub-directories than originally defined during installation. This leaves some internal parameters unchanged! Please, do not copy the program modules manually from one directory to another, but always use the installation routine, if you want to do so. Your basic data, i.e. your text-, video-, audio- or graphic files, have always to be stored in the same directory as AQUAD (for instance, "C:\Aquad7").

1.3 How Aquad facilitates to find out the meaning of data

If we mention here the term "texts," we understand according to Oevermann et al. (1979, p. 378) "protocols of real, symbolically mediated social actions or interactions, whether they may be written, acoustic, visual, in various media combined recordings or specifications that can be otherwise put into the archives." In short, a "text" as defined by Overmann and colleagues can be everything: a written text, audio/speech, video, pictures. A text is a protocol of something that really happened and which can be investigated with care. Principally, the analytic procedures of qualitative social research can be differentiated by which of two levels of the text's reality the analysis is focused.

Oevermann et al. (1979, p. 367) compare "the reality of a text's latent structures of meaning" with the "reality of subjectively and intentionally represented meanings of a text." Thus, Oevermann differentiates between people's subjective intentions and their objective motivations. The latter can be reconstructed by making use of inter-subjective methods of analysis, the former is unknown to us and will ever be as we cannot perceive people's mind and its content directly.

The latent structure of meaning can be deduced from the rules guiding any interaction. Oevermann et al. (1979, p. 370) mention "syntactic rules, pragmatic rules, rules of interaction sequences, rules of turn taking in conversations, etc.". The people involved in the production of the text, i.e. the interacting persons must not be subjectively aware of this latent structure, nevertheless these relationships are valid and can be deduced from the text. In contrast to this approach an analysis on the level of subjective meanings has to exploit what the actors subjectively are aware of, that is above all their motives, intentions and emotional experiences within the situation represented by the text.

1.3.1 Sequence analysis: The reconstruction of latent structures of meaning

The reconstruction of the latent structure of meaning has to follow the process of interaction step by step. At each point of an interaction sequence the actors have various possibilities to continue in accordance with the underlying rules. However, from this broad field of possibilities only one option can be selected and realized at a time. The sequence of those selections yields the structure of the interaction event (Oevermann, 1991). The sequence analysis is a method that takes seriously the dynamics of interaction and the selective nature of each single step realized in this process. That is why it does *not* start from an overview of the text as a whole and does *not* look only for those text segments that are relevant for answering the research question. Instead, the research concentrates on one text segment each and tries to interpret it. Let us take a text as an example that begins with two words:

Speaker 1: *"Hello, hello!"*

The task of the analyst during the first phase of generating hypotheses consists in noting all imaginable meanings of this text segment. Wernet (2009, p. 39 ff.) describes the whole procedure as a sequence of "telling stories" (stories in which the text segment could occur), "forming versions" (that is, to sort the stories by comparing them according to similarities and differences) and "confronting these versions with the real context." Our example could concern

- a speaker, who tries to cause the attention of an acquaintance, who is walking in some distance;
- or the speaker draws the attention of somebody, who does not notice that s/he has just lost something;
- or the speaker opens the door and greets somebody;
- or the speaker meets an acquaintance, but has no time and hushes away (in the sense of "Hello, hello - we'll talk later" or "I do not want to talk now - maybe later").
- or the speaker is indignant about something (in the sense of "Hey, hello – what's that about!");
- or the speaker tries to cause unspecifically the attention of a group of people.

Here we have three versions of the exclamation "hello, hello" (causing attention; greeting; expressing indignation). Additional meanings may be possible. The next text segment or the next segments will show, which version matches the context. But: If we read the next segment before we make notes about our possible versions, the scientific procedure would be broken, because we would use knowledge about parts of the text we want to investigate. This just leads to endless loops of self-fulfilling prophecies.

In this way we try to detect the objective meaning of the recorded actions or "expression gestalten", not only the subjectively meant sense of (inter-)actions and their consequences, which a person is aware of and which s/he is able to reflect. An "expression gestalt" is the real manifestation of a person's subjective dispositions. They are manifested in the practice of life. The methodological consequence is – as Oevermann (1996, p. 2) underlines – "that every subjective disposition, i.e. every psychic motive, every expectation, every opinion, attitude, value orientation, every imagination, hope, fantasy and every wish cannot be directly assessed, but only an expression gestalt or a trace, in which it manifests itself or which it leaves behind." This reflects the general conviction in social sciences that the relevant processes and conditions (motives, emotions, thinking) cannot be observed directly, but have to be deduced from observable indicators. What we actually do is to reconstruct and investigate tracks of life along their natural order.

The stance of deduction of non-observable personal constructs from indicators covers almost the complete field of test-theoretical applications in the social sciences. Whenever there are conclusions about persons based on their reactive behavior (mostly marking answers on scales with specific questions, for instance, "Do you like parties?"), methods of the so-called methodological behaviorism are applied (Groeben, 1977).

Consequently the objective meaning encompasses the area, in which the subjective meaning ("I want/think...") is expressed and leaves its traces. These traces correspond with the actual interactions and actions in the real world. In the approach of "objective hermeneutics" the concept of objectivity claims that by applying methodological operations a clear proof of the reality can be established and the interpretation of the traces can be called "objective." However once more, only the traces of the practice of life can be analyzed, not this practice itself in its continuous manifestations. Anyway, the approach claims a level of reality of its analyses that otherwise is claimed only in the natural sciences.

The goal of searching the latent structures of meaning relates the adjective "latent" to the fact that the reconstructed structures must not be conscious and that the people involved must not be able to reflect them. But, once the objective structures are reconstructed, we can deal with the subjective meaning of actions much more precisely. And we can confront people and their world view with ours – which is of practical advantage in real life settings (e.g. counseling, therapy, education, working contexts). The reconstruction is based on the assumption that people have in common (in their language, in their social norms) a set of inter-subjectively shared rules and meanings. These can be perceived intuitively. Thus, on a logical-analytical level of reconstructing the latent structures of meaning, i.e. the structures to be reconstructed, are independent of an actual (manifest) realization of these supposed structures within the actors' consciousness. The latent reality is abstract, cannot be perceived directly, but becomes evident as an experience by means of methodological rules.

Referring to generic and generally valid rules of language Oevermann evaluates the reconstructed structures as objective. In contrast to this position he rejects any formulation or assumption regarding the world of subjective experiences, because they cannot be reconstructed by the researcher. In this sense, the approach of "objective hermeneutics" is close to George Herbert Mead's social behaviorism. Applied in the research program of subjective theories, it does not follow the line of traditional psychological thinking (Groeben, 1986). However, these differences are mostly due to methodological procedures and probably can be solved in future.

The module "*Sequence Analysis*" in AQUAD organizes the work with a text into three phases:

(1) As a prerequisite for sequential analysis the texts have to be subdivided into segments.

(2) Subsequently hypotheses are generated in turn about all meanings of these segments the researcher can think of; these hypotheses are grouped into "versions" of reading the text.

(3) Finally, these hypotheses are critically checked using the remaining parts of the text.

AQUAD uses as text segments – depending on the preparation of the texts – complete sentences, parts of sentences (determined by the punctuation marks) or segments containing a determined number of words. As soon as the researcher is convinced during generating hypotheses to have noted all hypothetical meanings relevant for his/her research question, the remaining data are used to confirm or reject the hypotheses. Only in this phase of confirming hypotheses the process of analysis follows non-sequential patterns, looking for text segments appropriate for confirming (or rejecting) hypotheses.

The sequence-analytic interpretation of texts is guided by five principles: (1) independence of context, (2) literal account, (3) sequential order, (4) extensive analysis or totality and (5) austerity (see Wernet, 2009, p. 21-38). In chapter 5 the details are presented and discussed.

1.3.2 Content analysis: Qualitative content analysis / Quantitative content analysis

Since the beginnings of a broader reception of text analysis in the social sciences the strategies of accessing the meaning of texts describe an important difference between quantitative and qualitative approaches. In the same year, 1952, two trendsetting articles were published by Berelson and by Kracauer. While text analysis serves for Berelson to assess systematically, objectively, and quantitatively the *manifest* contents of communication, qualitative content analysis according to Kracauer tries to reveal the categories of meaning hidden or *latent* in the data.

Qualitative content analysis

Qualitative content analysis treats the complete text (or other qualitative data sets) instead of interpreting it stepwise following the sequence of text segments. The various forms of content analysis have in common that they try to understand the experiences and actions of people from their own subjective point of view. The actions of these people are explained by putting them down to the frame of reference of their subjective or implicit theories. In any case, the task at hand is to achieve access to the subjective world view of the text producers.

Therefore, critical data segments are interpreted and categorized, that is, a specific meaning is attributed to a critical data segment and a symbol representing this meaning is attached to this segment. Following the use of language by Miles & Huberman (1984) and by most software developers for qualitative analysis, we will call these symbols "codes." Coding of the text transforms the colorful, individual formulations of our subjects' everyday language into a less complex and less ambiguous system of notations. In order to illustrate this procedure, we will give you an example from a research project that we will refer to regularly in this manual:

> In a study by Carlos Marcelo at the University of Sevilla, Spain, on problems of beginners in the teaching profession, 105 novice teachers were asked about their experiences at school. According to the model of professional socialization developed by Jordell (1987), Marcelo started the reduction of data with the search for statements by the teachers about "personal", "teaching-specific," and "institutional" influences. The corresponding segments in the interview texts were coded as EAS (personal experiences as student), ELA (experiences in the training as student teacher), UEB (convictions), etc. (Huber & Marcelo, 1993).

When researchers were still coding their data texts by hand, they usually marked in some way the beginning and end of a text segment they found relevant, considered the content, and then wrote the appropriate code in the margin. In AQUAD the text is presented on the screen, the researcher marks with the mouse the segment to be coded. The code – or several codes – are entered into a separate coding window. At the same time, the program notes the number of the line on which the segment begins as well as the number of the ending-line.In addition, the exact beginning and the length of the coded text segment in the character string of the actual file are stored. AQUAD follows the same principles when you work on videos, audios, or images, of course, using the reading of a counter (frames, seconds) or image coordinates instead of line numbers. This procedure is explained elaborately in a later chapter. The boundaries of segments may overlap in any way the researcher desires, and smaller segments may be nested within larger ones. In addition, more than one code may be attached to the same segment.

It is very important that you have AQUAD line-number your text – and *not* your word processor! For some operations, AQUAD expects the line numbers at a particular location in your text file and adds these numbers "on the fly" whenever they are needed within a particular module – whereas your word processor would make line-numbers a permanent content of your text files. Please, let AQUAD number the lines of your text transcriptions; this is done automatically when you select a function that needs line-numbered text.

AQUAD will compile automatically a list of all the codes used in all of your files. This list is called the *master code list*. We must distinguish between this code list and those code files which are assembled by AQUAD for every single code in a file, together with the references to the place of their occurrence: number of the text, number of the first and last line of the segments. The code entries in code files can easily be corrected, added to, or modified in AQUAD. Furthermore, the organizing system itself can be restructured by combining particular categories, thus creating "meta-codes". You will learn more about this in chapter 7.

Once some or all of the data files are coded, the computer can be instructed to compile a collection of all text segments that have the same code, meaning that they belong in the same category. Some researchers may want to use the results to look for pervasive themes within one category, so that the category can be described in terms of the commonalities and uniqueness found in the data regarding the topic represented by the category. Researchers who are more interested in the generation of theory will want to inspect the content of their categories to assure consistency of coding, and category integrity.

Depending on these two researcher interests (descriptive/interpretational or theory-building), the nature of the codes will vary. When a researcher wishes to interpret and to create a higher-order analytical description of the data content, codes are more like pointers to a specific topic as alluded to by the subject. They may occur as often in a single data file as the topic is mentioned. When a researcher is interested in the construction of theory, the text itself is often not important in the later phases of analysis. What counts is the fact that a certain piece of evidence does or does not exist in the data file. The code signifies the occurrence of that piece of evidence. Although the evidence may occur several times in the same file, *one* occurrence is sufficient to characterize the file as "containing the evidence". The code, therefore, represents more a characteristic than a topic. Very simple codes may merely stand for a socio-demographic attribute, such as "male" or "female." Others may represent a particular attitude or experience, or anything else that may become an important category in the research context. More sophisticated codes may represent a sequence of other codes as an indicator of some complex topics found during a linkage analysis. AQUAD will add such 'linkage codes' or 'sequence codes' automatically to your code files, if you wish. More about coding in chapter 6.

Quantitative content analysis

The core procedure in quantitative content analyses is counting the frequencies of keywords in the text data of a project and comparing them. The keywords form a pre-constructed analytic net that allows so to speak "to catch" aspects of meaning relevant for the research question. As an example we show a catalog of keywords, which was used to analyze the transcriptions of so-called "TV-duels" between the applicants for the US-presidency. The catalog was based on the assumptions that the actual office-holder will point out himself and his achievements ("I" in various combinations), while the competitors will underline common interests with their voters ("we") or criticize specific activities and goals of the government ("they"):

```
i
i'd
i'll
i'm
i've
them
they
they'll
they're
they've
we
we'll
we're
we've
```

Besides, often coding is facilitated by locating words and phrases in data, especially in transcripts of interviews where subjects may use them to describe a particular fact, situation, experience, or opinion the researcher is interested in. AQUAD allows you finding such words and phrases in any data text. The frequency with which these critical words occur may be an indication of the strength of the emphasis placed on the concept expressed with them. Therefore, AQUAD will count the frequency of their occurrence. Furthermore, it will construct "key-word-in-context" (KWIC) lists, consisting of a print-out of all the text lines that include the word you are looking for, together with a reference to the place where the word was found in the data. All the researcher has to do then is to decide whether s/he wants to attach a code to this sentence or not.

1.4 Tables or matrices

Matrices are recommended as important forms of data display, especially by Miles and Huberman (1994). With the help of AQUAD, a two-dimensional net for a structured output of text passages can be created. To construct such a matrix, you must define the columns and the rows of the matrix. For the columns only "singular" or "profile" codes can be used, i.e., codes that are only used once in a given text file, as for instance, codes for the gender or age of the subject, or for the interview number, etc. Because these codes represent characteristics of the person interviewed, a site, etc., they are also called "profile codes". The rows are for "interpretational" or "conceptional" codes that are especially interesting for the perspective under which the data are displayed. For example, in the above mentioned study about experiences of beginning teachers we could get an interesting overview of the data simply by creating a table of text passages for which the columns would be defined by the gender of the subjects, and the rows by their "convictions," "worries," and "self-concepts." In this particular display we would see all the text segments from all the texts that correspond to the definition of this 2x3 cells matrix.

If we produce text matrices of this kind, we may well have to accept an enormous consumption of paper – and we need big empty walls in order to hang up the print-outs. As a second option for the display of qualitative data, AQUAD lets you print as the content of the cells only the names of the codes, together with their place of occurrence in the data. This form of matrix is less voluminous and easier to review. It allows you arriving at first ideas about the possible connections between categories. These hunches are the beginning of the development of qualitative hypotheses that you later may wish to test. Even smaller are frequency matrices as results of table analysis. They show only the frequencies of codes defining a row in case the code defining the corresponding column is given in a data text. Of course, you receive less information by this type of analysis.

1.5 Exploration of linkages and regularities

If the coding you have done refers not merely to a particular topic contained in the data (such as "schooling" or "relationship with father"), but tells us something about the actual content or quality (such as "12 years" of schooling, or relationship with father "strained"), AQUAD will help you discover whether there are patterns and linkages in your data. In fact, that is the main point of AQUAD: to assist in theory-building. Perhaps "12 years of schooling" regularly clusters together with "professional success" and "high income". Perhaps "strained relationship with father" appears only in data segments that also hold some other feature you are interested in. In order to find out about these patterns or linkages, you would first need to postulate one; you must know what you are looking for before you can have AQUAD look for it. Then you would make AQUAD examine all your data for such cases, and then draw your conclusions from what AQUAD finds.

Of course, this is a very simplified description of the actual process (later in this manual you will find out how complicated are the linkages you will be able to analyze with AQUAD). But it conveys the basic principles.

AQUAD has preformulated for you a number of search algorithms called "linkages". For instance, you might "hypothesize" that code "ABC" always appears in your data within a short distance from code "XYZ". Just substitute your own codes for the general ones here, make the appropriate choice from a "linkage" menu, and AQUAD will do the rest (all but tell you whether what you have found makes sense). The hypothetical linkages AQUAD has pre-formulated are the following:

1. Two codes occur in the same data document within a specified distance of each other (true for which cases?)

2. Two codes occur in the same data document within a specified distance of each other (true/false for which cases?)

3. Three codes occur in the same data document, with code #2 within a specified distance of code #1, and code #3 within a different specified distance of code #1 (true for which cases?)

4. One or both of two codes occur in the same data document (true for which cases?)

5. One, two, or all three of three codes occur in the same data document (true for which cases?)

6. Two codes occur within a specified distance of each other in a document, which contains a particular third code (true for which cases?)

7. Two codes occur within a specified distance of each other in a document, which contains a particular third and fourth code (true for which cases?)

8. Three codes occur in the same data document, with codes #2 and #3 being sub-codes of code #1 (true for which cases?)

9. Three codes occur in the same data document, with code #2 being a sub-code of code #1, and code #3 occurring within a specified distance of code #1 (true for which cases?)

10. Two, three or four codes related to the same speaker or question (cf. "speaker codes"; chap. 6) occur in the same data document within specified distancies (true for which cases?)

11. Two particular codes or two alternative codes (or one of the first pair of codes and another code) occur in the data documents within a specified distance (true for which cases?)

12. A particular code or an alternative code occurs in the data documents within a specified distance of a second code or an alternative second code (true for which cases?)

When AQUAD "tells" you what it has found, it does so by referring to the places in your data where the match occurred. If you work with text files as data base, extracted text segments are added, if you mark the appropriate checkbox.

In addition to these pre-formulated linkages, you can formulate your own by logically connecting up to five codes. A special section of this manual is devoted to a description on how to do this (chapter 8.3.3).

In case you have distinguished in your codes between various speakers (for instance, in a group discussion) or between the questions of a questionnaire as "as if"-speakers", a comparison of two speakers may be interesting. AQUAD offers 13 hypothetical "linkages" for the analysis of relations between the utterances of two speakers. Into the abstract formulations you enter your concrete codes, just as you do with variables in an equation. In the following you find a list with all hypothetical linkages actually available in AQUAD .

Code A, Code B, etc. represent coded utterances of speakers, in which code A, B, etc. may be identical for seaker 1 and 2 (see linkage 1).

1. S1: Code A -> S2: Code B
   "Speaker 1 says A, speaker 2 continues with B" (A and B may be identical, that is, speaker 2 repeats a remark of speaker 1)
2. S1: Code A -> S2: Code B and Code C
   "Speaker 1 says A, speaker 2 continues with B and C"
3. S1: Code A -> S2: Code B or Code C
   "Speaker 1 says A, speaker 2 continues with B or C"
4. S1: Code A and Code B -> S2: Code C
   "Speaker 1 says A und B, speaker 2 continues with C"
5. S1: Code A and Code B -> S2: Code C and Code D
   "Speaker 1 says A und B, speaker 2 continues with C and D"
6. S1: Code A and Code B -> S2: Code C or Code D
   "Speaker 1 says A und B, speaker 2 continues with C or D"
7. S1: Code A or Code B -> S2: Code C
   "Speaker 1 says A or B, speaker 2 continues with C"
8. S1: Code A or Code B -> S2: Code C and Code D
   "Speaker 1 says A or B, speaker 2 continues with C and D"
9. S1: Code A or Code B -> S2: Code C or Code D
   "Speaker 1 says A or B, speaker 2 continues with C or D"
10. S1: Code A and (Code B or Code C) -> S2: Code D
    "Speaker 1 says A und (B or C), speaker 2 continues with D"
11. S1: Code A and (Code B or Code C) -> S2: Code D and Code E
    "Speaker 1 says A und (B or C), speaker 2 continues with D and E"
12. S1: Code A and (Code B or Code C) -> S2: Code D or Code E
    "Speaker 1 says A and (B or C), speaker 2 continues with D or E"
13. S1: Code A and (Code B or Code C) -> S2: Code D and (Code E or Code F)
    "Speaker 1 says A and (B or C), speaker 2 continues with D and (E or F)"

AQUAD reports the findings by showing the positions within the data files, where you can find the linked segments. The files itself are not shown. On the level of forming hypotheses about linkages you do not work on the original data, but on a more abstract level on your interpretations represented by codes.

In addition to apply pre-constructed linkages you may also construct and test your own linkages. Chapter 8 (section 8.3.3) describes the necessary procedures.

1.6 How AQUAD deals with the exploration of causality

Causality is hard to prove. There are a number of quantitative methods by which researchers try to do so; their description would be beyond the scope of this manual. Ragin (1987) reminds us that quantitative methods (or the "variable-oriented approach", as he calls it) are not the only way to explore causality. In fact, they might be an inferior way, since it is difficult in variable-oriented research to deal with the complexity of multiple causation. Since quantitative researchers are aware of this, they have introduced sophisticated ways of remedying the problem. However, "as the complexity of the causal argument to be tested increases, intractable methodological problems are introduced" (Ragin, 1987, p. 68):

> The assumptions of statistical models become more strained in the face of intricate causal arguments, given a restricted sample size (Ragin, 1987, p. 68), and

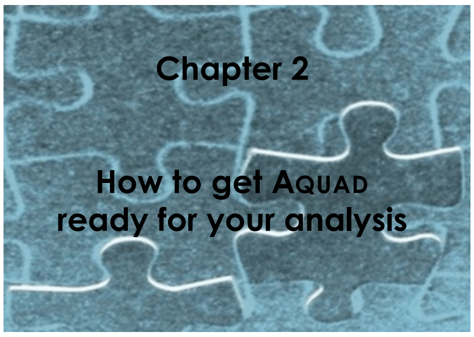
> ... the main weakness of the variable-oriented strategy is its tendency toward abstract, and sometimes vacuous, generalizations (Ragin, 1987, p. 69).

While "the main weakness of the case-oriented [qualitative] strategy is its tendency toward particularizing" (Ragin, 1987, p. 69), it will "allow analysis of parts in a way that does not obscure wholes" (Ragin, 1987, p. 83). It does so by "COMPARING wholes as configurations of parts" (Ragin, 1987, p. 84). The method Ragin recommends is the "Boolean method of qualitative comparison", so called because it uses some techniques invented by the mathematician George Boole (1815-1864).

This process of comparison, however, cannot be done without some kind of reduction. The "parts" Ragin is referring to can be understood as items occurring in a data text that are considered by the researcher "conditions" s/he suspects may be prerequisites for an "outcome" s/he is interested in. The items (which may actually consist of an entire text segment) are reduced to a code (as we would use it in the linkages described above), and the presence or absence of the code in the data is reduced to the dichotomous values "1" (present) and "0" (absent). Thus we have entered a still higher level of abstraction.

Let us assume we suspect that there are three "conditions" (A, B, and C) that may have something to do with "outcome" X. Are all three of them necessary to produce that outcome? Does outcome X also occur when none of these conditions are present? Perhaps combination AB or BC or AC is the one that leads to outcome X? Does it do so in all cases? Is it necessary perhaps for B to be absent so that X can result? In order to explore such questions we would begin by constructing tables in which we enter all theoretically possible combinations of the conditions (as zeros and ones), one combination per row (Boole calls these "truth tables"). Then we would examine our data and note first which combination (counting both presence and absence) exists in each case, and then whether outcome X is indeed present in that case. Gradually, we will encounter most of the combinations in our data, and we will also note in our table in how many cases each occurs. Some combinations of conditions may not occur at all in "real life" (as our data represent it), and we would have to enter a "?" in the column that records whether outcome X was obtained or not. The next step is to examine the table to see what it tells us about the connections of the various combinations of conditions to the outcome. This is done with the help of an algebraic method that is actually fairly simple, but requires some concentration to follow. It is explained in more detail in chapter 11. In this part of the manual you will also learn how to transform your qualitative (and quantitative) data into "truth values." The result of the process is the pinpointing of the one or the several constellation(s) of conditions that is/are so invariably connected to outcome X that we will have to accept it/them as the "cause."

AQUAD provides the module *Implicants* that allows you performing Boolean qualitative comparisons on your data. One requirement is, of course, that your data represent a large enough number of cases to make the comparisons meaningful.



Chapter 2

How to get AQUAD ready for your analysis

AQUAD and your data should be prepared for your project according to the following principles:

- First you install AQUAD on the hard disk of your computer by activating the program "*aquad7-setup.exe*" (on the CD-ROM or in your download directory).
- If you want to analyze texts as data, which were transcribed, you convert the text files *within your text program* into the ANSI-text format (*.txt) or the Rich Text Format (*.rtf; less recommended because of conversion problems in some text programs). Details are described in chapter 3. Audio files are expected either in the format *.wav or *.mp3, video files in the format *.avi, graphic files in the format *.jpg. More about these latter formats and how to digitalize and convert your original files will be described in chapter 4.
- When you set up your project (see chapter 2, paragraph 2.2), you create a list of files called "file catalog", which contains the names of all those data files you want to analyze later.

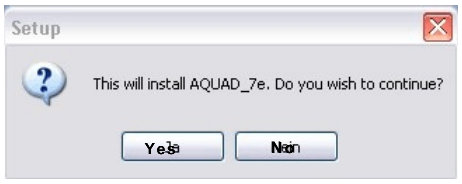
2.1 Installation of AQUAD

The installation of AQUAD is done by the program "*aquad7-setup.exe*" on your program-CD. It supports you also in setting up the directory where you want to have AQUAD installed. Here is what you should know before you start the installation:

- Your computer must have a hard disk with approximately 90 MBytes free disk space.
- AQUAD must have a chance to access the directory where Microsoft's WINDOWS is installed, because some files are stored there.
- Your computer must have a CD-ROM drive. In the following description we assume this drive is available and addressed as drive D:

2.1.1 The installation program[1]

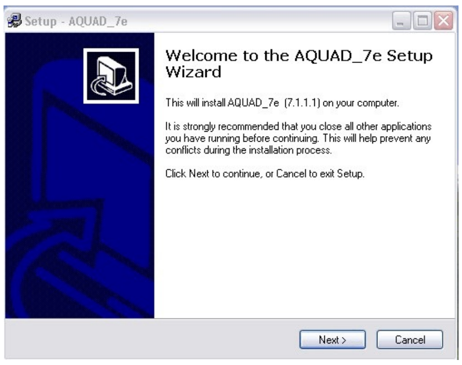
This is what you see on the screen when you start the installation:



This will install AQUAD_7e. Do you wish to continue?

Of course, you click on "Yes" to continue the installation.

[1] The installation is realized by Jordan Russel's "Inno Setup Wizard." We want to thank the author for distributing this excellent, most flexible tool as freeware. Downloads are available from http://www.jrsoftware.org


Setup - AQUAD_7e

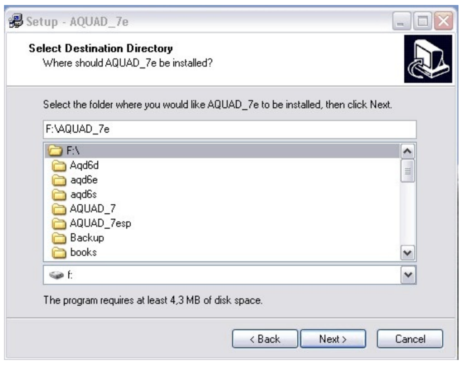
Welcome to the AQUAD_7e Setup Wizard

This will install AQUAD_7e (7.1.1.1) on your computer.

It is strongly recommended that you close all other applications you have running before continuing. This will help prevent any conflicts during the installation process.

Click Next to continue, or Cancel to exit Setup.

Here you click on "Next" to continue the installation.


Setup - AQUAD_7e

Select Destination Directory
Where should AQUAD_7e be installed?

Select the folder where you would like AQUAD_7e to be installed, then click Next.

F:\AQUAD_7e

F:\
Aqd6d
aqd6e
aqd6s
AQUAD_7
AQUAD_7esp
Backup
books

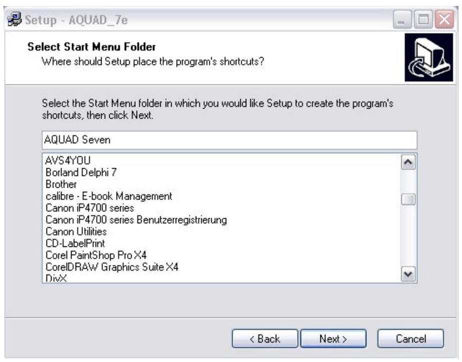
f:

The program requires at least 4,3 MB of disk space.

First you have to select a destination directory where you want Aquad to be installed. Originally the software suggested "C:\AQUAD_7".

We selected instead the drive "f:" in the window below and changed in the upper window the name to "AQUAD_7e" (English version).

If you click now on "Next", the software will ask your permission to create this directory – in case it did not already exist on your hard disk.


Setup - AQUAD_7e

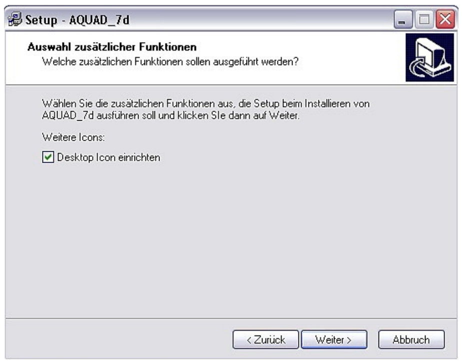
Select Start Menu Folder
Where should Setup place the program's shortcuts?

Select the Start Menu folder in which you would like Setup to create the program's shortcuts, then click Next.

AQUAD Seven

AVS4YOU
Borland Delphi 7
Brother
calibre - E-book Management
Canon iP4700 series
Canon iP4700 series Benutzerregistrierung
Canon Utilities
CD-LabelPrint
Corel PaintShop Pro X4
CorelDRAW Graphics Suite X4
DivX

In the next window you confirm the name of the place, where AQUAD will put its shortcuts. The setup routine has already entered "AQUAD Seven" as default name, therefore you may just click on "Next >".


Setup - AQUAD_7d

Auswahl zusätzlicher Funktionen
Welche zusätzlichen Funktionen sollen ausgeführt werden?

Wählen Sie die zusätzlichen Funktionen aus, die Setup beim Installieren von AQUAD_7d ausführen soll und klicken Sie dann auf Weiter.

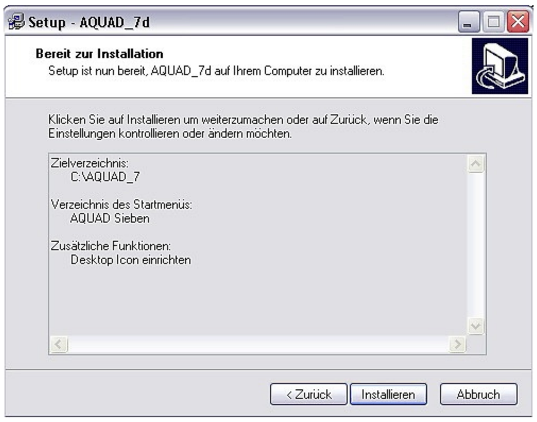
Weitere Icons:

☑ Desktop Icon einrichten

If you want to start AQUAD later just by clicking on a desktop icon, you confirm here by clicking on "Next >" that you want to have created such an icon.

Necessary sub-directories are created automatically by AQUAD, namely

- **cod** is for codefiles, memos, word and code lists, table definitions, etc.;
- **cod_s** is for safety copies of code files;
- **lit** is for the manual and the GNU GPL license;
- **mco** is for the meta-code files;
- **prg** contains the program and help texts;
- **res** is for saving results of various analyses.



Bereit zur Installation
Setup ist nun bereit, AQUAD_7d auf Ihrem Computer zu installieren.

Klicken Sie auf Installieren um weiterzumachen oder auf Zurück, wenn Sie die Einstellungen kontrollieren oder ändern möchten.

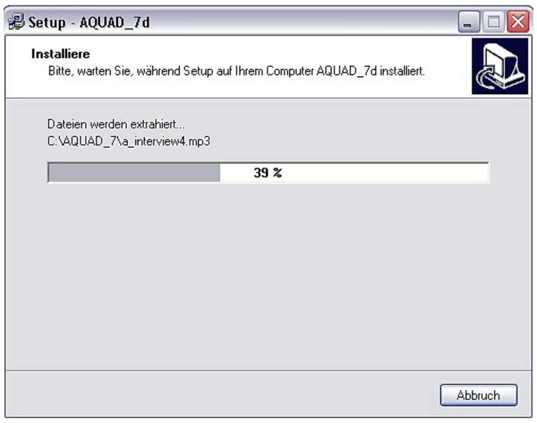
Zielverzeichnis:
C:\AQUAD_7

Verzeichnis des Startmenüs:
AQUAD Sieben

Zusätzliche Funktionen:
Desktop Icon einrichten

All data that you are going to analyze, i.e. *transcriptions* of texts, *audio*-files, *videos* and *image* files have to be copied into the *root directory* of AQUAD, here inour example in: "*C:\Aquad_7*"

EA summary shows your selections; after clicking on "*Install*" the installation will start.

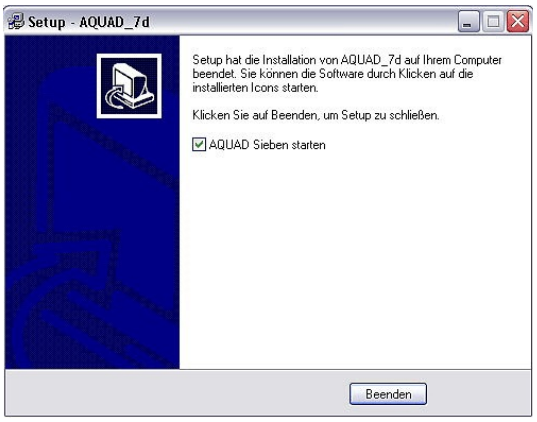


Installiere
Bitte, warten Sie, während Setup auf Ihrem Computer AQUAD_7d installiert.

Dateien werden extrahiert...
C:\AQUAD_7\a_interview4.mp3

39 %

The following window shows the progress of the installation.



Setup hat die Installation von AQUAD_7d auf Ihrem Computer beendet. Sie können die Software durch Klicken auf die installierten Icons starten.

Klicken Sie auf Beenden, um Setup zu schließen.

☑ AQUAD Sieben starten

Finally you can have AQUAD Seven started directly from the installation routine.

2.1.2 Starting AQUAD for the first time

The installation routine added AQUAD to the list of programs available on your computer. To start the program later, you just look for the desktop icon - if you had the installation routine create one – and double click on it. Alternatively you activate the program list as usual and click once or twice – depending on your WINDOWS installation – on "*AQUAD Seven*" or whatever name you selected for the program group.

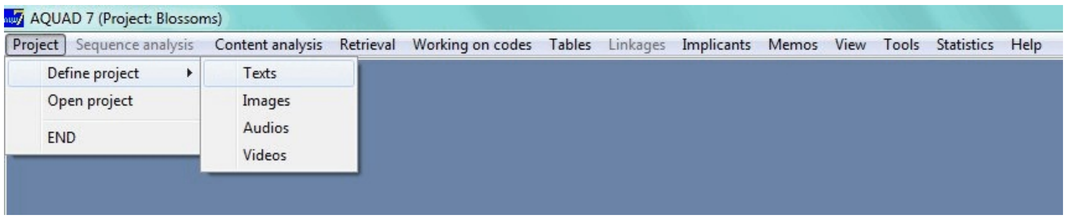
When you run AQUAD for the first time, all example files are installed, but of course the parameters of your research project are not yet defined. We suggest, you try first how to handle AQUAD's functions by creating a "*New project*" (see paragraph 2.2) named "interview" with the example files "interview_1.txt" ... "interview_4.txt."

These files are translations of interviews with four Spanish teachers at the very beginning of their careers. The data were taken from a publication of Marcelo (1991) and "enriched" – in order to enhance the possibilities to work with these examples – by statements of teachers in a study of Zabalza (1991).

Additionally we suggest that you take the manual and try to repeat what is described there with these texts and their corresponding files (codes, memos, code catalogs, etc.). You may explore the possibilities without preoccupations that you change, erase or overwrite anything. All data for these examples are on your installation file and you can restore the original state whenever you want, just install AQUAD anew (by the way, this can be done even when you work with your own data in your project, because the installation will not touch any other files than those with names of the original ones).

2.2 How to set up a new project

When you activate the set-up routine "*Project*" -> "*Define project,*" you have to tell AQUAD, which type of data will be analyzed – texts, pictures, audios (sound files) or videos. In this example we start to work with the interview files described above, therefore we click on "*Texts.*"

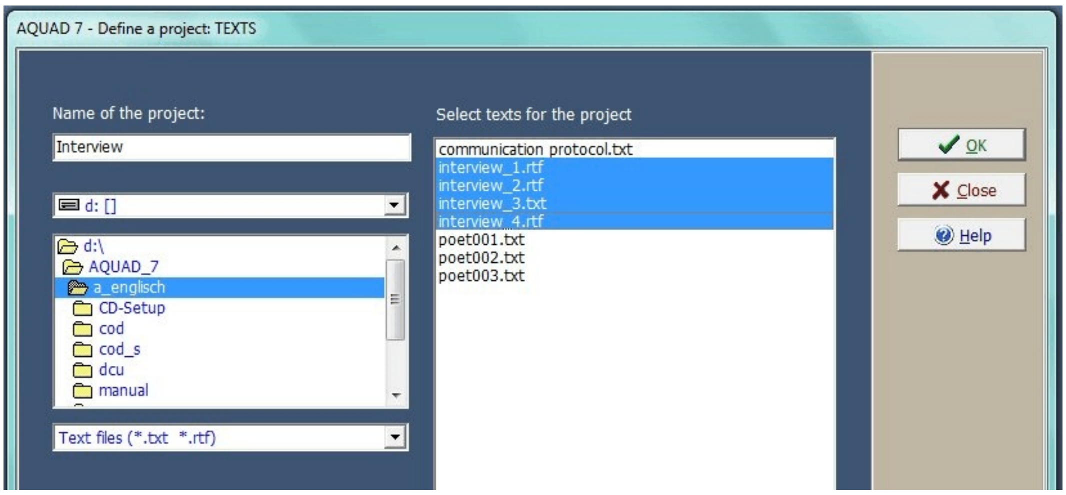


As no project was defined during installation, the actual path is the one to the directory of the program itself, that is "*C:\Aquad_7\PRG*". Of course, no text transcriptions are in this directory, because they were copied to AQUAD's root directory "*C:\Aquad_7*" – and the program will complain with an error message "File not found".

This is easy to repair: You select in the windows for drive and directory on the left exactly the path, were the texts are stored. The drive "*C:*" is correct, but you have to double-click in the large directory window on the line "*Aquad_7*". Immediately the window for the names of data files on the right side will show all text files *.txt (and *.rtf) saved until now (see next page).

Here you select all those text files, which you are going to analyze in your project "*Interview*". In case you have text files from several projects in the root directory it is helpful now if you followed the advice to give the same name to all texts of the same project, but differentiate them by numbers – as in our example.

Now we must not forget to enter a meaningful project name. Erase an old, inappropriate name by double-clicking on it and write a new name, in our example "*Interview*". Then you click on the button "✓*OK*".


AQUAD 7 - Define a project: TEXTS

Name of the project:

Interview

d: []

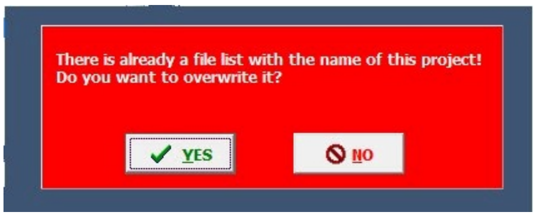
d:\
AQUAD_7
a_englisch
CD-Setup
cod
cod_s
dcu
manual

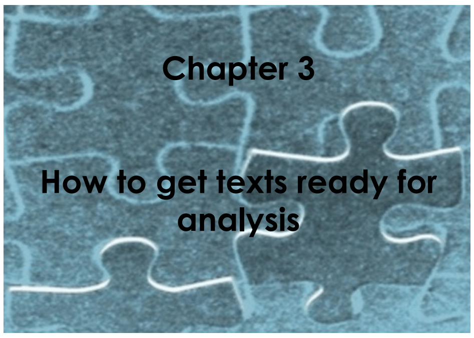
Text files (*.txt *.rtf)

Select texts for the project

communication protocol.txt
interview_1.rtf
interview_2.rtf
interview_3.txt
interview_4.rtf
poet001.txt
poet002.txt
poet003.txt

AQUAD will warn you with a red window that there is already a list of file names for this project. Of course, it was already copied into the sub-directory "*cod*"during the installation process. Just click on "✓*YES*" and overwrite the old settings.


There is already a file list with the name of this project!
Do you want to overwrite it?



Chapter 3

How to get texts ready for analysis

Although during later working sessions with AQUAD it is sufficient to click on an icon on the computer screen (to load AQUAD) in order to immediately afterwards continue working on your analysis, at the very outset you must do a number of things that make your data files available for AQUAD. In particular, you must

- format your text documents;
- convert the data you entered into your word processor into a format called "txt" (ANSI-text) or "rtf" (Rich Text Format);
- copy these text files into the root directory (here "C:\Aquad_7");

3.1 Format your text documents

You may use whatever word processor you prefer for entering your data. Before they can be worked with in AQUAD, however, you need to prepare each data file in a certain way. As a first step you format the texts anew: each of the lines should be no longer than 60 characters, and they could be even shorter if your texts contain a lot of information and therefore require heavy coding.

This "formatting" is done while your data are entered into your text processor. In most word processors you determine the maximum length of the lines by setting the margins accordingly (no margin on the left, a margin on the right wide enough to make sure not more than about 60 characters/spaces will fit on the line before the next word is wrapped to the following line).

What can you do if your texts are already written and formatted with a word processor? This is no problem at all. All word processors, even simple text editors offer options to re-format texts as described above. In most word processors you can define the layout of pages ("style" definition) and load this definition together with your text files, which then become adapted to the new format automatically. Please, refer to the manual of your word processor for details.

3.2 Provide proper file names

When you save your data files, AQUAD requires that you use only eight characters for naming your files. The rule for file names in AQUAD is:

- File names are composed of a name part and an extension part.
- You define only the name part, consisting of maximally 60 characters.

- Extensions are added automatically by your text program (for instance: *.txt; see below) and by AQUAD.
- Suggestion: All text files in a project get the same name (maximally 60 characters), but are distinguished by a different three-digit number.
- Never use "work" as a file name, because this word is reserved for internal use by AQUAD.

You will probably want to use the characters and numbers to indicate what the content of the particular file is, so you can easily recognize it. For example, if your data consist of 24 interview transcriptions which you have numbered, you might name the original files that are the basis of your analysis

inter_001, inter_002, ... , inter_024.

When AQUAD works with these files, it creates additional parallel files, as for instance files that hold the code information for a particular file. The program constructs the new file names from the elements of the original names, so that the connection to the original file can be recognized. In our example, files with the names

inter_001.aco, inter_002.aco, ... , inter_024.aco

would be set up by AQUAD. (After coding you can see these new names in the subdirectory "..\cod". If your data files are already saved with names that deviate from this convention, you may rename them. You can do this either within Microsoft's WINDOWS "File Manager" or "Windows Explorer."

3.3 Convert text files into the format ANSI Text (*.txt)

Each data document that you want to use in AQUAD *must* be converted into a plain ANSI text file or into RTF. The first abbreviation stands for "American National Standards Institute", the second for "Rich Text Format". When you create text, your word processor fills your documents with formatting and other information that is invisible to you. These commands instruct the program, for instance, when to break up a line and move on to the next line, where to set margins, tabs and indents, they specify line spacing, page breaks etc., tell the printer what font to use and many other things. Unfortunately the commands are unique to each word processor. For instance, the last sentence would be stored in WordPerfect, the text processor we used for this manual, as

UnfortunatelyÇtheÇcommandsÇareÇuniqueÇtoÇeachÇwordÇprocessor.Ç

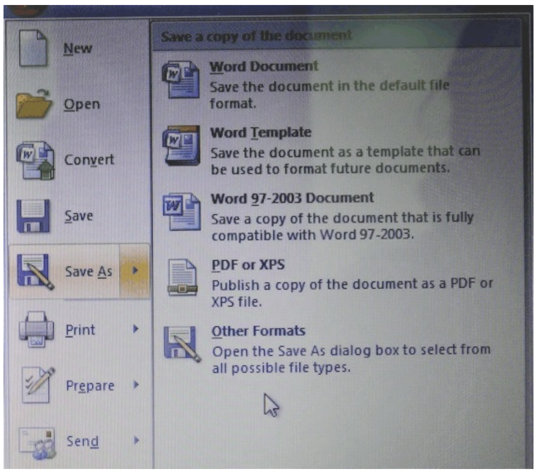
These program-specific characters need to be stripped away, if you want to exchange your texts via e-mail or to import your data into another program; they must be "standardized." In technical language, you must convert your documents into ANSI documents, the "standard code," or into RTF.

Most word processors offer a facility to convert text documents into ANSI or into RTF files when saving them within the function "*Save as...*". Please, read the manual of your word processor for further information. Key-words in the manual's index like "ANSI", "DOS text", "Import/Export", "text-in/text-out", "Rich Text Format", etc., may lead you to the information needed. If there is no choice between formats when saving your texts, then there should be at least a small additional program for text conversion. In this case, you should look in the directory, where your word processor is stored, for *.exe (program) files, whose names contain sequences of letters like "cnv", "conv", "textcnv", etc. (cnv or conv usually signals "conversion"). Usually, an utility program of this kind has to be started separately from your word processor to convert your data before you can import them into AQUAD.

With older versions of some word processors it may happen that the conversion facility contains bugs. The converted files then contain codes that are invisible in the text. The result in our own experience once was that AQUAD got caught up during loading. In this case the only thing you can do is to use the mode of your text program that reveals the codes so you can see them, or to use a special text editor. The annoying characters (mostly only one!) must be deleted manually. With newer versions of word processors such problems, to our knowledge, do not occur.

If your word processor is ancient enough not to include the option to save text in ANSI format or in RTF, find a small utility program that includes this procedure and use it on your data before you enter them into AQUAD.

The following two screenshots show this procedure when working with MS Word 2010:



Save a copy of the document

Word Document
Save the document in the default file format.

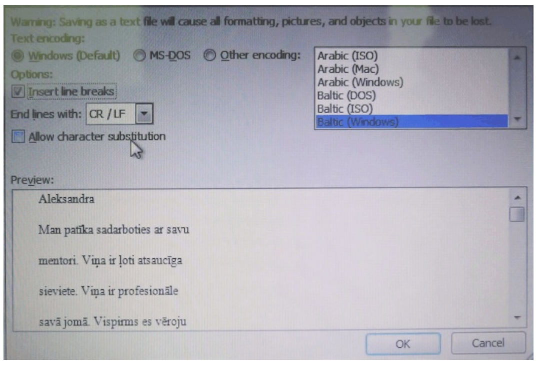
Word Template
Save the document as a template that can be used to format future documents.

Word 97-2003 Document
Save a copy of the document that is fully compatible with Word 97-2003.

PDF or XPS
Publish a copy of the document as a PDF or XPS file.

Other Formats
Open the Save As dialog box to select from all possible file types.

Here, the option "Save As" was selected, then we had to click on the right side on the option "Other Formats."



Warning: Saving as a text file will cause all formatting, pictures, and objects in your file to be lost.

Text encoding:
[✓ Windows (Default)] MS-DOS Other encoding: Arabic (ISO), Arabic (Mac), Arabic (Windows), Baltic (DOS), Baltic (ISO), [✓ Baltic (Windows)]

Options:
[✓ Insert line breaks]
End lines with: CR / LF
Allow character substitution

Preview:
Aleksandra
Man patīka sadarboties ar savu
mentori. Viņa ir ļoti atsaucīga
sieviete. Viņa ir profesionāle
savā jomā. Vispirms es vēroju

In the subsequent window we do not change the default settings "Windows" the language in the small windows on the right side (here "Baltic (Windows)" automatically selected on a computer in Riga), but we click on"Insert line breaks" und make sure that below the option is shown, which informs the computer to "End lines with" "CR/LF" (the standard codes: Carriage Return/Line Feed).

The next section can be skipped if you are reading this chapter for first orientation. It is important only once you have been working a while with AQUAD and discover belatedly that you still need to correct something in one or the other of your data files.

Editing data files after they have been converted into ANSI format or into RTF

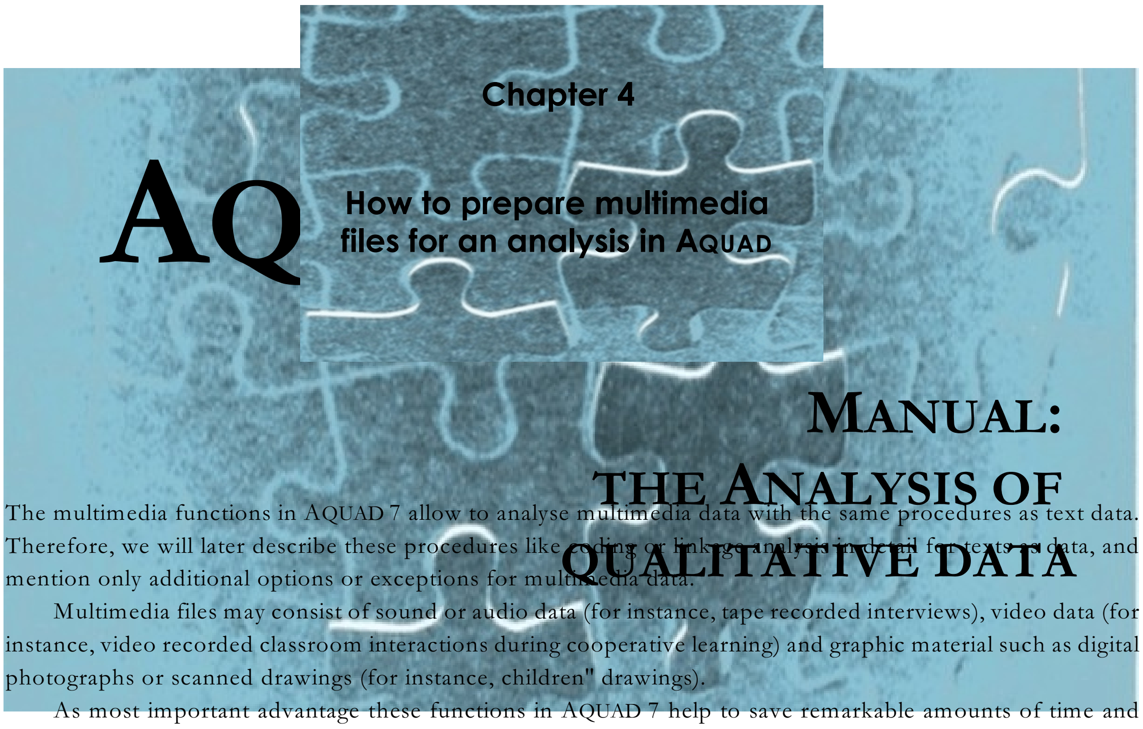
If you notice errors in your original texts, you may wish to correct them or add something to them. However, there is a basic rule in qualitative analysis that text documents should not be changed during the analysis. Many people found that versions 3.x of AQUAD made it too easy to violate this rule and they asked for some changes. What you can do now since version 4, if you happen to detect for instance an irritating typing error is only to go back to your word processor and edit your original document, convert it again into ANSI format or RTF, and save it in the path determined for data texts. There is one caution, though!

Whenever you make changes in a text file after you have started to code this text then the code entries in your code files still refer to the positions within the text before it was edited! In this case – and only in this case – the positions of units of meaning stored together with your codes *will not be valid for the newly formatted data. This means that all coding you might have done will be invalid, as well.* Therefore, we recommend that you keep editing to a minimum, that is, you keep it to changes like correcting typing errors, which will not lead to adding or deleting, but just changing characters.

3.4 Define a project (or various projects)

When you determined the settings for your new project (see chap. 2), you listed already all cases available for analysis in your project (text files with your transcriptions) in a file catalog. However, especially in the beginning of a project, you may want other file catalogs than one major catalog listing *all* files of your project. It is a good idea not to start your work not with all, let's say 65 files of your project, but to begin your work with one or more small samples of five or ten data files.

In this case you define just one (or more) additional project/s with different names. In our example the general project was named "*Interview*", now we define an additional project "*Interview sample.*" The codes, which you create during your work with the sample files will be available automatically once you work with all files in the general project.



Chapter 4

How to prepare multimedia files for an analysis in AQUAD

The multimedia functions in AQUAD 7 allow to analyse multimedia data with the same procedures as text data. Therefore, we will later describe these procedures like coding or linkage analysis in detail for texts as data, and mention only additional options or exceptions for multimedia data.

Multimedia files may consist of sound or audio data (for instance, tape recorded interviews), video data (for instance, video recorded classroom interactions during cooperative learning) and graphic material such as digital photographs or scanned drawings (for instance, children" drawings).

As most important advantage these functions in AQUAD 7 help to save remarkable amounts of time and money until now necessary for the tedious work of converting sound- or video-recordings into transcriptions, nothing to say about avoiding the danger to loose valuable information in the process of transcribing them. Instead, you will now be able – after coding your data – to move rapidly and exactly with a single mouse click to any important event recorded in your data.

Nevertheless it is possible and recommended to transcribe additionally some critical data segments even within AQUAD while you are analysing your data. You may also add memos to code files related to multimedia data. But you are no longer forced to reduce, for instance, videotaped observations to behavior ratings or descriptions. In AQUAD you mark the beginning and the end of relevant scenes and add meaningful codes. Afterwards you can analyse and retrieve these scenes with the same functions applicable to text data.

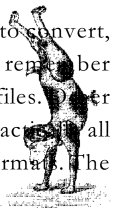
However, before you start an analysis of multimedia data you may have to convert your data files into formats, which AQUAD is able to read – just as is the case with text files. How to do this will be explained in the following paragraphs.

4.1 Pictures as data

Digital pictures and graphic files can be analysed directly in AQUAD. The software is equipped with a routine to import graphic files which come in the format

"*.jpg"

– but any graphic program that comes with scanners, digital cameras, etc. is able to save (and mostly to convert, too) graphic files in this format. As basic rule for analyzing pictures and graphics in AQUAD, please remember to save them as .jpg files. Thus you can compress them and work with small, easily manageable data files. Other graphic formats like, for instance, *.tif or *.bmp consume much more memory space. Meanwhile practically all graphic programs and most graphic viewers are able to convert a great number of different graphic formats. The maximal dimensions are 780 x 610 pixels. Larger files are adapted automatically and proportionally.



In the Internet you find suitable programs free on the pages ofcomputer journals, freeware pages and university servers. We have good experiences with the software "IrfanView" by Irfan Skiljan. This program is available free of charge for non-commercial application as, for instance, research projects at universities from

http://irfanview.tuwien.ac.at or
http://www.irfanview.de

4.2 Sound recordings (audios) as data

Audio recordings, which are stored in analog formats on cassettes, records or other storage media, are run through a process of digitalization and converted into a format computers can handle directly. Because the analog frequencies on the original data carrier are converted into digital signals, this process is called A/D conversion. As hardware you need for this conversion a sound card, which is able to import audio data through an "Audio In" connector (on most soundcards for 3.5 jacks). This input channel should be configured with the parameters

44,1 kHz, 16 bit stereo

corresponding to CD quality. Modern sound cards all fulfill these requirements. Should your initial audio data come from a DVD, you may need 48 kHz as setting. The manual of your sound card tells you about these technical details. Just try an A/D conversion and compare the quality of the digital product to the original recording. Many computer main-boards come with built-in sound chips, which are usually sufficient for A/D conversions and later work with AQUAD.

You import the analog data in most cases from a tape deck by connecting its "Line Out" jack with the "Line In" jack of your computer. Import and conversion are controlled by a software recorder, which comes on CD with most sound cards or can be found on the utility CD (directory "sound", "audio" or a corresponding label) of your computer. If you do not own a program of this type, it is easy to look for an audio recorder as freeware on the Internet.

We use as a program free of charge the digital audi recorder *Audacity*, which allows also (by integrating the software *lame*, see below) to produce mp3-files. The actual version of *Audacity* (Audacity 1.3.13 installer, .exe file, 13.8 MB, for Windows 2000/XP/Vista/7) including help files can be found for downloading under

http://audacity.sourceforge.net/download/beta_windows

Software recorders are basically handled the same way as their tangible counterparts, i.e. usual tape recorders. You just need to connect a microphone to the "Micro" jack on the sound card or somewhere on one of the sides of your notebook and start the software-recorder. There is a button symbolizing a record key. Just click on it with the mouse and press down the "Play" key of your tape deck (or from which device ever your audio data come). To terminate the recording, click on the software recorder's button "Stop" and save your digital audio file with an appropriate name. You will find this file then on your hard disk as an uncompressed audio file with the extension "*.wav" added to the name you have chosen for saving it. However, you should be aware that in this format

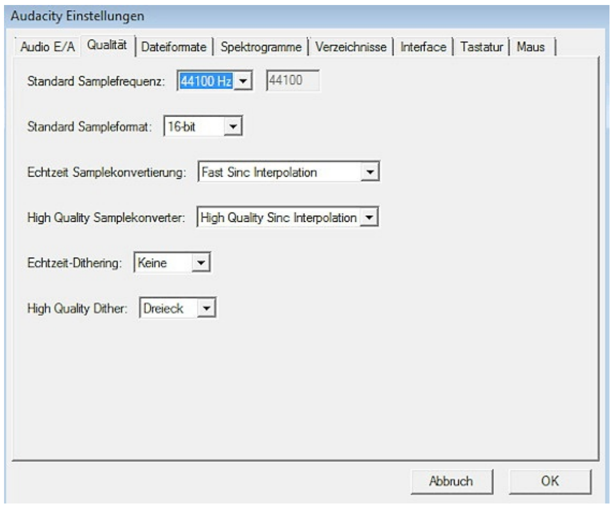
74 minutes (about a complete CD) of sound recording will need
about 650 Mbyte space on your hard disk.

However, even large hard disks reach the limit of their storage capacity, if you analyze many audio-taped interviews – nothing to say about video recordings. Therefore, digitized recordings should be converted from the original *.wav*-format into the common compression format *MP3* (exactly: MPEG-I Layer 3). If you are familiar with *MP3*-conversions, just skip the following paragraphs.

The audio format *MP3*, a development of the Fraunhofer Institute, allows to compress audio files by a factor of about 10 without reducing notably their quality. Conversion routines let you decide, which level of quality and consequently how much storage space you want for your audio files. As an example, a complete CD = 74 min play time contains 70 MB of sound data with a quality level of 128 kbp/s (kilo bit per second). This level corresponds more or less to the quality of regular CDs and is totally sufficient for sound documents. You see, compared to *WAV*-files, *MP3*-files need about 10 times less disk space. If you work already with an MP3-encoding program, just continue to use it for your AQUAD project. Most of these programs allow direct compression and recording in *MP3* without the detour of recording first in *WAV* format. Some sound cards come with software of this type. In case you don't have this kind of software, we recommend the free mp3-encoder *lame*. You will find it in the internet under

http://lame.sourceforge.net/ or
http://lame1.buanzo.com.ar/

Among the files of *lame* there is a file named *lame_enc.dll*. This file is used by *Audacity* to convert *wav*-files and export them as *mp3*-files. The effect is remarkable: The 25 seconds of recording in the file "example.wav" (See the following screenshot) needs 4.4 MByte of memory, the converted file "example.mp3" only 0.4 Mbyte.


Audacity Einstellungen

Audio E/A | Qualität | Dateiformate | Spektrogramme | Verzeichnisse | Interface | Tastatur | Maus

Standard Samplefrequenz: 44100 Hz 44100

Standard Sampleformat: 16-bit

Echtzeit Samplekonvertierung: Fast Sinc Interpolation

High Quality Samplekonverter: High Quality Sinc Interpolation

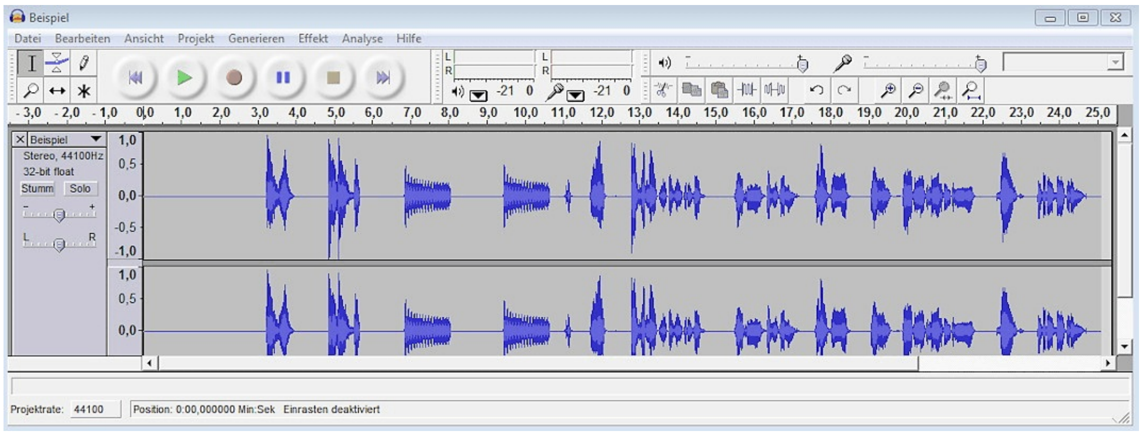
Echtzeit-Dithering: Keine

High Quality Dither: Dreieck

Abbruch OK

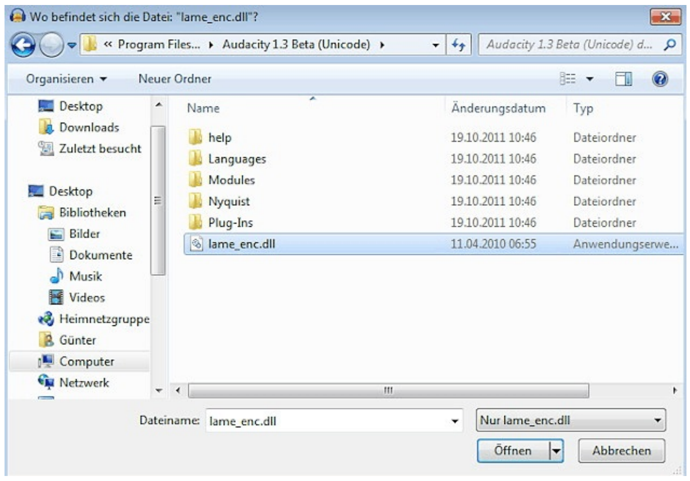
Below in the screenshot of a *Audacity* you see the usual buttons of an audioplayer and -recorder. The red button you can start directly digital recordings (via microphone!) or you may transfer an analog recording (Line-in from a cassette player) and digitalize it. In the following we assume that there is already a digital wav-file available.

The second group of functions "Bearbeiten" allows to set modalities for input and output as well as the quality of the recording. 44100 Hz and 16 bit are sufficient as standard sample format.


Beispiel
Datei Bearbeiten Ansicht Projekt Generieren Effekt Analyse Hilfe
Beispiel
Stereo, 44100Hz
32-bit float
Stumm Solo
Projektrate: 44100
Position: 0:00,000000 Min:Sek Einrasten deaktiviert

At the left you select within the group "Datei" (files) the function "Öffnen" (open) and select by clicking with the left mouse button the directory and the file name, here "example.wav".

Then you select within the group "Datei" the option "Export as mp3" and start the procedure. If you use *Audacitiy* for the first time, the program asks, where it can find the codec *lame_enc.dll* (see screenshot on the next page: "Where is the file lame_enc.dll?"). We have copied it directly into the directory of *Audacity:*



Name	Änderungsdatum	Typ
help	19.10.2011 10:46	Dateiordner
Languages	19.10.2011 10:46	Dateiordner
Modules	19.10.2011 10:46	Dateiordner
Nyquist	19.10.2011 10:46	Dateiordner
Plug-Ins	19.10.2011 10:46	Dateiordner
lame_enc.dll	11.04.2010 06:55	Anwendungserwe...

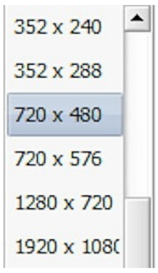
Dateiname: lame_enc.dll
Nur lame_enc.dll

The we can start the process of conversion /export. As result you will find a *mp3*-file on your hard disk. This file can be opened in AQUAD and coded.

4.3 Videos as data

Converting video files is just as simple as handling audiofiles – if you apply adequate software, so called video converters. Since today digital video recorders are standard, we do not describe how to prepare analog videos for data analysis.

In Europa the TV-norm PAL is standard. Depending on your recorder you can produce videos with various resolutions. Usual video resolutions (in pixel) of PAL-videos are



352 x 240
352 x 288
720 x 480
720 x 576
1280 x 720
1920 x 108(

Considerung theamount of memory necessary to store videos with higher resolutions, we suggest to work with dimensions of 720 x 480 or 720 x 576 pixels – absolutely sufficient for video analyses. AQUAD expects in ist videos 25 frames per second (fps). You have to determine this value when converting and compressing videos from a camcorder or a digital camera.

How to handle videos recorded under the TV-norm NTSC and ist problems of "inverse telecine" and "de-interlacing" cannot be discussed in this manual. In case you have American videos, please read the web pages listed below, where you find information how to convert NTSC videos.

Video recordings consist of two components: the actual visual data, that is, the video stream, and the corresponding sound data, that is, the audio stream. The components of digital recordings are processed (for instance, compressed) simultaneously. But you need much empty space on your hard disk until the compressed end product is available. You may reckon on about 20 GB (Giga-Bytes!) for a 1½ hours video recording in DV quality (720 x 576 pixels).

For practical reasons, the original video files should be converted into a less space consuming format. The reduction in quality of visual and sound information can be neglected, if we chose appropriate parameters (details see below) for the conversion procedures – and do not want to broadcast our recordings, but analyze them assisted by AQUAD. The audio component is converted to the format *MP3* (see above), for the video component we apply a codec named *DivX*. Both components are reunited in WINDOW's video format *AVI*. You can watch the final product, that is, the *AVI*-file in AQUAD, but also with the tool "Mediaplayer" that comes with WINDOWS.

Principally, *AVI* is just sort of a box or exactly a *wrapper*-format that determines *how* the audio stream and the video stream are arranged within the file – without demanding explicitly which codec, that is which algorithm of audio/video compression you have to apply. It is up to you to make this decision. In this manual, we confine ourselves to describing the combination of *MP3/DivX* and tell you, which soft- and hardware is necessary. How much time you will need depends on the speed of your PC and its components. In this respect, quantity – here the CPU's speed expressed in Ghz, memory and hard disk capacities – really is an advantage, particularly if you apply compression codecs (like DivX).

After converting your data, you can burn all audios/videos on CD or DVD, but we **strongly advise against (!!)** analyzing these data from CD or DVD in AQUAD. During this work, you will over long periods of time listen to or watch the same scenes again and again and compare them with some similar or related scenes, that is, the CD/DVD drive has to read short data segments at various locations on the disk surface again and again. These drives, however, are built for continuous play and wear out much faster then expected, if you use them for "stop-and-go" data traffic. Therefore, you should analyze your data from the hard disk and apply CDs or DVDs only to file away your data for security reasons (which is strongly recommended!).

What we do not treat here is cutting and editing of videos. In general, these operations are not necessary if you are just going to analyze your videos – and because of the authenticity of your data you should refrain from these operations. During the work with AQUAD you select critical scenes, mark and interpret them by adding a code. Besides, most camcorders come with editing software.

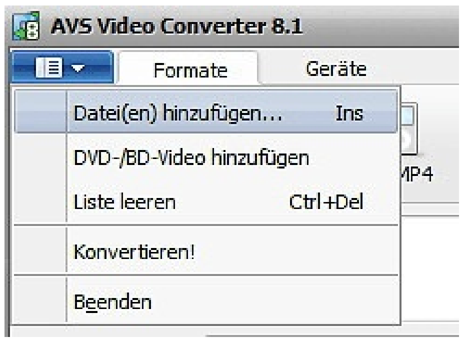
What we have to elaborate on are compressing the initial filses and converting them into the format AVI. For this end we have experimented with several free programs. However, handling videos with these programs is either somewhat complicated or the possibilities to set important parameter values is somewhat limited. Therefore we frequently run into problems when playing the converted files in a mediaplayer. That is why we recommend to buy an inexpensive software tool for converting videos.

On our own videos in format MOV from a digital photo camera we worked with *AVS Video Converter* (actual version, end of 2011: 8.1). A trial-version, which adds a watermark in the middle of the screen can be downloaded free of charge, activating the regular version without any time limit costs (end of 2011) € 39.- + AVT. Most interesting about this license offer is that you may download additional software of AVS without additional costs. Among these tools is a video-editor, a video-recorder, a video-player, an audio-converter, an audio-recorder, software for editing photos and much more tools. The address to download AVS software is

http://www.avs4you.com/de/downloads.aspx

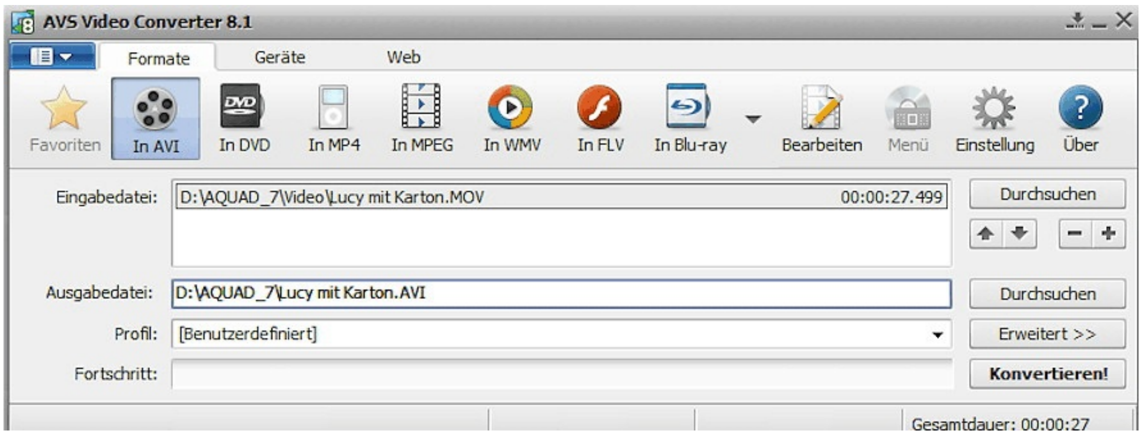
Subsequently we show the settings for converting a video of ca. 27 seconds "Lucy with a box" ("Lucy mit Karton.mov"), which needs about 103 MByte of memory.

The screen shot is in German, because we use this language version of the *AVS Video Converter*.



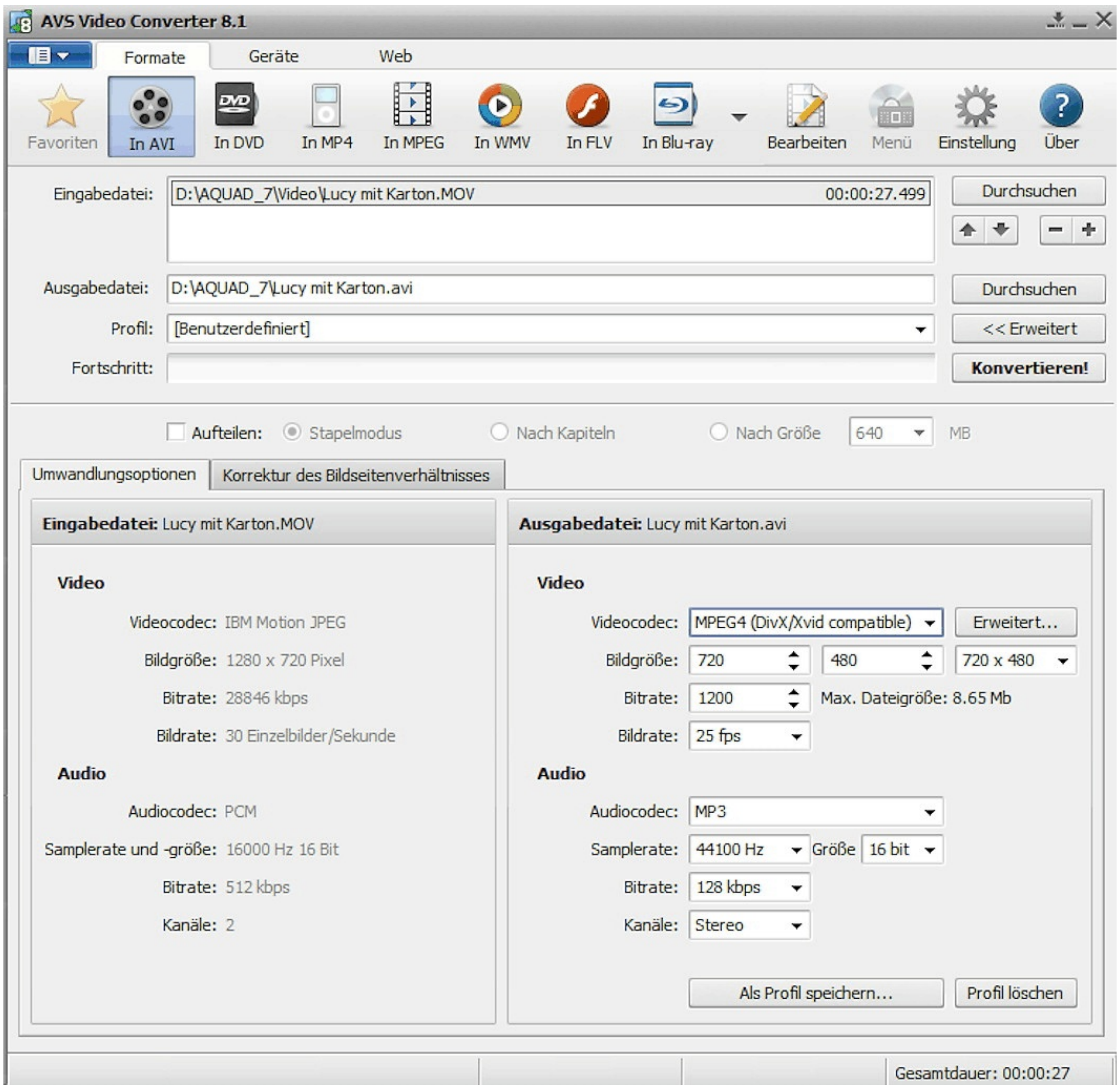
From the group of file-functions (Icon in the left upper corner) we select the option Option "Add file(s)" ("Datei(en) hinzufügen") and look on the hard disk for directory and name of the file we want to convert.

As you can see on the next page we have chosen the input files "D:\AQUAD_7\Video\Lucy mit Karton.MOV" and as output file "D:\AQUAD_7\Lucy mit Karton.AVI". Additionally we have marked the format into which we want to convert the video, here (see the icon on the left in the icon bar) "In AVI".



Eingabedatei: D:\AQUAD_7\Video\Lucy mit Karton.MOV 00:00:27.499
Ausgabedatei: D:\AQUAD_7\Lucy mit Karton.AVI
Profil: [Benutzerdefiniert]
Fortschritt:
Gesamtdauer: 00:00:27

However, before we click on the button "Convert" ("Konvertieren"), we have to set the conversion profile according to our needs:



Eingabedatei: D:\AQUAD_7\Video\Lucy mit Karton.MOV 00:00:27.499
Ausgabedatei: D:\AQUAD_7\Lucy mit Karton.avi
Profil: [Benutzerdefiniert]
Fortschritt:

Aufteilen: Stapelmodus ○ Nach Kapiteln ○ Nach Größe 640 MB

Eingabedatei: Lucy mit Karton.MOV

Video
Videocodec: IBM Motion JPEG
Bildgröße: 1280 x 720 Pixel
Bitrate: 28846 kbps
Bildrate: 30 Einzelbilder/Sekunde

Audio
Audiocodec: PCM
Samplerate und -größe: 16000 Hz 16 Bit
Bitrate: 512 kbps
Kanäle: 2

Ausgabedatei: Lucy mit Karton.avi

Video
Videocodec: MPEG4 (DivX/Xvid compatible)
Bildgröße: 720 480 720 x 480
Bitrate: 1200 Max. Dateigröße: 8.65 Mb
Bildrate: 25 fps

Audio
Audiocodec: MP3
Samplerate: 44100 Hz Größe 16 bit
Bitrate: 128 kbps
Kanäle: Stereo

Gesamtdauer: 00:00:27

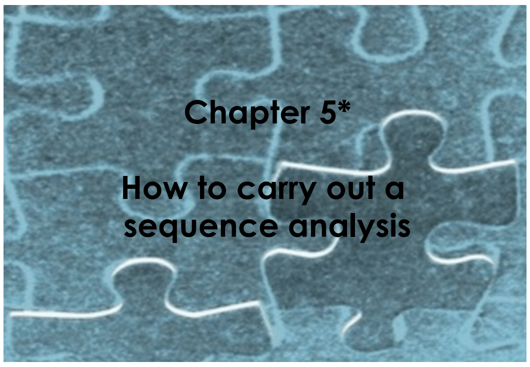
Comparing the settings for input and output you see that we did not only select the DivX/XviD-Codec for compression, but we reduced also the size of the frames (720 x 480), the bit rate (1200), the frame frequency (25 frames/second) and the bit rate for the sound (128 kps). These settings may be saved for further conversions (see the button below "Save as profile..." ("Als Profil speichern...").

In case the *XviD-Codec* is not yet installed on your computer, you may download it free of charge under

http://www.chip.de/downloads/Nic-s-XviD_13008478.html

or under various other addresses, which you can easily find, if you have a search engine look for "xvid codec". Alternatively you can apply the original DivX codec, which is available also in a free version.

Now finally we can click on the button "Konvertieren!" and we will receive after ca. 33 seconds an AVI-file that needs only 4,75 MByte. This file can be used immediately in AQUAD.

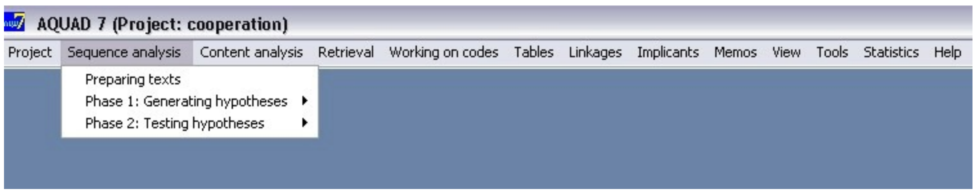


Chapter 5*

How to carry out a sequence analysis

The goal of a sequence analysis is to develop a hypothesis about the structure of the case. This done stepwise, taking each text segment in its natural order within the text and generating all possible hypotheses about its meaning and check these hypotheses later. The finally resulting case structure allows to formulate general statements way beyond the individual psychological state of the acting persons, because their communication and interaction is guided by a common and general linguistic-cultural system of rules. The module "sequence analysis" in AQUAD organizes the sequence-analytic work on the researcher's data into three parts:

(1) As a pre-condition for a sequence analysis the texts have to be divided into segments.
(2) Then the researcher generates successively for each segment all hypotheses s/he can imagine.
(3) Finally, using the complete available text, these hypotheses are tested critically.



5.1 Preparing the texts

Working in the sequence-analytic mode our interpretation follows the sequence of the events or interactions exactly in the order, in which they are documented in the text. Wernet (2009, p. 27) underlines the importance of a "basic attitude of interpretation" that "takes the text seriously as text" and does not use it "as a quarry of information or a fair, offering a range of meanings" or even "cannibalizes" it. This is due to the goal of the method of sequence analysis to detect within the texts, that is within "protocols of real, symbolically mediated social actions and interactions" (Oevermann et al., 1979, p. 378) the latent structures of meaning and to contrast them with the manifest content: "Texts of interactions constitute objective structures of meaning based on rules that can be reconstructed; these *objective structures of meaning* represent the *latent structures of sense* themselves" [italics in the original]. They are the analytical (although not empirical) reality (and are long-lasting) independent of the concrete intentional representation of the meanings of interaction on the part of the individuals involved" (Oevermann et al., 1979, p. 379). That is, we do not know anything about the intentions of these people, but we are able to reconstruct their motives by analyzing the text protocols. As a consequence the procedure is preferably realized in a group setting, so that highly idiosyncratic and subjective interpretations are ruled out.

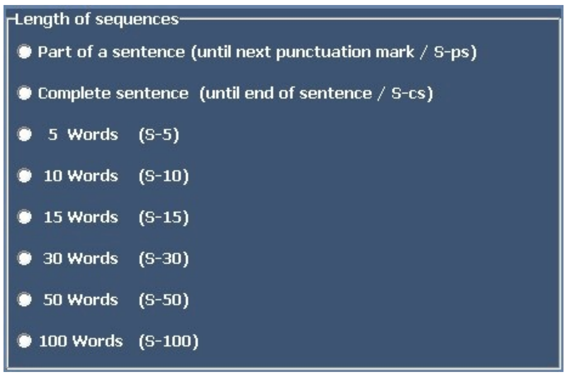
* Two examples and explanations of contents are taken from Gürtler, Studer and Scholz (2010) and Studer (1988).

We aim at inter-subjective valid interpretations and not on subjective images without any empirical base. In a group setting it is much easier to exclude personal ignorance, because we assume that every individual's personal ignorance is different. Other colleagues are less ignorant on topics where we are rather blind, and vice versa.

According to this position, we cannot deduce the general meaning of a text/interaction protocol by trying to fathom out the participants' motives, intentions, norm orientations etc., i.e. the "intra-psychic reality of the acting subjects" (Oevermann et al., 1979, p. 379). Instead, we try to reconstruct the text's objective structure of meaning. This structure of meaning exists independently of psychic states, viewpoints and intentions of the acting subjects as "social reality", as a "reality of possibilities" (Oevermann et al., 1979, p. 368), which are at the subjects' disposal because of the given system of social norms and rules – or they can be concluded from transgressions of these rules. The objective structure of meaning or latent structure of sense has to be studied independently of subjective meanings and before these subjective meanings are taken into consideration. Meaning is for Oevermann et al. (1979, p. 380) an objective social structure that appears within interactions – and which "for its part has to be taken as precondition for any intentions." This, on the other hand, implies that the way the actors see themselves, their intentions and motives *do* play a role in the process of interpretation, but only against the background of the latent structure of meaning of an interaction – that is, after the sequential analysis.

Oevermann et al. (1979, p. 354) understand this approach as "a rather simple perspective that arguments with really trivial assumptions." However, from these assumptions follow strict methodological consequences. Above all we have to observe "that no information and no observation from later interactive episodes is used to interpret a previous interactive episode" (Oevermann et al., 1979, p. 414). Or in Wernet's (2009, p. 28) words: "You are not wandering through the text looking for useful passages, but you follow the protocol step by step." And in addition: "It is absolutely important for a sequence analysis *not to pay attention* to the text that follows a passage that is interpreted at the moment" [italics in the original]. This does not mean that we have to interpret a text beginning with the first sentence or not to omit any passages, but those passages selected as relevant for the research question must be interpreted sequentially. This ensures a serious scientific work in which we work stepwise: hypothesis formulation – hypothesis testing. We express thoughts about reality and then check (proof) these against what actually happened (in the text).

Therefore, as a first step, the text has to be subdivided into a sequence of text segments. The following criteria are offered for this work in AQUAD:


Length of sequences
- Part of a sentence (until next punctuation mark / S-ps)
- Complete sentence (until end of sentence / S-cs)
- 5 Words (S-5)
- 10 Words (S-10)
- 15 Words (S-15)
- 30 Words (S-30)
- 50 Words (S-50)
- 100 Words (S-100)

AQUAD allows to subdivide the text according to syntactic rules, that is to dismantle a text by parts of sentences (criterion is always the following punctuation mark) or to take it to pieces in form of complete sentences. In addition there are six mechanical operations, which determine a new segment always after a particular number of words. The example, which was installed together with AQUAD, was subdivided into complete sentences. To the name of the text "*communication protocoll.txt*" a preceding part "{*s-cS*}" was added to identify the new, sequentially structured text: "{*s-cS*}*communication protocoll.atx*" (the extension "*atx*" shows that

AQUAD has structured the text). Correspondingly, "{*s-pS*}" denotes a text subdivided into parts of sentences, "{*s-5*}" segments with a length of five words each, etc.

5.2 Generating hypotheses

5.2.1 About the theoretical background

In chapter 1.3 we stated that for Oevermann et al. (1979, p. 378) texts are all forms of "protocols of real, symbolically mediated social actions or interactions, whether they may be written, acoustic, visual, in various media combined recordings or specifications that can be otherwise put into the archives." In short, a "text" as defined by Overmann and colleagues can be everything: a written text, audio/speech, video, pictures. A text is a protocol of something that really happened and which can be investigated with care.

Oevermann (2002, p. 33) explains that "in a sequence analysis like in real life principally a decision has to be made among the still open options for an open future." The sequence analysis "follows the sequential nature, which constitutes human actions" (p. 6). However, the sequential course of analysis does not mean just to work from the beginning to the end. Rather the task is to open new possibilities or "versions" strictly following the flow of the text – and to close them again, if they do not stand up to a closer examination in the light of later text segments. Thus, sequence analysis is a process of alternating between possibilities or "open options for an open future" (cf. above) and reality, that is to find these options again within the text.

As also already mentioned, Wernet (2009, p. 39) gives a simple "answer to the question: What do I have to do, if I want to realize an operation of reconstructing meanings according to given rules, which can be verified methodologically?" The answer consists in a methodological triple jump. In the case of generating hypotheses, the first two of these steps are important: "I have (1) to *tell stories*, (2) to *form versions* ... ." The third step, that is confronting the versions with the real context, will become important during the phase of checking the hypotheses according to the principle of falsification.

In this process we have to pay attention to the unique individual experiences of people, but above all to their objective reality, which we can be revealed in form of their individual, social, culturally embedded patterns of action. The strict orientation on reality guides the procedures of Objective Hermeneutics. The base is the general (normal) case, which becomes public and generally accessible by reconstructing the latent structure of meaning. Only subsequently the concrete manifestation, the expressions and statements of the empirical person and her subjectivity can be interpreted more precisely.

To illustrate these abstract theoretical position we quote an example taken from a potential client's letter of application for a therapy against drug abuse. We can read there a widely used formulation of therapy motivation:

"I am still very interested to be able to handle my addiction."

We can read this text segment as "I want to be freed from my addiction" or "I want to be able to live abstinently," however also as "I want to be able to control my drug consumption" – because what one is able to handle one must not let go. Further analysis of this letter of application (see Studer, 1996) shows indeed that in the case of this potential client the motivation "to handle my addiction" is focused on control, but not on overcoming completely her drug dependency. The client wants control to avoid serious damages by her drug consumption. This reconstructed knowledge about latent meaning structures is most helpful for the practical work, because they are relevant for the client's everyday activities. For the time being here motivation to change her life has to be taken seriously and appreciated. However, in addition the frame of possibilities has to be explored to avoid therapeutic illusions and to adapt individually the possible therapies. As an illustration we present a text segment from the letter of application of another client (see Studer, 1996):

"But we had also very good conversations ..."

We can read this segment as a further example ("... we had *also* ...") for the client's appreciation of her partner. But we can also read ("*But* we had ...") that the writer of the letter was considering two different things at the same time: "On the one hand my partner got on my nerves, on the other hand we could talk very good with each other." The further analysis shows that the writer does not get apart these two ideas and mixes the contradictory intentions of convergence and detachment. In the practical work with the client, separating these ideas opens for her the space of possibilities to detect new aspects of herself. But at first one has to understand what is the actual topic. Which action-relevant reality is behind the client's verbal expression? What is motivating her actions?

Oevermann (2002, p. 33) explains: "The sequence analysis nestles up to the basic structure of the real human-social events and is therefore not like the usual procedures of measurement and classification a method external to its object, but a method corresponding and adequate to it. Indeed, also in real life we have principally to decide at each point in the sequence among still open options for an open future." The sequence analysis "...follows the sequential character, which is constituent for human action" (2002, p. 6). But demanding sequential processing does not mean to work just from the beginning to the end. Instead, following strictly the sequence of the text, the task is to open new possibilities (the term "versions" is used) and to close them again, if they cannot be checked successfully. A sequence analysis is the interplay of possibility (of "open options for an open future") and reality.

5.2.2 Principles for generating hypotheses

As we already quoted, Wernet (2009, p. 39 ff.) gives a simple "answer to the question: what do I have to do to reconstruct the meaning of a text by a methodologically controllable operation according to valid rules?" The answer describes a methodological triple jump or sequence of "*telling stories*" (stories in which the text segment could occur), "*forming versions*" (that is, to sort the stories by comparing them according to similarities and differences) and "confronting these versions with the real context." This last step, i.e., the confrontation of versions with the real context, wil be the content of the following section 5.2.3 on testing the hypotheses.

Before we start to interpret the segments of a text, we have to clarify *what the case is* and in which context it is embedded. According to Wernet (2009) this includes on the one hand to reveal one's own research interest, and on the other hand to clarify what at all is recorded in the transcript, which social reality it describes or more concretely, what for instance an interview may contribute to answer the research questions. We start with the objective data of the case (birth, milieu, professions, date of death, etc.) and only afterwards the real text (classroom interaction, therapy recordings, biographic intervews, etc.) is analyzed.

This analysis is guided by the five rules of (1) independence of context, (2) literal account, (3) sequential order, (4) extensive interpretation or totality and (5) austerity (see Wernet, 2009, p. 21-38):

Independence of context:

Of course the interpretation of a text has to take into consideration the context in which the protocol was recorded – but only after exploiting possible meanings of a text segment independent of this context. That is for the time being one invents stories, in which the critical text passage could make sense. Wernet talks about designing "contexts as thought experiments" to generate in a first access to the text possible meanings of the text segment. Wernet also reminds the researcher to put aside any pre-existing knowledge, as otherwise the interpretation would depend on the interpreter's everyday knowledge or from his/her non-scientifically constructed knowledge. Thus, a circular process of interpretation would be started.

Literal account:

The interpretation is oriented exactly, word for word and exclusively at the actual and provable content of the text – not at something that the text maybe would have liked to express. The text per se is part of the reality and has to be handled like that. Two important conditions follow from this principle:

(1) The text protocol has to be given – really quite natural – in the original version, for instance as a literal transcription of a social interaction, not in form of a paraphrase that is modified by everyday knowledge and social conventions, at least reduced in its possible meanings. Paraphrases, on which sometimes qualitative content analysis is based on, are absolutely inadequate for a sequence analysis.

(2) The interpretation has to treat the text really finicky. We must not overlook clumsy or apparently inappropriate details, as we would do under everyday circumstances. Wernet demands, "to *weigh the words* in a way ... that would appear petty-minded" in everyday life.

This approach that would be inappropriate, maybe even impudently critical in everyday conversations, makes the latent meaning behind the manifest formulation accessible. If we must conclude finally that there is more than one possible interpretation, the literal interpretation at least has revealed divergences between what was said and what was meant and thus also revealed the ambiguity of the text and discrepancies in the life of its speaker.

Another important aspect of literal account is mirrored in the fact that researches should not hold back promising hypotheses because of moral considerations or other personal reasons. There should not be any taboo while formulating hypotheses. Therefore, research subjects themselves should never be part of the analysis team. They could perceive some hypotheses as insulting.

Sequential order:

This principle defines the core element of the analysis, because the procedure is strictly logical and follows a step-by-step orientation. Thus the method gains scientific quality. We do not look unsystematically for evidence for possible interpretations across the text (looking for confirmation). Rather we alternate in a strict sequence between generating hypotheses, condensing them in versions of understanding and checking them within the same text.

Were exactly the interpretation starts, which text segment opens the sequence of interpretation has to be justified based on the content. That is, we have not necessarily to start with the first sentence or the first part of this sentence. Then, however, we follow strictly step by step the order of the protocol. What comes after the actual segment must not be taken into consideration – for the time being. Of course, every text segment is embedded in the "internal context" of the meaning constructed up to now. Not taking into consideration each time the following segment is most important from methodological and practical points of view: "The continuing thought experiment makes clear that the concrete case has to come to the 'decision' to be what it is ... ." In this cryptic words Wernet underlines that he refers to "... the reconstruction of a practice of life that has 'become like this and not otherwise'."

The principle of sequential order does not prohibit to skip over text segments and to select relevant text passages depending on the research question and the progress of reconstructing the meaning structure of the text. However, each new start has to be justified again and the interpretation must again follow strictly the rules of sequence analysis from this point on. Orientation for the selection of later passages is given by the fact that these segments confront or support the versions of interpretation.

Sequential order means also that no pre-existing knowledge about later developments in the text must influence the interpretation, but just the step-by-step development. It is important not to search only confirmation of hypotheses, but rather explicitly evidence that may refute them. This corresponds with the procedure of falsification in the approach of critical rationalism (Popper, 1934). The reconstruction of meaning has to hold exactly on to the text and has to be checked there. However, we also find approaches to stabilize empirically one's own ideas instead of putting them on trial. Thus, a fair checking of competing hypotheses is doubtful.

Extensive interpretation:

Extensive interpretation or totality is a balancing principle preventing that we follow our subjective or arbitrary intentions when we interpret the text. It demands that apparently unimportant details are overlooked and on the other hand not to concentrate only on apparently important passages. Considering a passage as "important" or "unimportant" are consequences of subjective prejudices. Here again we are reminded to analyze the text stepwise without preferring or discriminating particular parts. All parts of the text must be treated equally. Above all, "... the types of contexts of the thought experiment have to be clarified completely... ."

Austerity:

This principle demands that we suppose normalcy, that is everyday life and its routines, while we construct possible interpretations of life practices. We must "... allow only those versions that can be checked based on the text." Thus the "story telling" is limited to versions that are compatible with the text as a whole. The principle of austerity excludes science-fiction stories, esoteric excursions and above all not justified attributions of pathological deviations from normal behavior in interpretations of life practice.

Deviations from norms, often taken as pathological, attract the attention in protocols, because the variance of possible interpretations is drastically limited. Routines or practicable problem solutions are less prominent, because they open a broad spectrum of possibilities. That is why normalcy is harder to reconstruct than deviations from norms. That is why we should understand "austerity" in a double sense: On the one hand we should always consider normalcy as given and we have to justify any assumption of deviations. We want to detect really important deviations from the normalcy of human actions. On the other hand is the space of possibilities limited by the fact that we interpret only the available text and have to base all conclusions on this text, not on everyday experiences.

We may summarize as guideline: What we interpret must be substantiated in the text – and what is substantiated must be included in the interpretation. Above all – we assume normalcy. The text must prove that this assumption is false.

5.2.3 Combining hypotheses in text versions (grouping of hypotheses)

The rules for generating meaning, which produce at any point in the text sequence always again various options, consist of a set of algorithmic-typologic rules. As examples Oevermann (1996b, S. 7) lists the syntax of language, pragmatic rules of speaking and acting and logic rules of formal and content-oriented conclusions (induction, abduction, deduction; cf. Reichertz, 2000). Oevermann refers to interpretations developed from these options or possibilities as "well-formed expressions," that is grammatically correct expressions or completely normal sentences.

These sentences or stories according to Wernet (2009; p. 39 ff.) are grouped or sorted. The groups determine types of stories or versions of the text. In AQUAD Seven grouping is facilitated by the operations behind the button "Subsuming hyp." (see above, fig. 3).

Once this step is finished and *only after* this step these versions are confronted with the reality of the complete text. In this way the interrelation of the general or abstract structure (independent of the particular case) and the concrete, particular practice of life becomes clear and outlined more exactly.

5.3 Testing the hypotheses

After reconstructing the elementary structure of the text we enter the phase of checking the hypotheses. Technically spoken we follow the principle of falsification (Popper, 1934). That is, we do not keep a hypothesis because we could verify it, but because we could not prove the contrary , namely that it cannot be applied in the given case. There is no forever valid truth, but there are hypotheses for which we could not yet establish proof that they are invalid. Checking a hypothesis comes down to try to prove and substantiate that this hypothesis and the text are incompatible.

In this phase we do not have to proceed stepwise or in sequential order. When we search for instances supporting the stories or the versions of text, which we have produced during the phase of generation of hypotheses, it is necessary to "wander" around in the text – and this in a very specific way exactly to find counter-evidence.

Difficulties in checking the hypotheses are mostly due to previous infringements of the rules of text interpretation. Maybe we have not written down and pursued risky assumptions, although precisely the possibility of failure contributes to the explanatory power of an assumption. A hypothesis is strong and has a larger realm of validity, if it takes the risk to fail even because of tiny discrepancies. A hypothesis that cannot be refuted principally is worthless from the scientific point of view.

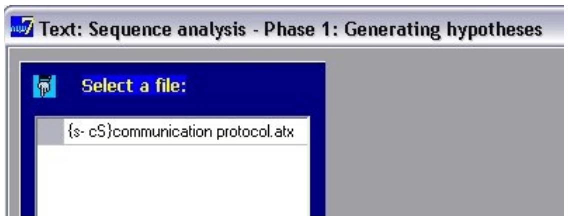
We want to recourse here to the recommendations for the generation of hypotheses and complete them by giving the hint that any possibility – as small it may be – to understand a text in a particular way should be studied. Of course, thus the procedure becomes rather costly, however then a very small sample is sufficient to cover a whole field of research. Oevermann personally goes as far as to claim that it is able to formulate general conclusions from a single, well analyzed case. Hildenbrand (2006) takes about eight or ten cases as a starting point necessary to reach a theoretical saturation in the sense of Glaser's (1998) strategy of grounded theory; more cases do not promise additional insights.

When may we end the sequential analysis? Oevermann (1996) advises to analyze until we reach the point at which the case structure is completely repeated. The internal logic is reproduced anew in every passage, therefore it is sufficient to study a few text passages very precisely and nevertheless gain a general view of the case. This stability justifies to analyze (only) until the (first) repetition of the case structure – and then search for counter-evidence. As a result of this analysis we learn about the manifest and obvious as well as about the latent meaning structures of daily routines in standard situations (Oevermann, 1996, p. 76 f.).

5.4 Generating and testing hypotheses in a sequence analysis with AQUAD

In paragraph 5.1 above we already described how to prepare your texts for a sequence analysis. Subsequently we explain how to generate and test hypothetical interpretations. As an example we use the text "*communication protocol.txt*" that was already copied into the root directory of AQUAD (for instance, "C:\AQUAD_7") during the installation of AQUAD Seven. Now we have AQUAD divide this text (applying the function "Preparing texts") into complete sentences. Afterwards the newly formatted text is ready for further analysis as file "*{s_cS}cooperation protocol.atx*" in the subdirectory "C:\AQUAD_7\cod" (in our example).

5.4.1 Generating and grouping hypotheses


Text: Sequence analysis - Phase 1: Generating hypotheses
Select a file:
{s- cS}communication protocol.atx

We start from the main menu "*Sequence analysis*" –> "*Phase 1: Generating hypotheses*" –> "*Carry out*".

This opens a window, where you chose a file of your actual project. Well, the project "Communication" does not contain more than the one file "*{s_cS}communication protocol.atx*". You click on this filename.

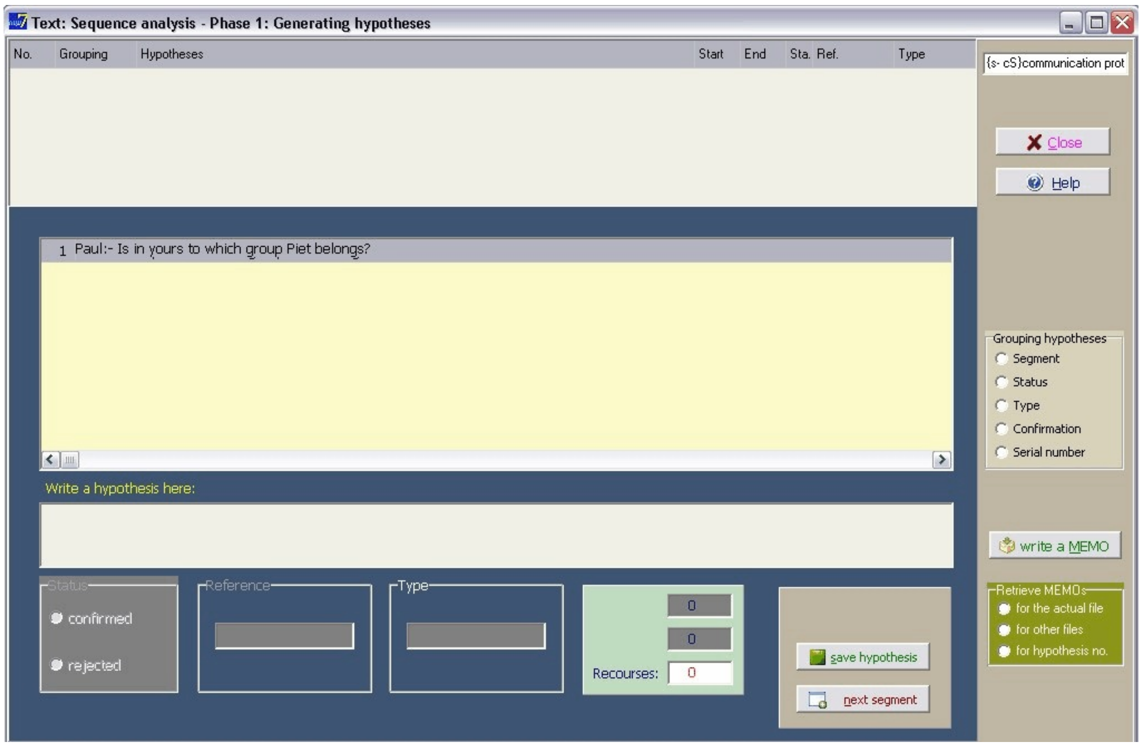
Now the window for entering hypotheses opens (see next page). It shows at the moment only the first text segment in the upper field within the broad blue frame:

Paul:- Is in yours to which group Piet belongs?

Below you see an empty window, the yellow headline of which asks you: "Write a hypothesis here." After you have done this, don't forget to click below on the right on the first button, which says in green letters: "save hypothesis." When doing so, the hypothesis gets a current number and is shown above in the window for hypotheses (see also second screen shot on the next page):

No..	Grouping	Hypotheses	Start	End	Sta.	Ref.	Type
1		Behind this question is Paul's hypothesis that information about Piet is relevant for the solution.	1	1			

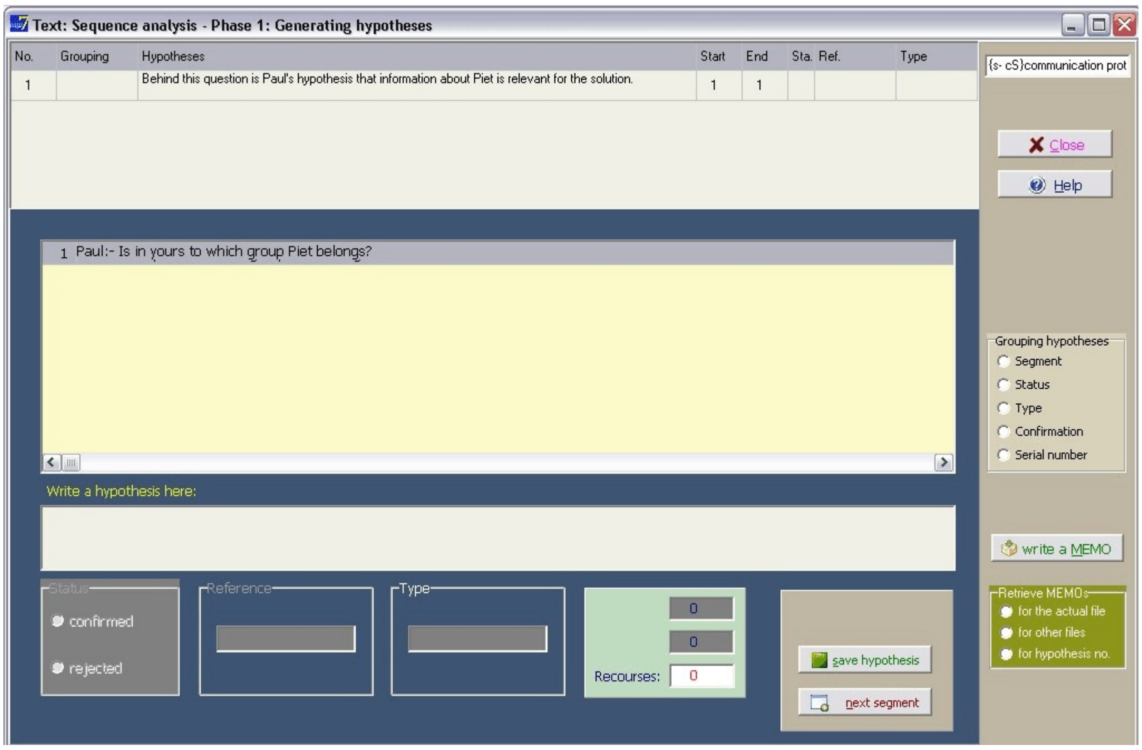
The column "Grouping" will show later when you try to group your hypotheses into various "versions" to which version this hypothesis was attributed. The name of the version is shown in the (last) column "Type" – in case you have entered something in the frame "Type" (below the hypothesis). "Start" and "End", here marked as "1", refer to the text segment, for which this hypothesis was generated. In column "Sta." (Status) you can later mark wether this hypothesis was confirmed or has to be rejected. "Ref." may contain references to other segments or hypotheses, which provide proof for the confirmation or rejection.


Text: Sequence analysis - Phase 1: Generating hypotheses

No.	Grouping	Hypotheses	Start	End	Sta.	Ref.	Type

1 Paul:- Is in yours to which group Piet belongs?

After clicking on "save hypothesis" the entry window is empty again, the hypothesis appears with its appropriate number in the upper part of the window. Now we may generate an additional hypothesis interpreting the first data segment in a different way:



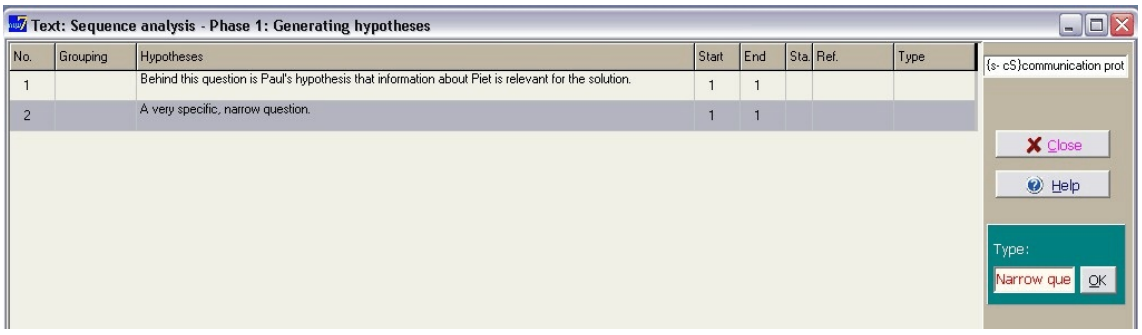
Text: Sequence analysis - Phase 1: Generating hypotheses

No.	Grouping	Hypotheses	Start	End	Sta.	Ref.	Type
1		Behind this question is Paul's hypothesis that information about Piet is relevant for the solution.	1	1			

1 Paul:- Is in yours to which group Piet belongs?

Write a hypothesis here:

Comparing the two screen shots on the preceding page you may conclude that the "Type" or the version can entered additionally once you have written a hypothesis into the entry window, but not yet clicked on the button "save hypothesis." However, you can change always later the type of a hypothesis, that is the version of reading this segment. To do so you move the mouse pointer within the line of the critical hypothesis into the cell of the column "Type" and click there. Now a small blue-green window opens in the gray field in the right side; there you can write the type (up to 10 characters) and transfer it to the hypothesis-table by clicking on the "OK"-button on the blue-green window. If you have entered only a blank character, an already defined type will be erased:

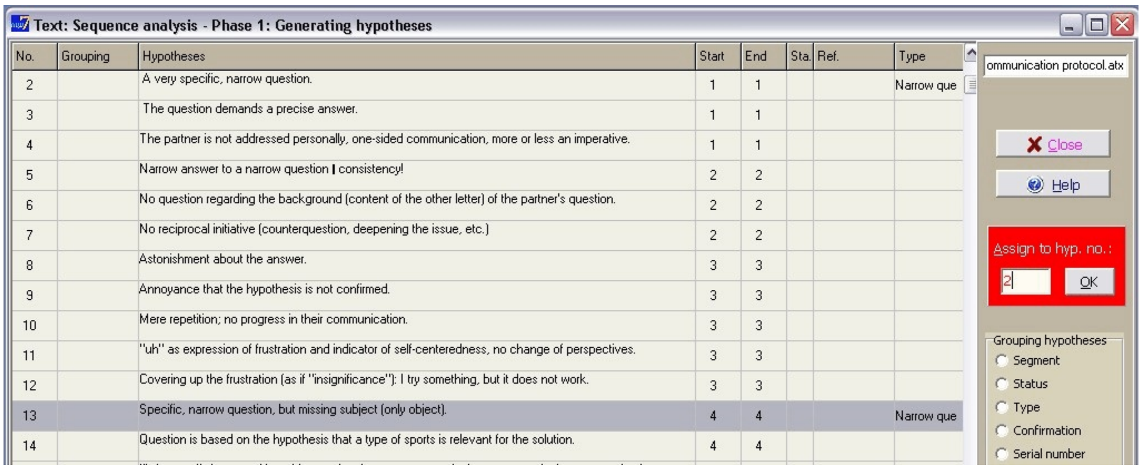


Text: Sequence analysis - Phase 1: Generating hypotheses

No.	Grouping	Hypotheses	Start	End	Sta.	Ref.	Type
1		Behind this question is Paul's hypothesis that information about Piet is relevant for the solution.	1	1			
2		A very specific, narrow question.	1	1			

Type: Narrow que

Whenever you click on the Button "save hypothesis" (green letters) the hypothesis you wrote into the entry window will be numbered and transferred to the hypothesis table above – and you can continue generating hypothesis about the actual text segment until no new idea comes to your mind or emerges in your work group.


Text: Sequence analysis - Phase 1: Generating hypotheses

No.	Grouping	Hypotheses	Start	End	Sta.	Ref.	Type
2		A very specific, narrow question.	1	1			Narrow que
3		The question demands a precise answer.	1	1			
4		The partner is not addressed personally, one-sided communication, more or less an imperative.	1	1			
5		Narrow answer to a narrow question I consistency!	2	2			
6		No question regarding the background (content of the other letter) of the partner's question.	2	2			
7		No reciprocal initiative (counterquestion, deepening the issue, etc.)	2	2			
8		Astonishment about the answer.	3	3			
9		Annoyance that the hypothesis is not confirmed.	3	3			
10		Mere repetition; no progress in their communication.	3	3			
11		"uh" as expression of frustration and indicator of self-centeredness, no change of perspectives.	3	3			
12		Covering up the frustration (as if "insignificance"): I try something, but it does not work.	3	3			
13		Specific, narrow question, but missing subject (only object).	4	4			Narrow que
14		Question is based on the hypothesis that a type of sports is relevant for the solution.	4	4			

ommunication protocol.atx

Assign to hyp. no.: 2

At that point you click on the button labeled with red letters "next segment" and bring the next text segment into the text window; then you develop your hypotheses about this segment's meaning, and so on.

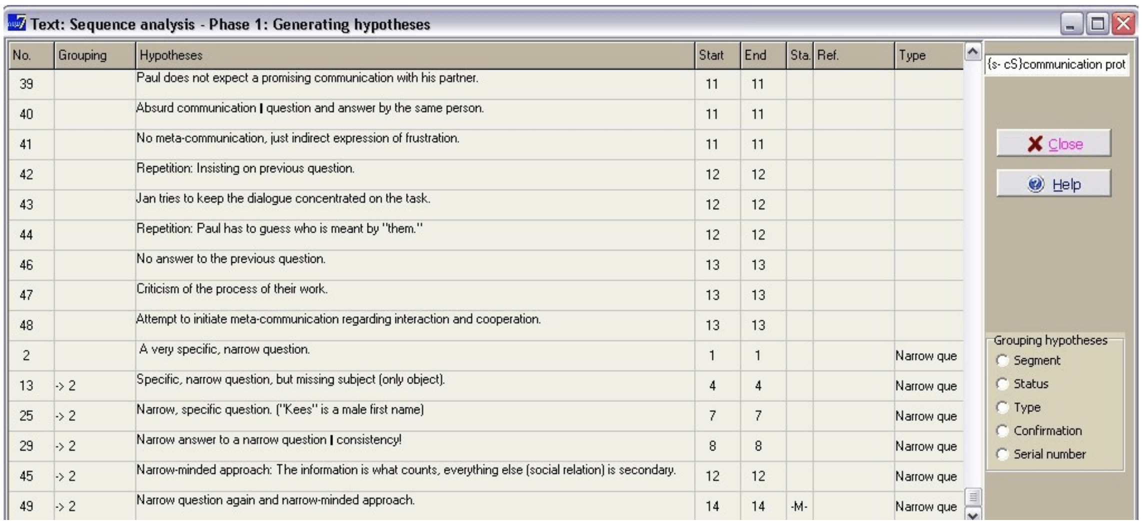
You can see the complete listing of hypotheses generated for this text in the example "Communication protocol" that was installed during the setup of AQUAD 7. If you open this project and activate "Phase 1: generating hypotheses" you will se that we terminated generating hypotheses after segment no. 14, because the sequence patterns begin to repeat themselves.

Now we may group the hypotheses according to versions or "types". For instance, we could group hypotheses no. 2 for segment 1 ("A very specific, narrow question") into one type together with hypothesis 13 (segment 4), 25 (segment 7), 29 (segment 9), 45 (segment 12) and 49 (segment 14). As type we enter "Narrow que."

To group the hypotheses we click in the row of the hypotheses, which we want to classify as belonging to the same type as another one, in the second column ("Grouping") into the empty cell. In the next screenshot you see that we are going to assign hypothesis no. 13 to hypothesis no. 2. This can be done, of course, independently of attaching a type name first. At the right side a red window opens into which we write the number of the hypothesis, to which the highlighted hypothesis (here no. 13) shall be attached (here no. 2). Clicking on the button "OK" realizes the operation. If we enter a blank character and click on "OK", an already existing attachment is erased.

The effect is visible in column "Grouping", which was empty, but shows now "-> *2*". In the same way we treat the other "narrow questions".

You get an overview on all hypotheses joined in the same version of reading the text segments, if you click on the otion "Type" in the gray box "Grouping hypotheses" on the right side. The hypotheses table will be sorted according to types. The last option "Serial number" restores the sequential presentation of your hypotheses.


Text: Sequence analysis - Phase 1: Generating hypotheses
{s- cS}communication prot
Close
Help
Grouping hypotheses
Segment
Status
Type
Confirmation
Serial number

No.	Grouping	Hypotheses	Start	End	Sta.	Ref.	Type
39		Paul does not expect a promising communication with his partner.	11	11			
40		Absurd communication I question and answer by the same person.	11	11			
41		No meta-communication, just indirect expression of frustration.	11	11			
42		Repetition: Insisting on previous question.	12	12			
43		Jan tries to keep the dialogue concentrated on the task.	12	12			
44		Repetition: Paul has to guess who is meant by "them."	12	12			
46		No answer to the previous question.	13	13			
47		Criticism of the process of their work.	13	13			
48		Attempt to initiate meta-communication regarding interaction and cooperation.	13	13			
2		A very specific, narrow question.	1	1			Narrow que
13	-> 2	Specific, narrow question, but missing subject (only object).	4	4			Narrow que
25	-> 2	Narrow, specific question. ("Kees" is a male first name)	7	7			Narrow que
29	-> 2	Narrow answer to a narrow question I consistency!	8	8			Narrow que
45	-> 2	Narrow-minded approach: The information is what counts, everything else (social relation) is secondary.	12	12			Narrow que
49	-> 2	Narrow question again and narrow-minded approach.	14	14	-M-		Narrow que

In the example installed together with AQUAD (Project "*communication*") only one type of hypotheses ("narrow que") is grouped up to now. You may continue the work and look for the version(s)/type(s) into which the each time following reactions of the interaction partner can be grouped.

Finally a suggestion from the field of text processing: If you want to enter several times the same type one after the other (here "narrow que"), then you mark the first time the entry with the mouse and copy it by pushing the combination of keys

"Strg + c" (c for Copy).

For the next entry you just push the combination

"Strg + v"

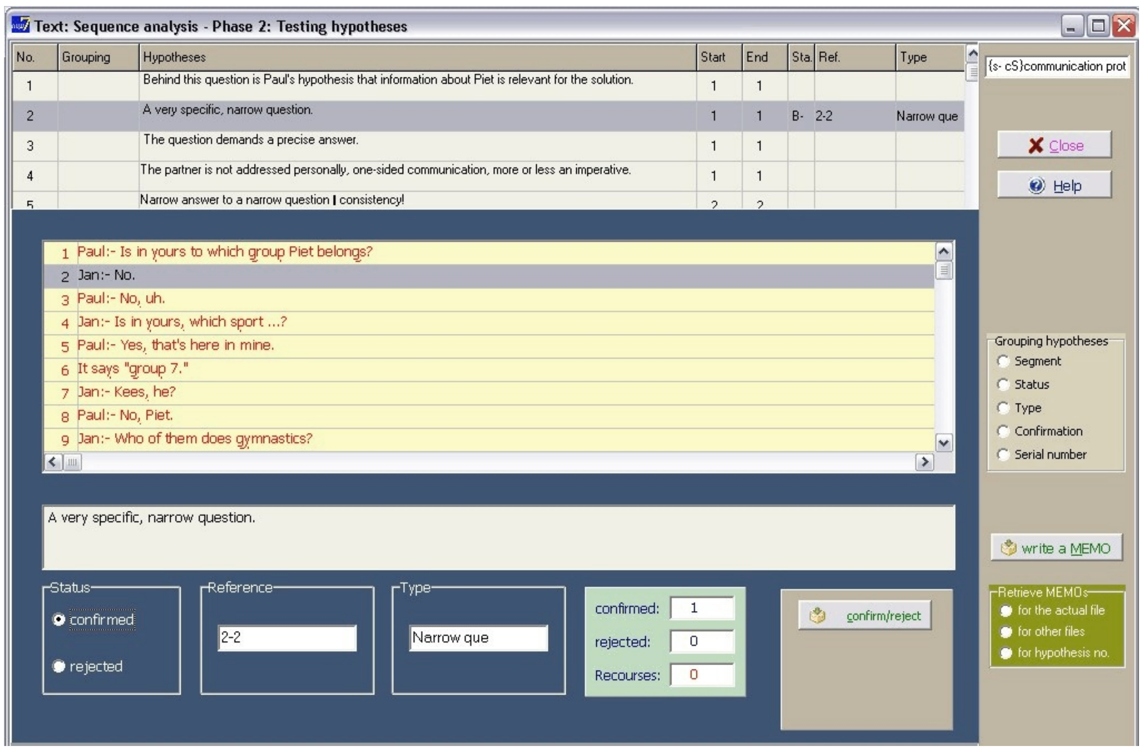
and the type name is pasted into the window.

Once you are convinced that the hypotheses show a pattern of repetition, you may stop to "tell stories" about the next segment. Instead, you continue reading the segments until new aspects appear in the text, which are relevant for your research question. From this point on you have to work again in strictly sequential order! Finally, you may proceed to the second phase, testing your hypotheses.

5.4.2 Testing hypotheses

In the main menu we click in"*Sequence analysis*" on the option "*Phase 2: Testing hypotheses*" and then on "*carry out*". Again we select the actual file (in our example only one, "*{s_cS}communication protocol.atx*") of our project and start our work. Testing a hypothesis is done in three steps:

(1) In the hypothesis table we click on the critical hypothesis (third column),
(2) then we look in the complete text field below for confirmation or contradiction and mark the corresponding line(s) with the mouse. Automatically the line numbers auf the selected text segment(s) will shown in the column "Ref."
(3) Finally, we have to mark in the box "Status", whether we assume that the hypothesis is confirmed or rejected.


Text: Sequence analysis - Phase 2: Testing hypotheses

No.	Grouping	Hypotheses	Start	End	Sta.	Ref.	Type
1		Behind this question is Paul's hypothesis that information about Piet is relevant for the solution.	1	1			
2		A very specific, narrow question.	1	1	B-	2-2	Narrow que
3		The question demands a precise answer.	1	1			
4		The partner is not addressed personally, one-sided communication, more or less an imperative.	1	1			
5		Narrow answer to a narrow question I consistency!	2	2			

{s- cS}communication prot

1 Paul:- Is in yours to which group Piet belongs?
2 Jan:- No.
3 Paul:- No, uh.
4 Jan:- Is in yours, which sport ...?
5 Paul:- Yes, that's here in mine.
6 It says "group 7."
7 Jan:- Kees, he?
8 Paul:- No, Piet.
9 Jan:- Who of them does gymnastics?

A very specific, narrow question.

Status: [✓ confirmed] rejected

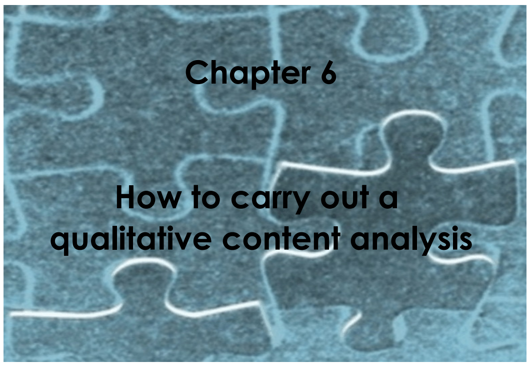
Reference: 2-2

Type: Narrow que

confirmed: 1
rejected: 0
Recourses: 0

The gray box on the right side allows also to sort the hypotheses according to their "status" (confirmed or rejected) in a table; sorting with "Serial number" restores the initial state. If you want to receive an overview on text segments and attached hypotheses in the same presentation – not in two separate tables as during generating hypotheses – you just click into one of the columns 3 - 6 (Start, End, Sta., Ref.) and you see a list, in which the text segments are followed directly by the hypotheses assigned to them. This list may be saved or printed for further use.

Now, what is the result of sequentially analyzing this "*communication protocol*"? We suggest you go through testing the hypotheses yourself with the available material or you erase in sub-directory ..\cod the files "*{s- cS}communication protocol.shg*" und "*{s- cS}communication protocol.shc*". Alternatively you may move these files into another directory. Then you can gain for yourself experiences with a sequence analysis – working from the very beginning with your own hypotheses. Anyhow, the resulting case structure will demonstrate that communication fails, if the partners do not share their resources, but one partner follows his/her own ideas and tries to treat the other one simply as a supplier of information.



Chapter 6

How to carry out a qualitative content analysis

In this chapter we try to outline typical phases in the process of a qualitative text analysis. Particular analytical steps are marked as characteristics of these phases, although concrete processes of qualitative analysis usually follow a cyclic path, i.e., during every phase a researcher may be engaged part of the time in activities characterizing other phases of analysis. In addition, this chapter tries to give an overview on the usage of computer assistance for qualitative analysis as provided by AQUAD. Details are explained in later chapters. However, the following chapters of this manual cannot offer more than a general introduction to methodological approaches to qualitative analyses and possible contributions of computers. We suggest that users who want to learn more about underlying principles read the following books:

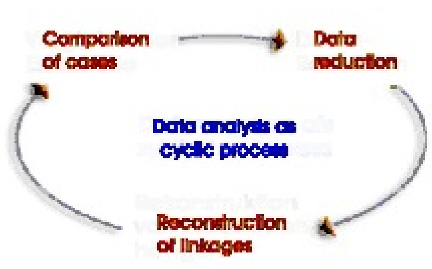
- General contributions of computers and software tools to qualitative research are presented in Tesch (1990), Kelle (1995), Fielding and Lee (1998), and Lissmann (2001).
- Miles and Huberman (2nd edition, 1994) give a very detailed introduction to interpretational analysis of qualitative data, based on numerous examples. Concrete examples from a variety of qualitative studies in educational research can be found in Bos and Tarnai (1998) or Schratz (1993) or Schratz (1993), from the field of psychological studies in Kiegelmann (2001, 2002, 2003).
- An introduction to techniques of theory-building in qualitative analysis, based on the approach of "grounded theory", is offered by Strauss and Corbin (1990).

This list is far from being exhaustive, but names some selected books which seem appropriate for beginners in qualitative analysis and which demonstrated their usefulness in Huber's seminars at the University of Tübingen. All of these books also offer excellent approach to more specialized literature. In the following, this chapter borrows from an introduction by Huber (1992) which unfortunately is not available in English.

6.1 Which are the steps in a qualitative content analysis?

As typical phases of qualitative analyses of data we can distinguish the reduction of the original data base, the reconstruction of linkages, and the comparison of findings. Especially in psychological studies researchers often are interested in inferring inductively from *one* subject's data regularities in this person's experiences and behaviors. Whether commonalities can be found when taking into account data of several persons is interesting, too, but only during a later stage of investigation.

- The first phase of qualitative analysis is characterized by *reducing* the overwhelming amount of data (texts, sound or video recordings, graphic files) by identifying the content of more or less encompassing data segments. A "code" as abbreviation or name is attached to each segment. In the following, these codes are used as representatives of data segments or "units of meaning" in the data. Fundamentally, this is a process of categorization, where the categories may emerge during data interpretation or may be taken from an already existing category system – depending on the researcher's epistemological orientation.



- During the second phase researchers try to *reconstruct* the data producer's subjective meaning system from the units of meaning in their data. By "data producer" we refer to the researcher's interview partners, to writers of diaries, to observers who took field notes in a setting, the child playing with an object in a video, etc. In order to reconstruct meaning systems we are looking for regular linkages between units of meaning in the data, which are characteristic for a person and/or her situation.

- In the third phase finally researchers try to infer invariants or general commonalities by *comparing* individual systems of meaning or "cases" (see Ragin, 1987).

It is important to keep in mind that these phases neither are strictly demarcated nor do they follow each other in a linear sequence, but they overlap and are linked to each other in circular patterns (cf. Shelly & Sibert, 1992). During data reduction we may start to ponder about a person's implicit theory or we may permanently compare the data at hand with other data which we have analyzed earlier. Thus we may perhaps detect in person C's data an aspect of meaning which we overlooked in person A's data. As a consequence, we repeat the process of data reduction for person A. In all of these phases it is necessary to affirm deductively the validity of our generalizations. That is, we try to infer particularities from our general findings and then return to our data and try to find evidence in form of specific information, i.e., statements in the texts or recordings, sequences of action in the video, etc.

6.2 How to reduce qualitative data?

The principles of reduction are obviously simple, but their application soon proves to demand very much work, to consume very much time, and to be very prone to errors. Voluminous verbal materials have to be reduced to the units or categories of meanings they contain. Tesch (1992) describes these principles as retrieving and marking of text segments, which are relevant for the question under study, by an abbreviation, that is a code for the particular category of meaning. Tesch also compared computer assisted analysis and "traditional" approaches, which use segments of texts or text clippings in the literal sense or which transfer relevant text segments to index cards. The work load is tremendous in both cases. Computer software not only assists in reducing the amount of data, but also in reducing all the mechanical labor otherwise necessary.

Instead of handling verbose clippings of text distributed over many piles of index cards, further computer assisted analyses use just the codes of these text segments; that is, after data reduction you work with category names and information where to find the categorized text segments in your texts. If you need to scrutinize the original text segment again during the course of your work, the computer will retrieve it immediately for you.

If the original data, however, are not available in form of text files, but come as hand-written diaries, sound- or video-recordings of interviews, graphic notes or drawings, etc. we had to transcribe these data until recently, that is convert them into text files. Particularly in these cases AQUAD 7 helps to save a lot of time and money, because it allows to reduce directly all these types of data to codes – without a detour via transcriptions. Of course, you may still transcribe critical parts of your data files for (verbal) publication in later reports, articles, etc. There is a small text processor within the "Coding" components of AQUAD , and the "Memo" function (see chap. 8) also offers the possibility of transcribing multimedia data.

In case of any type of original data, the critical question in this phase of analysis is: *How and where do I find units of meaning in my data?* Beginners in qualitative analysis as well as experts who try to get familiar with a new content domain ask this question again and again. Weber (1985) describes six widely used general possibilities to define text segments, namely to choose as unit of analysis single words, meanings of words, sentences, topics, paragraphs, and the complete text (for instance, if the texts are short as in the case of letters to the editor or if you want to produce head lines or abstracts). However, this choice cannot be made mechanically, but it needs itself tentative qualitative decisions. Not so obvious when single words are used as units of analysis, but quite obvious when we use more complex alternatives like word meanings, sentences, typical sequences in a video, etc. we need preceding insights or hypotheses that the unit we have chosen will contribute to answer our research question. Additional qualitative decisions are necessary, for instance which words are used synonymously or which idiomatic expressions have similar meanings for the writers/speakers in case of verbal data.

The strategies of defining units of meaning describe an important difference between quantitative and qualitative approaches to text analysis since the beginnings of a broader reception of text analysis in the social sciences. In the same year, 1952, two trendsetting articles were published by Berelson and by Kracauer. While text analysis serves for Berelson to assess systematically, objectively, and quantitatively the *manifest* contents of communication, qualitative content analysis according to Kracauer tries to reveal the categories of meaning hidden or *latent* in the data. Both authors relate their controversial positions to a debate, which Thomas & Znanecki (1918) had initiated 35 years earlier with a meanwhile classical analysis of letters of Polish immigrants to the United States. Obviously the question of adequate units of analysis – here words vs. meanings – is confounded with the particular goal of text analysis.

On the level of processing both approaches do not necessarily exclude each other. Especially with assistance of computers we can apply various strategies from the quantitative approach to qualitative text analysis in order to gain support for our interpretational endeavors.

Principally we should keep in mind to develop the analytic units *during* data interpretation. This is true in analyses which try to *understand* the experiences and actions of people from their own verbal descriptions as well as in analyses which try to *explain* specific actions by relating them to the frame of reference of these people's implicit theories. Only this approach helps to open a door to the subjective world views of our interviewees or producers of other types of data. Otherwise, we would be in danger of grasping only some partial aspects of their world views, maybe isolated from their subjective context, which our personal analytic grid is able to comb out. The strategy of developing categories "on the fly" corresponds to the approach of "grounded theory", an empirically based procedure of generating theories recommended by Glaser & Strauss (1979).

However, this procedure demands enormous work as soon as more than two or three data files are to be analyzed. Because we usually want to compare the results from single cases in advanced stages of analysis, we have to ensure that we defined and coded the units of meaning in all of the cases coherently. Usually this process has to be repeated again and again, and in this process units of meaning and their codes have to be modified. Miles & Huberman (1984; 1994) suggested a compromise, which structures the process of data reduction from the beginning: Before you start analyzing your data files you state a very general *frame of orientation* without any references to particular contents; then you try to find specific units of meaning within this framework. This structuring does not contradict the demand of openness for emerging categories while approaching your data directly, because usually the reduction of potential data by structuring starts even before you enter the phase of coding, for instance when planning and deciding which persons, cases, sites, types of data, etc. should be included in a study.

6.3 General structure of the process of analysis

(1) In the beginning it is indispensable to become first of all acquainted with the subjects' perspective of the problem under study – above all, if we were not present personally in the phase of data collection, that is, if we did not talk as interviewers with our research partners or did not run the video observation, etc. Or expressed the other way round: We should not try to establish a differentiated system of categories already while reading, watching, listening to the first data set. For sure, this system would be valid only for the first case, and we would have to revise it fundamentally as soon as we get a closer view on the second case.

Supposing you work in your project with a larger number of data sets, we recommend instead that you select a few cases to begin with either by chance or according to the principle of "theoretical sampling" in single case studies (cf. Yin, 1992) from all of your data files. For instance, if we were going to analyze the teaching of mathematics in different schools, we would have to consider possible differences depending on age of the students, school type, and teachers' professional experience. According to these criteria we would select a few data sets, for instance, video recordings of math lessons, and elaborate a provisional frame of orientation. Maybe we video-taped a priori in classrooms, which were known for different approaches to teaching and learning mathematics. Of course, we would then base our first orientation (also) on this criterion.

Obviously this first step modifies a recommendation by Miles and Huberman (1994; see above) to decide about a general, not content-specific frame of orientation already before we start to read the data texts or watch/listen to any recordings. Instead, we suggest to draw a sample of data sets/cases to test and differentiate any frame of orientation, which may already be available, at least in outline, just because we have a research question.

(2) If we decide to select a sample of data sets according to the principle of "theoretical sampling", we are forced to give the matter of general characteristics of our cases considerable thought. In other words, we have to ruminate on possible determinants of the "profile" of various cases. In the example of classroom observations we would think about the age of students, school level, professional experience of teachers, number of students, didactical orientations of teachers, etc. It would be a good idea to note these characteristics in a memo (see chap. 10) and to apply them later as "profile codes" (see chap. 7.1.1) or "singular codes" (which are attached only once) to characterize the data set in general. If we had selected some data sets, however, by chance we could now consider in advance, which characteristics of persons and/or situations involved in our study may become relevant as "profile features" later during more detailed analyses.

(3) We have a look at the selected data sets without getting involved in sophisticated codifications, but try instead to understand the content as a whole, differentiate essential parts, find out about what the cases have in common and what makes them different. Thus we mark out gradually a frame of orientation. A decisive aspect during this step is to note all preliminary interpretations and ideas in memos (see chap. 10), including at the end a short (!) thematic summary, in which we note our "first impression" of the general meaning of each data set, and what we conceive of as its central aspect at this early point in the analytic process.

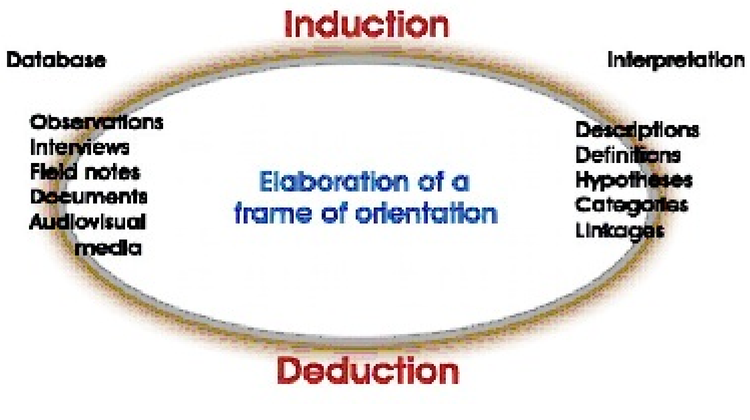
(4) What follows then is a critical decision about the general strategy of interpretation we want to follow: Should we try – as already initiated in step 3 by writing a thematic summary – to find more general units of meaning, mark them by codes, and differentiate these units stepwise and in repeated sessions? Or should we just in the beginning pay attention to *all* the details, maybe somewhat restricted by our general frame of reference? In this case we would try in further sessions to summarize many of these detailed codes in more abstract categories, thus also making the cases better comparable. The first strategy moves from general to particular considerations ("differentiation"), while the second strategy starts with particular aspects and leads to general insights ("generalization").

Which strategy we should prefer depends, of course, on the research question, but also on the researcher's experience with his/her topic. Principally, we want to advise beginners *against* the strategy of generalization, which urges them to concentrate on tiny specifics in a text, picture, or a recording and broaden their perspective only later, when they try to develop more abstract, general categories. Thus beginners prevent mostly successfully any deeper, at the beginning uncertain and difficult theory-based analytic considerations, but run the risk of indulging in superficial hustle and bustle. Novices in coding use to be content in this situation, because they feel busy, they produce codes – and notice only much later that their activities were not necessarily goal-directed. In one of my seminars, for instance, a group of students generated codes for – in their opinion – keywords in a text, basically doubling the data instead of reducing them. In another case, a novice of qualitative content analysis developed about 1500 (no typing error!) codes for his interviews until he realized that he was unable to see the wood behind all of these trees.

Therefore we judge it to be most promising, if above all novices take the principle of "permanent comparisons" very seriously, which is basic within the approach of grounded theory (Glaser & Strauss, 1979). Permanent comparison demands to look within and later across all data sets immediately after introducing a category for congruent and incongruent examples. This orientation prevents to a large extent that the qualitative analyst gets lost in an overwhelming number of interesting details, but tries to elaborate insight into the structure of meaning in his/her data. The following table contrasts both strategies:

Strategy:	Differentiation	Generalization
Coding starts with	Search for general categories	Search for specific aspects
Further steps	Differentiation with the aim of uncovering specific differences	Generalization with the aim of finding common grounds

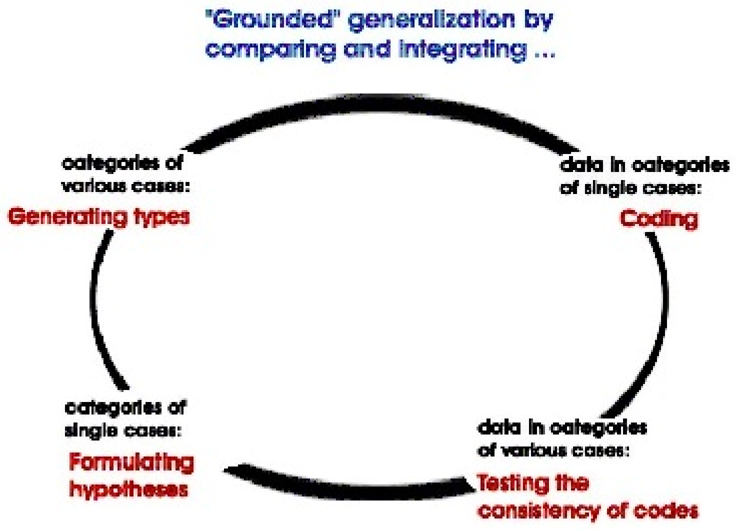
Both strategies are not mutually exclusive, but are on the contrary mutually dependent. The interpretation of qualitative data is not a linear, but a cyclic process:



Based on ideas of Dewey, Shelly and Sibert (1992) described the relation of inductive and deductive thinking in qualitative analysis as illustrated in the figure above. From this point of view we should object to the recommendation for novices (see above: start data interpretation from general categories) that there is only a chance to find something new within the data, if we interpret small data segments. In text files this means to code the data more or less line by line. For sure, the higher probability to introduce new ideas by inductive reasoning is counterbalanced by greater danger to miss a general and essential aspect.

If you follow our recommendation to approach your data with a strategy of gradual differentiation, the accent lies on the side of "Interpretation" in the figure above. Of course, interpretations and interpretive knowledge may have developed already inductively during concrete experiences in the field or sifting through the data files, for instance in form of some very general categories (example from a study in a kindergarten: "The nursery school teacher directs the kids' attention;" "She supports them in case of difficulties;" "She makes them verbalize their experiences;" etc.) or working hypotheses, which already influenced the formulation of research questions. Necessarily the database will be differentiated as soon as we analyze the data from different persons who were observed or who talked to us in different situations. That is, if we apply the strategy of generalization we begin to analyze our data with specific meanings, which emerge from within the data. What we need then are signs that show the direction from specific interpretations to general interpretive knowledge. "Permanent comparison," the basic principle of grounded theory was already characterized as guideline for generalizing. These comparisons follow a cyclic pattern, too, as Shelly and Sibert (1992) have underlined.

Following their scheme of procedures, we can describe permanent comparisons and their goals during the progress of qualitative analysis as depicted in the figure on the next page:


"Grounded" generalization by comparing and integrating ...
data in categories of single cases:
Coding
data in categories of various cases:
Testing the consistency of codes
categories of single cases:
Formulating hypotheses
categories of various cases:
Generating types

(5) After the principal strategic decision we start to code a sample of data files (see step 1). Necessarily we will again and again switch from comparing and integrating single data within one file to comparing a category across several files – and back again. That is, we move forward in smaller, less encompassing circles within the general analytic cycle, first of all to test whether we apply our categories consistently.

(6) In this way we develop a set of coding rules, which we finally apply to all data files available in our project. Of course, we must not wonder, if we come upon exceptions and even contradictions, which make us modify the coding rules. In some cases it may even be necessary to leave the cycle of data reduction/-interpretation and enter once more the phase of data collection.

(7) During the whole process there will be many occasions and good reasons to write memos. When you come now to the point, where you try systematically to formulate hypotheses about your subjects' world views or to

generate types, you will be grateful to find your initial ideas and assumptions from earlier stages of analysis again. In this stage we concentrate our efforts on elucidating relations or "linkages" between critical categories (see chap. 8). This was described in step 4 as integration of categories in "hypotheses", for example: "If person P observes the events A or B, s/he reacts by doing X." Or: "If teacher T talks about her/his students' lack of motivation, s/he continues by complaining about the influence of TV and the role of parents."

Experiences in this phase, for instance detecting contradictory interpretations, may be a reason for an analytic loop back to the phase of coding and testing consistency (see step 4). Also some additional data collecting, maybe for some selected cases and/or with modified questions or observation methods may turn out to be helpful.

(8) Given a sufficient number of cases, we may finally try to group these cases into types, based on comparison and integration of (theoretically selected) categories. As result we achieve a differentiation of cases according to typical configurations of characteristics (see chap. 11).

(9) Finally, we must not fail to point out something that is relevant for the process of qualitative analysis as a whole: Qualitative analysis has something to do with the "quality" of events and states captured in our data. Therefore, our codes should not represent neutrally the fact of specific events or states, but have to inform also about their quality – or expressed bluntly: Were these events good or bad? For instance, we would not be able to formulate hypotheses or generate types in a biographic study on problems of youth, if we marked those interview segments, in which young people talked about their parents and added simply the code "parents." Can you imagine young people who do not talk about their parents in a biographic interview?

The big question is: *How* do they talk about their parents, family climate, social relations, etc.? A characteristic or a variable, which appears without any variation in all cases, does not contribute to "permanent comparisons" and is therefore not suitable for differentiations or explanations that would get us anywhere. Consequently we have to qualify statements like those about parents. In the beginning rough grades like positive/indifferent/negative (or abbreviated as appendix to a code: +/0/-) will be sufficient for a search of differences and common grounds.

From an interpersonal point of view it is a most pleasant trait of many novices in qualitative analysis to refrain from jumping to evaluations and qualifications particularly of personal issues. On the other hand, avoiding what is meant to be rash attributions of quality ("positive", "negative") misses the purpose of qualitative analysis! At least during a test of consistency of coding all "neutral", non-qualifying codes should be checked, whether the attributed data segments need a repetition of earlier coding steps with the goal to make sure that these codes depict differences – if there are any – in the meanings of the data segments.

6.4 How to find units of meaning and codes?

When looking for units of meaning in the data files, you should be open for emerging structures in your data. However, recipes enforcing structures on the data may often be welcome because they help to shorten a phase of maximal uncertainty particularly during the first steps of data analysis. On the other hand, the price for certainty and savings of time and effort may be too high: concrete guidelines would lead only to those units of meaning which could be foreseen, while surprising, rare, but maybe most interesting aspects are in jeopardy of being omitted from further analysis. The following hints should be understood as heuristics, that is, as general directions which provide assistance in identifying units of meaning and attaching appropriate codes, but not as algorithmic rules which could be followed step by step to a predetermined goal. Three heuristics will be described in the following. AQUAD offers support particularly for the first two of them and their variations, which are also outlined.

- When looking for *categories* we read a text, listen to a recording or watch a video and try to be sensitive to emerging concepts, to statements about situations, events or persons, to opinions, ideas, etc., which can be attributed to a general, super-ordinate category.

- When looking for *sequences* we pay attention to statements of linkages, connections, relations, etc., expressed in the data, and we try to put these subjectively linked statements together in a unit of meaning – which will be more comprehensive then the results of categorical data reduction.
- When looking for *themes* or topics we have to go to the most abstract level; in some cases a complete data set is reduced to its topic, for instance, when we reduce short texts like "letters to the editor" etc.

6.4.1 How to find categorical codes

In addition to looking for units of meaning in the data, we have to decide, whether our categories and the corresponding codes should describe, interpret or explain the content of a particular data segment (see Miles & Huberman, 1994, p. 56). This is a decision for which we cannot expect computer assistance. Here, the computer assists "only" in documenting, in sorting, and in revising – if necessary – all the decisions a researcher has made on his/her own responsibility. Then, three possibilities are widely used to find categorical codes:

Applying pre-determined category systems

If a qualitative study does *not* aim at constructing theories from concepts emerging in the data, there is a very simple means of finding codes: You can use an available category system and reduce your data according to the interpretational schemata contained in the system chosen. Categorical systems may be "available" from earlier studies on the same topic or from publications of other authors.

When using a pre-determined category system we have to decide which segments of a given data set correspond to which of the given category definitions. Within AQUAD we have then to note where each of these segments is located in our data and which codes apply to them. How this is done will be explained in detail in chapters 7 and 8. Without fail we will have trouble assigning specific data segments to one of the given categories, and we will not always use the codes consistently. In these cases, the function to *retrieve coded data segments* is very helpful to control our work. After entering a critical code (or a whole list of these codes), the program shows us very quickly all data segments in all data sets analyzed, which up to now were marked with this code.

However, this approach to test the reliability of our coding does lead us only to those data segments which were assigned erroneously to a particular category, but not those segments which belong to the definition of this category but were *not* assigned to this category. In conventional approaches to data analysis we would detect this type of error when we notice inconsistencies within the sets of segments assigned to other categories. With computer assistance it is no problem to control for "missing" data segments by checking all segments which were assigned to related and thus "error prone" categories. In addition, we can make use of the complementary relations of manifest and latent units of meaning in our data – however, only if these data are texts (!). Three strategies appear to be useful:

- We define *key words* as manifest indicators of a critical meaning and have the computer retrieve all their occurrences in the texts. If we find our key words in a specific text segment, we can decide whether this segment should not better be assigned to the category indicated by this key word. For this strategy most text processors can be used.
- We assemble key words in a *dictionary for analysis*, called a *word catalog* in AQUAD, that is we use a list of key words for the retrieval of critical text segments. Then we find the information needed in a single run of the program.
- AQUAD offers both of these possibilities, automatically combined with a third strategy called *keywords-in-context* (Popko 1980). This function tries to retrieve one or more key words in our texts and prints it within the context of the line where it was found in a text. The researcher is responsible then for further decisions, for instance about the range of a corresponding unit of meaning and about marking it with a particular code.

Hypothesis-based categorization

The research question often supplies the researcher with hypotheses that may be used as a sort of guidance when looking for units of meaning in the data. Based on these hypotheses the researcher tries to define categories and rules of coding for her/his data.

> For instance, the approach of Marcelo (see p. 18) to code his interviews with beginning teachers was hypothesis based. A theoretical model of professional socialization served as a framework to design a preliminary category system. With increasing familiarity with the texts and deepened insights into the subjective points of view of the beginning teachers, some of the categories had to be omitted from the system as inadequate, while some new categories emerged from the analytic process. Some "narrow" categories could be combined to more comprehensive units in terms of the teachers' subjective theories (see Huber & Marcelo, 1992). Of course, new or combined categories had to be compared to critical text segments in all interviews again.

Properly, the researcher decides about concrete categories already when designing the collection of data. Thus, for instance, the formulation of guiding questions for an interview would be deductively determined by the researcher's basic hypotheses. On the other hand, the researcher will for sure experience interview situations or find data segments in the interview transcriptions or recordings which cannot be assigned to predetermined categories. For these data segments categories have to be developed inductively. The new categories may relate the data to the original hypotheses – but they may also be a stepping stone from which modifications of the original frame of reference can be realized. We see that the researcher's degrees of freedom as well as challenges to her/his interpretational aptitudes are increasing when applying this second strategy. Most important, the decreasing structuredness of this approach augments the chances to take into account the subjects' points of view.

When applying this strategy, the researcher has also to meet the demands of the "method of permanent comparison", which is a trade mark of the approach of grounded theory (Glaser & Strauss, 1967, 1979; Strauss & Corbin, 1990). Shelly & Sibert (1992) described this method in detail and related it to models of the researcher's cognitive activities. An activity of major importance in "permanent comparison" is to test every inductive conclusion from particular data to more general principles – here the analytical categories – by means of deductive conclusions, that is deductions of specific units of meaning in the data base.

Together with the demand to apply interpretation in data analysis the importance of computer assistance is increasing. When we try to get an overview on linkages between categories, simple retrieval functions reach their limits. From this point in data analysis on, AQUAD serves you especially well, because it supplies you with routines for deductive conclusions based on the principles of logic programming (Tesch 1990; Shelly & Sibert 1992).

- The functions for *retrieval of coded data segments* and for retrieval of *key words* (including retrieval dictionaries and the KWIC-function) may be used here to check the consistency of coding within and across data sets in the same way as in studies applying ready-made category systems (see above).
- From the hypotheses, which mark out a framework for the development of categories, we get hints to specific *relations of categories*. AQUAD supports hypothesis-based data reduction with functions for testing relations of categories. Among other relations you can find out about the *super-ordination/subordination* of categories, *sequences* of categories or *clusters* of particular categories. These three types of relations probably represent the most frequently tested relational patterns tested in data analyses. The test activities demand that you focus your attention not only on one category and the data segments it represents, but on two or more categories and the defined relations between them -- which may include negations! Of course, non-events or missing relations have to be registered in this process, too.

Categories by theory-building

The most exacting mode of reducing qualitative data refrains from any prescriptions for data reduction as for instance category systems and from structuring the process of qualitative analysis by hypothetical frameworks. Consequently the researcher has to keep in mind his/her subjective experiences, opinions or prejudices as regards the world views or behavior of his/her subjects, and the researcher has to avoid premature stabilization of emerging principles of data reduction by permanently comparing the momentary categories and relevant statements in all available data files. To be sensible of one's own way of reading a text or interpreting a video is particularly important if data are analyzed which were written or recorded long ago or in a context that differs markedly from the researcher's way of life (Fischer, 1982). Often specific information about the speaker or writer of a text is very helpful. If this information is not available in a text, it may be necessary to look for additional sources. These sources and their information help to approach better the goal of viewing the world of the subjects of a study through their own eyes and to understand it from their own perspectives (or, for more auditive readers: these sources may support listening to the speakers' own voices in their texts).

In this process we try not only to describe subjective world views, but to order them by matching concepts and to reconstruct systematic relations between these concepts, that is, we are occupied with what Glaser & Strauss (1967; 1979) called the *discovery* of a "grounded theory". The term "discovery" accentuates an essential difference from methodological approaches which are applied to confirm given theories. Grounded theories are developed during content analysis, therefore we do not start from an available theory looking for verifying or falsifying data in the data, but we start from a phenomenon the data refer to, which we want to understand and to explain (Strauss & Corbin, 1990).

This approach demands a maximum of interpretational efforts. Exactly this aspect seems to cause problems for beginners of text analysis. Pre-defined category systems come with explicit interpretation rules or they transport these rules implicitly in a set of examples for each category. Researchers using a hypothesis based approach to categorization can at least rely on a general orientation what to look for in the data. In the process of theory construction, however, researchers have to find out first what a data file is telling them.

Beginners often tend to avoid the risk of errors and to interpret as parsimoniously as possible. In extreme cases this tendency results, for instance, in reading a text for the appearances of critical formulations – comparable to the application of key words when using ready-made category systems. If text segments containing such a critical formulation are marked by a specific code, this code does not necessarily represent a "unit of meaning" or signify a subjective viewpoint of the writer/speaker, but more probably the mere fact, that a particular formulation was used in this text segment. Its meaning still may be unclear.

This tendency is especially pronounced, if a researcher does not know about the possibilities of tentative interpretation and easy revisions in computer-assisted qualitative analysis. Supported by AQUAD a researcher is not punished for "playing" with ideas by tremendous labor when it comes to revisions of codes during a later phase of analysis, for instance, when the researcher finds data segments not matching a category introduced earlier. On the contrary, computer assistance encourages creative interpretation, because we have to introduce codes as markers of data segments from the very beginning of an analysis, and changes, summaries, differentiations, etc., of categories can be realized smoothly (Tesch, 1992).

As a *rule of thumb* for theory construction by categorization we can state: You should look in the data for units of meaning *as large as possible* in order to find something to interpret at all; at the same time these units should be *as small as necessary* to avoid representing incongruent contents by the same code.

This approach could be called a *strategy of differentiation*. AQUAD supports this strategy, because units of meaning can be defined without limitations; above all, units of meaning may overlap, and the same data segment may be assigned to several categories, that is, it may be marked by several codes. If you are already familiar both with qualitative analysis and with the domain addressed in the datas of your study, you may reverse this method and approach your data following a *strategy of generalization*.

These prerequisites given, firstly the probability is low that you fall a victim of details in the data and miss to reveal its essential meanings. Secondly you are then also accustomed to software functions for producing meta-codes, which assemble a number of too detailed categories into one more comprehensive category. In a generalizing approach it is particularly important to compare coded data segments permanently in order to find out inductively about all content dimensions expressed in these segments and the adequate super-ordinate categories. AQUAD supports this strategy, too. You can retrieve easily those data segments which are similar to a particularly coded one, which contradict its meaning, which depend on it or are related in other, specified ways to this data segment. Based on findings about such relations you can then try to combine individual codes into a more comprehensive one.

When we reduce data to categorical codes, we try to distinguish data segments in such a way that we can assign them to well defined, mutually exclusive categories. Unequivocal assignment of meanings to categories does not imply, however, that the text segments involved always have to be clearly different from each other. Depending on writing styles or communicative styles of the data producers or depending on the researchers' ways of interpreting their data, segments assigned to differing categories may overlap or the same data segment may even become assigned to more than one category.

6.4.2 How to find sequential codes

The starting point for sequential coding is the detection of specific relations between data segments. If specific linkages of segments emerge from the data, a researcher may want to mark their appearance in a data file and to represent the type of linkage by a particular code – in the same way as the researcher used categorical code. Here, however, the code represents a defined sequence of meanings found in a data file. The unit of meaning is the complete data section, in which the sequence of meanings was found.

Which strategies are at hand that may help us to go beyond assigning categories or subcategories to data segments and to detect sequences or linkages of meaning in a data file? In the following we differentiate between strategies that inquire into simple sequences and complex sequences of meaning.

Looking for simple sequences

Besides looking for *hierarchical sequences* of super- or subordinate categories as recommended by Strauss & Corbin (1990) for textual data, a number of other simple sequences appear to be interesting from the point of view of grammatical or linguistic properties of the text. When using these strategies, we should be aware, however, that we may be about to abandon looking for emerging categories and begin to enforce categories on the text (see Glaser, 1992). Whether this switch is permissible and which of the possible strategies is adequate cannot be answered absolutely, but depends on the research question. Often researchers are looking for *causal sequences* (using key words like "because", "based upon", etc.), *temporal sequences* ("while", "then", "before", etc.), *concessive sequences* (positive concessions like "not ... but ..." or negative concessions like "indeed ... otherwise ..."), *conditional sequences* ("if ... then..."), *final sequences* ("so that", "lest", "in order to", etc.), *comparative sequences*, *modal sequences*, and *defining sequences*.

Neither this list of types of sequences nor the quoted key words claim to be complete. They only want to stimulate your own trials to retrieve sequences of meaning in the texts according to the researcher's questions. In this context we should again accentuate the function of computers and software as useful *tools* for text interpretation, but not as agents of qualitative analysis. Functions for word retrieval may be especially helpful, but within very narrow limits. Except in cases where a researcher analyses carefully formulated texts, the yield of dictionaries of particular sequences of meaning is usually limited. In freely spoken interview texts even sentence structures are often hard to define: Where is the principal clause, where does a subordinate clause begin? Often conjunctions critical for defining a specific type of sequence may have been in the mind of a speaker, but they do not appear in the spoken and transcribed text. In addition, many speakers do not use critical conjunctions or constructions according to the grammatical rules. Therefore, looking for key words or applying whole dictionaries of these words never substitutes for empathetic interpretation of texts – but these strategies may be useful heuristics.

Looking for complex patterns of sequences

Structural or content characteristics of data may inspire the search for complex sequences of meaning. For example, when a researcher tries to reconstruct theories of action implicit in a text, audio- or videotape he or she could look for sequences of appraisals of situations, reflections of alternatives for action, expectations of action effects, and evaluations of potential personal consequences like satisfaction or disappointment. This means, the researcher would have to keep in mind a number of categories and to look for them and their proper sequence simultaneously when interpreting a data set. Such an analysis is both demanding and prone to errors. The variety of structures of possible linkages of categories is demanding for software, too. It would not make much sense to implement all sequential combinations of possibly linked categories in form of abstract deductive algorithms into a program, so that the researcher would have to enter during run-time only some concrete codes as variables. For the retrieval of complex patterns a researcher needs access to the source code of her/his software in order to add those rules with minimal programming effort, which s/he expects to apply to the linkage of categories in the data. AQUAD offers exactly this possibility.

6.4.3 How to find thematic codes

The most radical approach to data reduction is applied, if we try to code a data file just by one central category, that is, its message or its topic. Since qualitative analysis is a cyclic process, we cannot locate the strategy of thematic coding at a particular place within the interpretational process, let's say at the end as a sort of summary of findings. Thematic coding may play an important role at this stage of content analysis, but this strategy is also useful at other stages:

- When we analyze relatively short and homogenous data sets (for instance, "letters to the editor") or paragraphs in a text, thematic coding may be all we need or want to reach our interpretative goals.
- In the case of large, heterogeneous, and complicated data sets interpretation may start with thematic reduction. Here, this strategy may serve us as an important heuristic. It may help us to find one or some main ideas in the data and not get entangled in a great number of potentially controversial details.
- Usually, thematic coding appears in the final stage of data analysis, often only after several cycles of data reduction, reconstruction and comparison of meaning structures during which the "Leitmotiv" of different data sets was elaborated more and more clearly (see Shelly & Sibert 1992; Strauss & Corbin 1990).

Strauss & Corbin (1990) recommend five steps for thematic reduction of data (in their case: texts), which are compatible with each of the three locations of thematic reduction:

(1) We start with identifying a main idea or a central category;
(2) then we look for subordinate categories,
(3) which often need to be differentiated or to be linked with each other;
(4) then we examine hypothetical relations among sub-categories as well as between these categories and the main idea by comparing data segments;
(5) discrepancies, for instance, inconsistent categories stimulate further cycles of analysis.

6.5 How to reconstruct systems of meaning?

In theory-constructing qualitative analysis we try to generate the speaker's or writer's theory of the issue her/his data file is talking about, that is, we try to reconstruct the speaker's or writer's subjective system of meaning. Solutions can be developed inductively, deductively or by combining inductive and deductive strategies. Which approach is to be preferred depends on the research question.

If we analyze transcriptions of minimally structured interviews or texts, for instance entries in a diary, we usually start by identifying units of meaning and assigning them to specific categories. Then we may look for typical sequences of categories. Finally we will try to subsume such sequences under more abstract categories, that is the message or the topics of the data producer. This analysis starts with inductive processes, but alternates between inductive and deductive conclusions at least from the stage of thematic reduction on (cf. chap. 6.3).

If we approach our data under a particular theoretical orientation or from the point of view of particular interests, we will try to retrieve specified relations from the beginning. That is, we start with hypotheses of possible relations and try to confirm them deductively in the data. Inconsistencies or contradictions then will cause processes of inductive analysis, which in turn may modify our hypothetical orientation.

In inductive approaches we try to generalize categories and their systematic linkages from concrete data segments. In deductive approaches we try to find concrete data segments which may confirm general assumptions or hypotheses about how specific categories are linked. The following overview on strategies offered in AQUAD for computer-assisted reconstruction of systematic linkages is structured according to these two approaches.

6.5.1 Reconstruction from the context of particular data

We are looking for systematic occurrences of a particular category (or several particular categories) in a data set which is characterized by one or more profile codes, usually socio-demographic codes. For computer-assisted reconstruction we create code matrices (Miles & Huberman 1984) or code tables (Shelly & Sibert 1992). The contents (that is, data segments) in the cells of such a table are determined twice, both by the category serving as column header (profile code) and by the category serving as row header. Thus, the interpretation of findings also is much more guided than in the case of just registering frequent closeness of otherwise "unconditioned" categories in space (line numbers, frame numbers) and time (sequence of data production). On the other hand, the construction of a table for analysis demands much more conceptual investments, that is greater progress in the process of data analysis. The module "Tables" (see chapter 8.2) is available for this type of reconstructions in AQUAD.

6.5.2 Reconstruction by testing particular relations

Here we can also distinguish two approaches, which correspond to the just described strategies of coding simple or complex sequences of meaning in the data files.

(1) *Confirming simple code sequences.* Let us assume that an interview with parents talking about their educational practices suggests that a father is trying hard to justify his ways of education. Then we could activate the module "Linkages" (see above and chapter 8.3) and examine our tests, looking for sequences of relevant codes, for instance for final or causal codes, in order to confirm our impression deductively.

(2) *Confirming complex relations of codes.* If a researcher wants to confirm relations of codes which exceed the complexity of those linkage structures already built into AQUAD, s/he can use an "design" module in AQUAD, where the researcher add her or his case-specific linkages of codes. How you put your assumptions about complex linkages of codes into a form which can be confirmed or rejected by AQUAD is described in detail with the help of several examples in chapter 11.

6.6 How to compare relations of meaning

Permanent comparisons of interpretations within a data set and across different data sets are at the core of all procedures of qualitative analysis (see Shelly & Sibert 1992). Even when reducing the data within a file to codes it appears to be impossible to attain reliable codings without permanently comparing codes/categories and data segments in the file at hand as well as in the other files of the research project. However, in most projects we want to achieve more than to get access to unique world views expressed in individual files by means of a category system valid for all data sets. At some point in the research process we usually want to establish general assumptions across files/cases. This task confronts the researcher with the proverbial danger not to notice the wood because of all the various trees. Tackling all the different formulations in the data texts and elaborating their precise meanings should not prevent us from looking for common elements. Therefore we have to notice systematic relations across files and to compare them, too. Comparing relations between social phenomena, however, regularly leads to findings of numerous conditions in manifold, sometimes controversial combinations. In addition, in many studies one's own findings have to be compared with findings from other studies, that is to perform a *meta-analysis* of qualitative findings. Even if other researchers have used the same research questions as we did or at least comparable questions, they will have found answers that may be only partially compatible with ours. When looking for configurations of conditions of a particular phenomenon, sciences dealing with the complexities of natural systems usually find highly varying constellations across different studies.

Let us take an example that is often discussed hotly in everyday life: "Is there a relation between cancer of the lungs and smoking?" Whoever wants to affirm this question is regularly confronted with the case of an 80 years old grandfather, who smoked all his life long, or with the opposite case of a victim of cancer that never touched a cigarette. Obviously a lot of conditions are to be considered. Empirical studies provide us with many empirical arguments and you can choose the ones you need in order to immunize your point of view against counter-arguments. Only a meta-analysis of all the relevant studies or at least a representative sample of these studies could give clarity about the relevant constellations of conditions.

By applying Boolean algebra to qualitative data Ragin (1987) has developed an important approach comparing qualitative data. The procedure is based on the Quine-McClusky-algorithm of "logical minimization".

According to Ragin (1987, p. 121) this approach fulfils the demands for a qualitative comparative research strategy, because

- a large number of cases can be compared;
- complex causal conjunctures can be addressed;
- "parsimonious" reconstructions or explanations can be produced, if desired;
- individual cases can be investigated both in relevant parts and as wholes (see also above: strategy of thematic reduction);
- competing reconstructions or explanations can be evaluated.

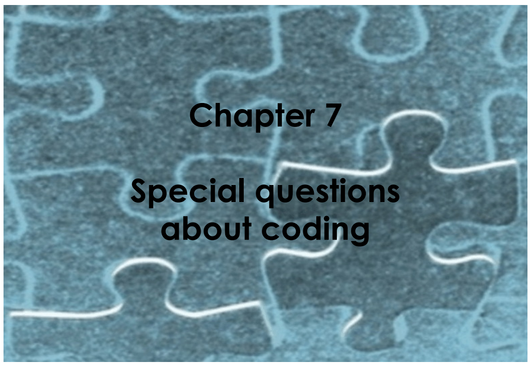
If we compare this strategy to variable-oriented approaches, which assume that variables can be combined additively, we find that Ragin's case-oriented comparisons (1987, p. 51 f.) appear to be able to

- uncover patterns of invariance or constant configurations of conditions by minute comparison of individual cases;
- react more sensitively to meaningful configurations of conditions than to relative frequencies of typical cases -- which implies that even a single contradictory case has to be attended to;
- consider cases as entities, that is to understand the conditions of a case in relation to each other instead of in relation to their distribution pattern in the population;
- provide a basis for examining how the conditions found combine in different ways and in different contexts to produce different results.

In order to apply the rules of Boolean algebra to qualitative comparisons we reduce in every single case all codes radically to "truth values.". We are satisfied with the binary statement "condition true" (i.e., given) respectively "condition false" (i.e., not given). The configuration of conditions of a particular case is represented by a row in the table of truth values. The conditions for a case are combined through Boolean multiplication (logical *and*). The different configurations, i.e., the rows in the resulting table are combined additively (logical *or*). In this way, we first represent the single cases as configurations of characteristics or conditions and then compare the patterns of configurations. It does not matter if we do not use exclusively qualitative data; scores on a quantitative dimension can also be reduced to truth values.

AQUAD offers the possibilities of logical minimization for qualitative analysis in its module "Implicants." Whenever comparative operations are necessary in the process of qualitative analysis, we should consider these possibilities. Because of the cyclic nature of qualitative analysis, logical minimization may be helpful during each of its phases.

As a *heuristic* this strategy is already helpful when we start to generate categories for text interpretation, even if we have only analyzed a few texts, that is even with a small basis for comparative operations. During the final phase of analysis, when we want to create a *summary of results* or when we want to group our findings, to distinguish types of writers or speakers, to determine key texts in our data base, etc., the strategy of logical minimization appears to be indispensable. As Ragin (1987, p. 51) has stated, "the potential volume of the analysis increases geometrically with the addition of a single case, and it increases exponentially with the addition of a single causal condition." AQUAD offers well elaborated procedures for grouping or clustering cases by means of logical minimalization of critical conditions. Finally, we can apply the strategy of logical minimization when we intend to *compare several qualitative studies*. This is a methodological step usually called *meta-analysis*. In the area of qualitative research, the strategy of logical minimization could supply the simplicity, transparency, reliability, and documentation desired for meta-analytical comparisons. You will find more details in chapter 11.



Chapter 7

Special questions about coding

7.1 How to code data?

7.1.1 Types of codes

It might be helpful for your work with AQUAD to distinguish clearly between six types of codes when working with AQUAD:

- *Conceptual codes* are used for "labels placed on discrete happenings, events, and other instances of phenomena" (Strauss & Corbin, 1990, p. 61), as for instance the concepts "mentoring" or "counseling". Wherever in a data set a researcher has the impression, for instance, that somebody receives an advice or is giving some advice, the researcher may mark this data segment by an adequate code label like "counseling" or "giving advice". Conceptual codes may have up to 60 characters (letters and/or numerals) in AQUAD. There is no need to use cryptic abbreviations as codes, because you have to type each code only once. Then it is listed alphabetically in a master code file, from where you can take it just by clicking.

- *Profile codes* or *socio-demographic codes* are used for characterizing the *profile* of an entire data file. They may be qualitative or numeric. A characteristic of a particular data file could be that it is an interview with a "female" person, or that the field notes come from a school with "more than 1000" pupils. Since these characteristics are mutually exclusive (the person interviewed cannot be female AND male, the school studied cannot have more AND less than 1000 pupils), they can appear in each data document only once. That is why we may call these codes also *singular codes*. Principally profile codes can be written in the same way as conceptual codes. Within AQUAD there is a rule to use the "/" as beginning character of profile codes to distinguish them from other types of codes. Thus, in our examples we would enter
  /gender: f
  /school size: < 1000

- *Numeric codes* are used where the phenomenon the researcher is interested in comes in quantitative values, such as test scores or time marks. They are especially useful for AQUAD's hypothesis-testing functions and for its table analyses. **We recommend using the "/" as beginning character also of numeric codes to distinguish them from other types of codes**. After the slash a word may follow to characterize the meaning of the following quantitative datum. If we want to include the age of our interview partners, we could use for instance the notation "/age:18" to include the information, that this person was 18 years old.

- *Control codes* are used only for internal purposes in AQUAD. They do not transport information about the data file and its producer. Therefore, they do not play a conceptual role in the process of interpretation. Usually, AQUAD uses character codes as internal switches for specific functions. At the moment only one control codes are defined – and it makes sense only if applied to text files:

  $do not count

  This control code excludes text segments from an analysis on the level of single words (counting, retrieving). More information about the use of this control code can be found in chapter 9.

- *Sequential* codes or *linkage codes* are functionally the same as conceptual codes, but are used as labels for systematic sequential co-occurrences of other codes, signifying complex clusters of meaning. Thus, they do not relate to *one* particular data segment only, but to a determined sequence of data segments. You may AQUAD have add them automatically to an existing code file while you run an analysis of linkages. The following example will demonstrate how sequential codes represent linked concepts:

  Imagine we notice in our data files of everyday conversations that people often get an advice (conceputal code: Advice), which is only sometimes gratefully accepted but in other cases rejected. Now we are able to differentiate between two typical conversation episodes: "Advice - acceptance" and "Advice - rejection." Exactly these codes could be added to our codification, marking to internally linked data segments, which start with the advice and end after an accepting or rejecting reaction (maybe we should include "Advice - no reaction" as third alternative, depending on our data).

  You will find more details about sequential codes and how to add them automatically in chapters 8.2 and 8.3.3.

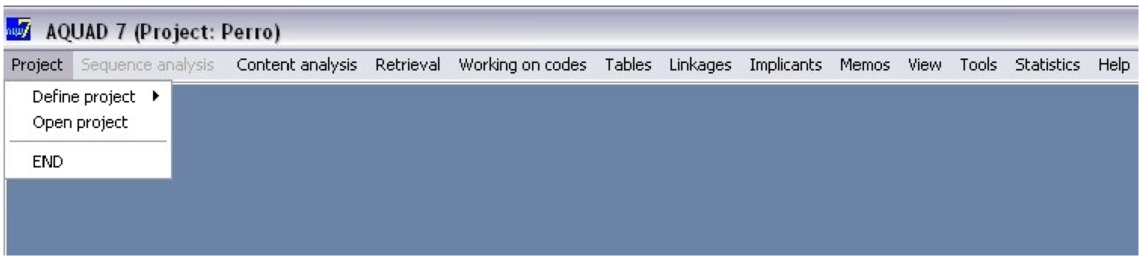
- *Speaker codes* start with two determined characters **/$** and behave like a mixture of profile codes and control codes. They split text files virtually in sub-files during retrieval runs and table analysis. We call them "speaker codes" because they were introduced to enable the analysis of transcriptions of group discussions in total as well as group member by group member (for instance /$Joan, /$John, etc. You could also apply them to a file containing open answers from a questionnaire from many people; thus, you would attribute text segments to people. If each person's answers are saved in a separate file, you could discriminate questions by "speaker" codes (see example in chapters 12.3 and 14.5). Be careful to include the whole text segment of a speaker when coding. Of course, you can apply the same code to several different text segments of the same speaker.

  Attention: Speaker codes are valid only
  - in code retrieval functions (that is, not for counting codes - this is done within "Table Analysis" -> "Frequencies").
  - on the first level of table analyses, that is as column headers on the first level.

Back to the beginning of the process: first all data must be coded. The mechanical part of the operation is greatly facilitated by AQUAD compared with manual coding. The preparations for all types of data, however, are the same: First the data files the researcher wishes to code must be identified for AQUAD and imported (text files in ANSI format or RTF; sound files as WAV or MP3 files; videos in AVI format; graphic data as JPG files). From this point on, AQUAD does not use the original files, but their internal copies, that is, nothing will happen to your original data! Let's now get into the nitty-gritty of coding, one move at a time. We will explain the basic procedures in the paragraphs on coding text data, while only the particularities of coding other data types will be described in the remaining paragraphs (7.1.12 - 7.1.14).

7.1.2 Loading AQUAD and starting the module "QUALitative content analysis"

First all text files, which you want to code, have to be imported as ANSI-text files or in Rich Text Format (*.rtf). From this point on, AQUAD does no longer work with your original transcriptions, but only with internal copies. Let us start coding step by step. After starting AQUAD you see its title page. Clicking on the button "*Open*" opens the main menu. If you have not done it yet, it is time now to determine the permanent data of your project, that is the settings like directories of various files, project name, and the file catalog of data texts (see chap. 2 and 3). To do so, you click on "*Project*", then on "*Define project*", and select the option "*Texts*." These settings are saved automatically and activated whenever you start AQUAD – until you set up another new project or you open another project, which you have already defined.

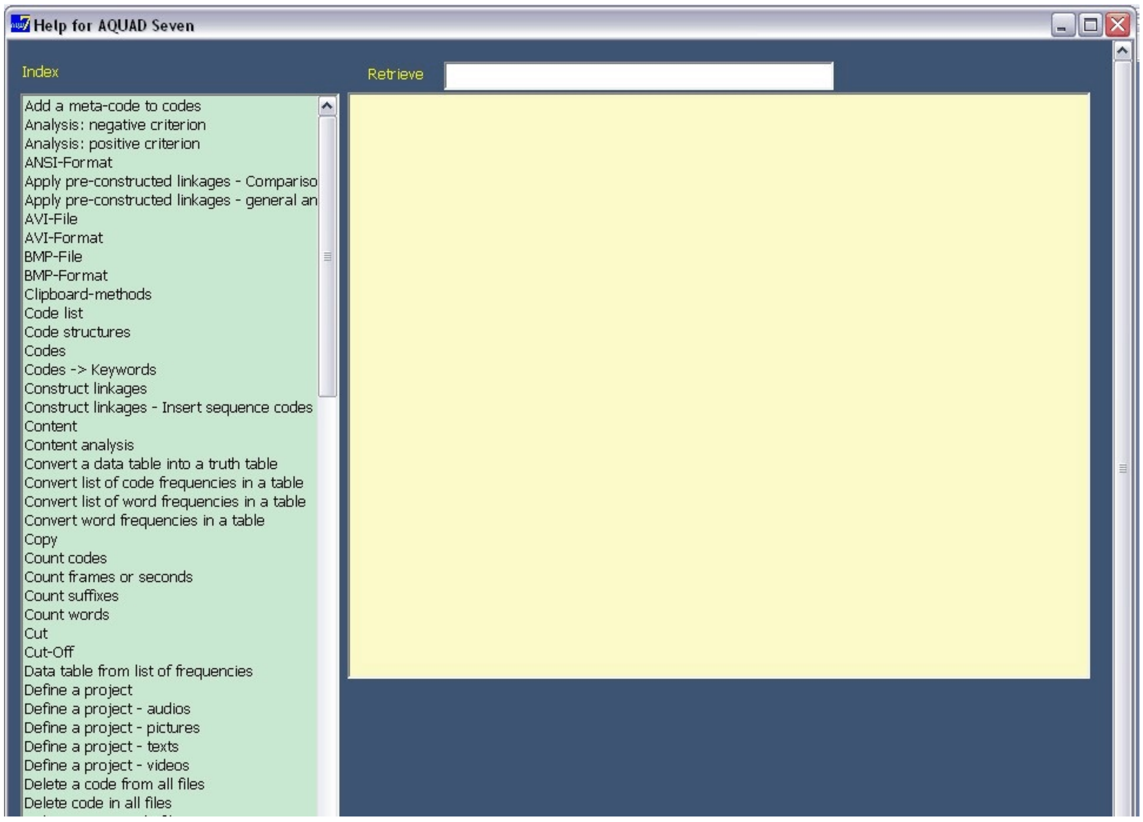


Maybe you would like to get acquainted with coding with AQUAD by working with the sample texts on the distribution CD. To give you an example how to work with AQUAD, you find a fairy tale by the Danish poet Hans Christian Andersen already installed as sample project "Poet" as well as four interviews with teachers (text data and audio data). Here we refer to the shortest example. The text is stored in three files "poet.001", "poet.002," and "poet.003" in ANSI format in the root-directory, which you defined for the installation of AQUAD and – with built-in path definition – for keeping your data texts. Since AQUAD always activates the actual project automatically, you do not have to repeat all the settings again and again. You will jump directly from the opening window to the main menu. To continue your coding, you choose "*Content alaysis*" from the main menu, which gives you among other choices the option "*QUALitative content analysis.*"

Surely you do not want to search every time in the manual in case the meaning of a menu option is not quite clear to you. No problem; that is why AQUAD offers general help for general problems from the main menu and context-sensitive help within various windows. Let us click on the "*Help*" option in the main menu, then on the option "*Content*" and see what happens (screen shot on the next page).

Within he running program AQUAD a help screen appears, offering additional information to the different program modules. We move the mouse pointer over "*Define a project - texts*", which appears in the list of key words in the green part on the left side of this figure or we write a keyword into the small entry field above the yellow window. Clicking finally on our key word opens a screen with some general information on necessary procedures in AQUAD and options for more information (see second screen shot on the next page).

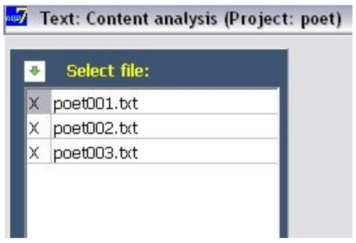
Additional information is accessible by clicking on text lines marked by an arrow ("->"), in the screen shot on the next page, for instance, " -> ANSI-Format".



Add a meta-code to codes
Analysis: negative criterion
Analysis: positive criterion
ANSI-Format
Apply pre-constructed linkages - Compariso
Apply pre-constructed linkages - general an
AVI-File
AVI-Format
BMP-File
BMP-Format
Clipboard-methods
Code list
Code structures
Codes
Codes -> Keywords
Construct linkages
Construct linkages - Insert sequence codes
Content
Content analysis
Convert a data table into a truth table
Convert list of code frequencies in a table
Convert list of word frequencies in a table
Convert word frequencies in a table
Copy
Count codes
Count frames or seconds
Count suffixes
Count words
Cut
Cut-Off
Data table from list of frequencies
Define a project
Define a project - audios
Define a project - pictures
Define a project - texts
Define a project - videos
Delete a code from all files
Delete code in all files

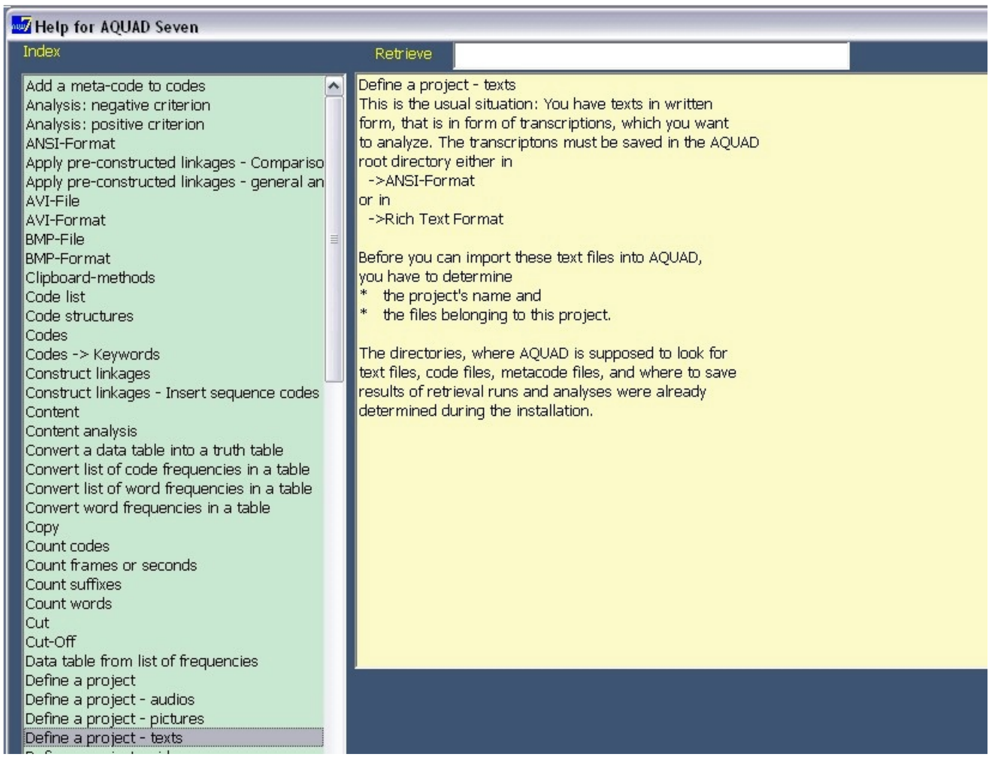
7.1.3 Choosing text files for coding

Let us find out now more about coding. If you click on "*QUALitative content analysis*" in the "*Content analysis*" sub-menu, the coding window will open and display first a box labeled "Select a file". This box contains the names of all data texts in your project. *Before you can start adding codes to a text, you have to decide which text file you want to select for coding*! The file names are shown in this window. Just double-click on one of them. (The sample text "poet" was split into three parts poet.001, poet.002, poet.003 – just to give you some opportunities for getting acquainted with AQUAD. Please, do not look for too much meaning behind this file organization!)



An 'X' in front of the name indicates that there are already codes added to this file, that is you already worked with this text earlier. The example here shows the three sample files that come with AQUAD. Since the samples are already coded, there are Xs in front of the file names. In case this box opens, but *you find nothing to select* in it – then you have not defined a project yet!

If we decide to choose "poet.001," AQUAD will load the main coding window and give access to the options shown on the buttons on the right side. More about that later. Before you click on "*poet.001*", let us consider a problem which may become obvious at this point: The text lines should not bee too long – see chapter 3.


Help for AQUAD Seven
Index
Retrieve
Add a meta-code to codes
Analysis: negative criterion
Analysis: positive criterion
ANSI-Format
Apply pre-constructed linkages - Compariso
Apply pre-constructed linkages - general an
AVI-File
AVI-Format
BMP-File
BMP-Format
Clipboard-methods
Code list
Code structures
Codes
Codes -> Keywords
Construct linkages
Construct linkages - Insert sequence codes
Content
Content analysis
Convert a data table into a truth table
Convert list of code frequencies in a table
Convert list of word frequencies in a table
Convert word frequencies in a table
Copy
Count codes
Count frames or seconds
Count suffixes
Count words
Cut
Cut-Off
Data table from list of frequencies
Define a project
Define a project - audios
Define a project - pictures
Define a project - texts

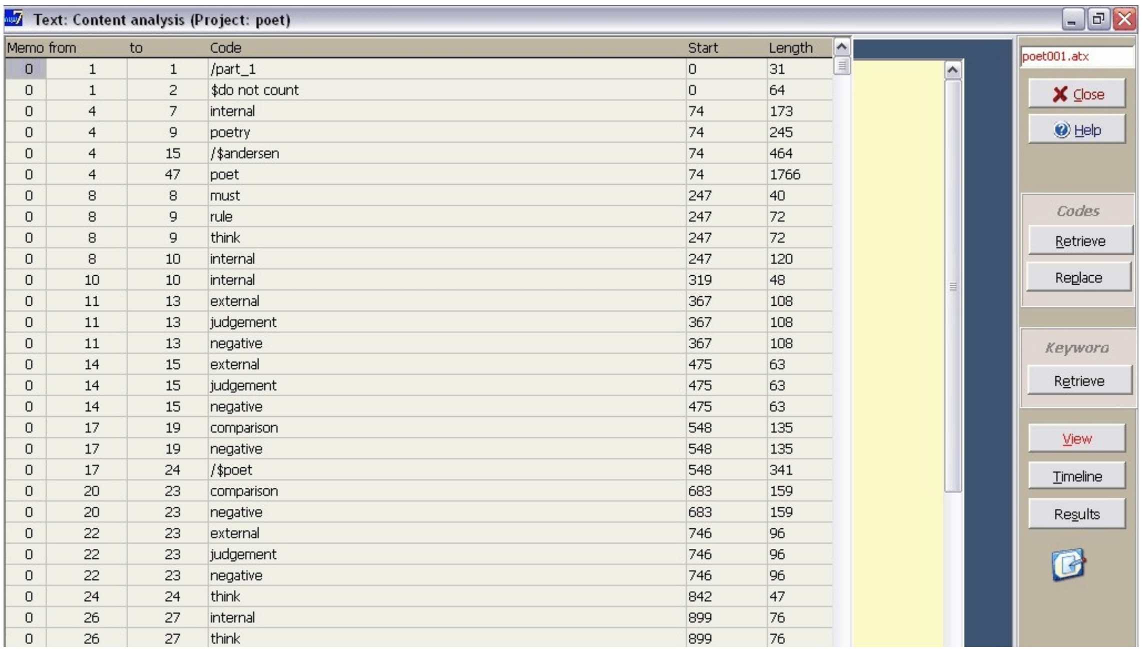
Define a project - texts
This is the usual situation: You have texts in written form, that is in form of transcriptions, which you want to analyze. The transcriptons must be saved in the AQUAD root directory either in
->ANSI-Format
or in
->Rich Text Format

Before you can import these text files into AQUAD, you have to determine
* the project's name and
* the files belonging to this project.

The directories, where AQUAD is supposed to look for text files, code files, metacode files, and where to save results of retrieval runs and analyses were already determined during the installation.

7.1.4 Rules of codification

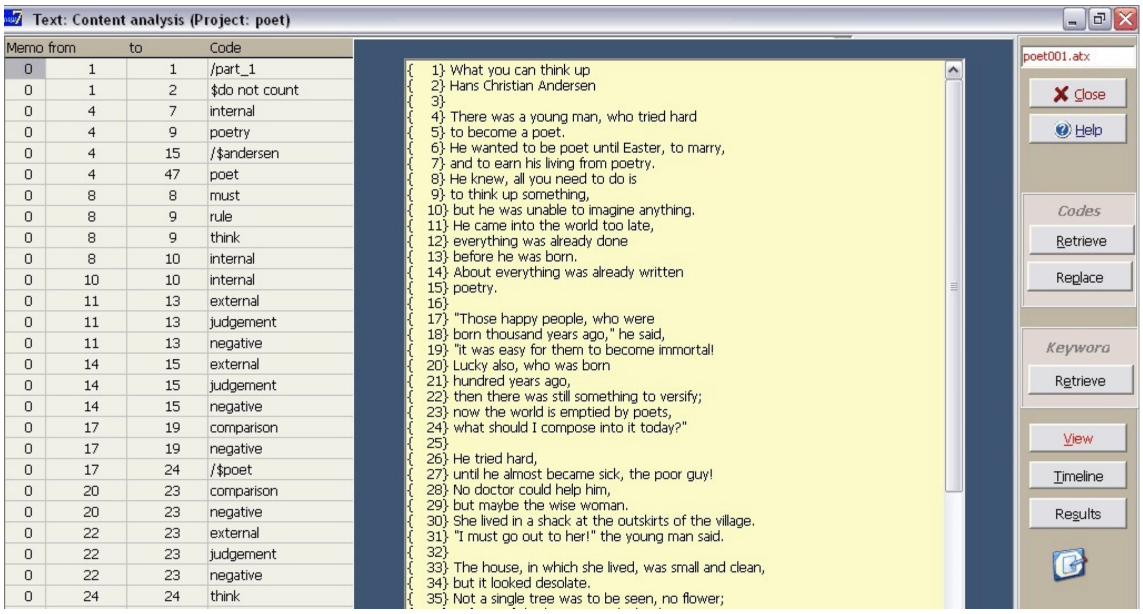
After selecting a file for codification a window will open that shows on the left an empty grey table. In case you have already worked on this text, the table shows all the codes you have attached to the text up to now. On the right side you see the yellow text field, partially overlapped by the code table:



Memo	from	to	Code	Start	Length
0	1	1	/part_1	0	31
0	1	2	$do not count	0	64
0	4	7	internal	74	173
0	4	9	poetry	74	245
0	4	15	/$andersen	74	464
0	4	47	poet	74	1766
0	8	8	must	247	40
0	8	9	rule	247	72
0	8	9	think	247	72
0	8	10	internal	247	120
0	10	10	internal	319	48
0	11	13	external	367	108
0	11	13	judgement	367	108
0	11	13	negative	367	108
0	14	15	external	475	63
0	14	15	judgement	475	63
0	14	15	negative	475	63
0	17	19	comparison	548	135
0	17	19	negative	548	135
0	17	24	/$poet	548	341
0	20	23	comparison	683	159
0	20	23	negative	683	159
0	22	23	external	746	96
0	22	23	judgement	746	96
0	22	23	negative	746	96
0	24	24	think	842	47
0	26	27	internal	899	76
0	26	27	think	899	76

The first column on the left – now filled with "0"s – will later show how many memos you have added to particular codes. The columns "from" and "to" tell us in which lines the coded text segment starts and ends. After the code follow the columns "Start" and "Length". Their contents record an alternative localization of the coded text segment, namely the ordinal number (in the whole text) of the character, with which the segment begins and the number of characters, the segment is composed of. You conclude correctly that text segments can be defined as you like, not just by line numbers.

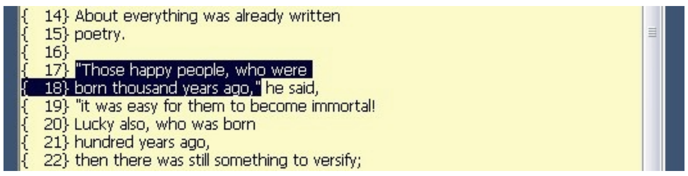
If you now move the mouse pointer over the partially hidden text field, this part of the window jumps into the foreground:



Memo	from	to	Code
0	1	1	/part_1
0	1	2	$do not count
0	4	7	internal
0	4	9	poetry
0	4	15	/$andersen
0	4	47	poet
0	8	8	must
0	8	9	rule
0	8	9	think
0	8	10	internal
0	10	10	internal
0	11	13	external
0	11	13	judgement
0	11	13	negative
0	14	15	external
0	14	15	judgement
0	14	15	negative
0	17	19	comparison
0	17	19	negative
0	17	24	/$poet
0	20	23	comparison
0	20	23	negative
0	22	23	external
0	22	23	judgement
0	22	23	negative
0	24	24	think

```
{   1} What you can think up
{   2} Hans Christian Andersen
{   3}
{   4} There was a young man, who tried hard
{   5} to become a poet.
{   6} He wanted to be poet until Easter, to marry,
{   7} and to earn his living from poetry.
{   8} He knew, all you need to do is
{   9} to think up something,
{  10} but he was unable to imagine anything.
{  11} He came into the world too late,
{  12} everything was already done
{  13} before he was born.
{  14} About everything was already written
{  15} poetry.
{  16}
{  17} "Those happy people, who were
{  18} born thousand years ago," he said,
{  19} "it was easy for them to become immortal!
{  20} Lucky also, who was born
{  21} hundred years ago,
{  22} then there was still something to versify;
{  23} now the world is emptied by poets,
{  24} what should I compose into it today?"
{  25}
{  26} He tried hard,
{  27} until he almost became sick, the poor guy!
{  28} No doctor could help him,
{  29} but maybe the wise woman.
{  30} She lived in a shack at the outskirts of the village.
{  31} "I must go out to her!" the young man said.
{  32}
{  33} The house, in which she lived, was small and clean,
{  34} but it looked desolate.
{  35} Not a single tree was to be seen, no flower;
```

For coding you move the mouse pointer at the beginning of the line or exactly at the beginning of the first word of the text segment that you are going to interpret by attaching a code. Then you move the pointer – keep the mouse button pressed! – behind the last word of the text segment and let the mouse button go:



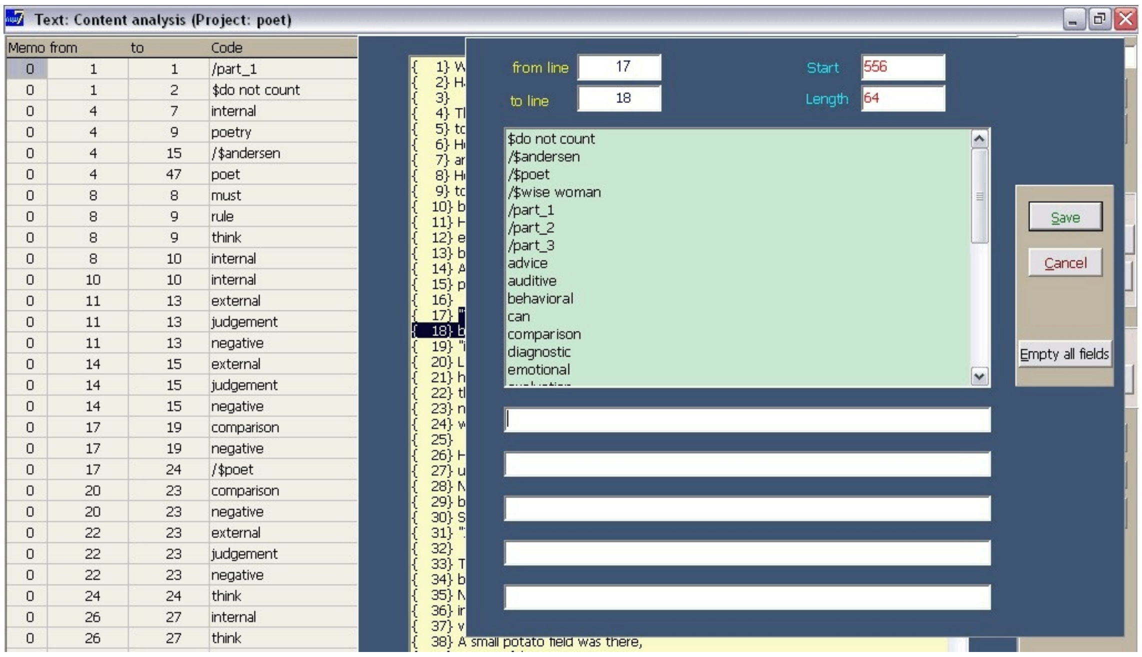
```
{  14} About everything was already written
{  15} poetry.
{  16}
{  17} "Those happy people, who were
{  18} born thousand years ago," he said,
{  19} "it was easy for them to become immortal!
{  20} Lucky also, who was born
{  21} hundred years ago,
{  22} then there was still something to versify;
```

Two things are important now: First, at the right we are just able to see that the selected text segment is now marked in black color. Second, the window for entering codes is visible now. Our selection is entered automatically: The text segment is located in lines 17 and 18, starts at exactly the 556. character of the file and is 64 characters long.

From the master code list, shown in the light blue window in the upper part, we may select one of the already available codes or we write a new code in one of the five entry slots. Your conclusion is correct: You can attach up to five codes to a text segment in a single step. After clicking on the button "Save" the code/s are stored, new codes are integrated into the master code list, and the added code/s appear/s on he left in the grey code table. This table is moved again to the foreground, if you touch it with the mouse pointer.

It is recommended to add a short definition to each new code name. Thus, you can retrieve quickly the exact meaning of a code and you are able to describe your coding system later very simply showing codes, definitions, and an example for each code. To add definitions or comments you use the function "*Memo*". It is activated by clicking into the first cell on the left in the row of your code (see above). More details are found in paragraph 7.1.7 below.

Please, remember when marking something in the text field: The text is your data base and you cannot alter your data during coding. AQUAD just *shows* the text, but does not allow any changes. However, you may copy parts of the text (or the complete text) and paste it into a memo. For copying you just click on the *right* mouse button while within the text field. For pasting you do the same within the memo field.



Text: Content analysis (Project: poet)

Memo	from	to	Code
0	1	1	/part_1
0	1	2	$do not count
0	4	7	internal
0	4	9	poetry
0	4	15	/$andersen
0	4	47	poet
0	8	8	must
0	8	9	rule
0	8	9	think
0	8	10	internal
0	10	10	internal
0	11	13	external
0	11	13	judgement
0	11	13	negative
0	14	15	external
0	14	15	judgement
0	14	15	negative
0	17	19	comparison
0	17	19	negative
0	17	24	/$poet
0	20	23	comparison
0	20	23	negative
0	22	23	external
0	22	23	judgement
0	22	23	negative
0	24	24	think
0	26	27	internal
0	26	27	think

38} A small potato field was there,

from line: 17
to line: 18
Start: 556
Length: 64

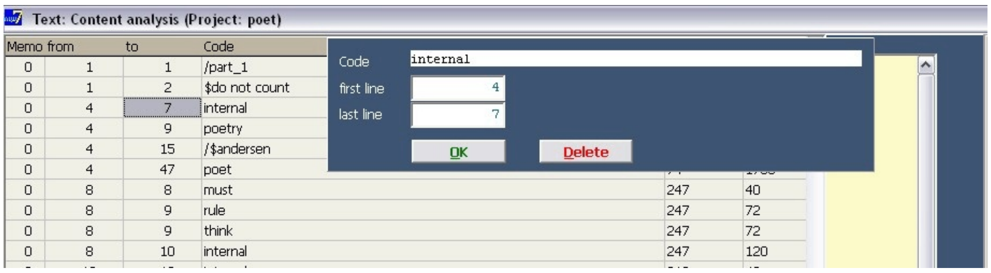
$do not count
/$andersen
/$poet
/$wise woman
/part_1
/part_2
/part_3
advice
auditive
behavioral
can
comparison
diagnostic
emotional

7.1.5 How to delete or replace codes?

If you click on "*Cancel*" (instead of "*Save*") in the coding window (see above), your last entry or entries are just ignored. In case you detect a typing error before you have clicked on "*Save*", just double click into the entry slot that contains the erroneous code and type the code anew. Of course, you may move the cursor within the slot, erase characters or overwrite them – just as in a text processor.

But what to do, if you find out later, after saving a code and attaching it to a text segment that there are typing errors or the code does not match the meaning of the text correctly? Of course, you now want to eliminate any error. Here a first suggestion how to proceed. Later we show a more elegant, but maybe also more dangerous possibility to retrieve codes and replace them automatically.

- Find the critical code in the code table on the left and click into the preceding cell "*from*" or "*to*". A small blue window will appear (see screenshot below) and can be moved around on the screen.
- To delete the selected code you click on "*Delete*".
- Afterwards you can enter a correct code name.

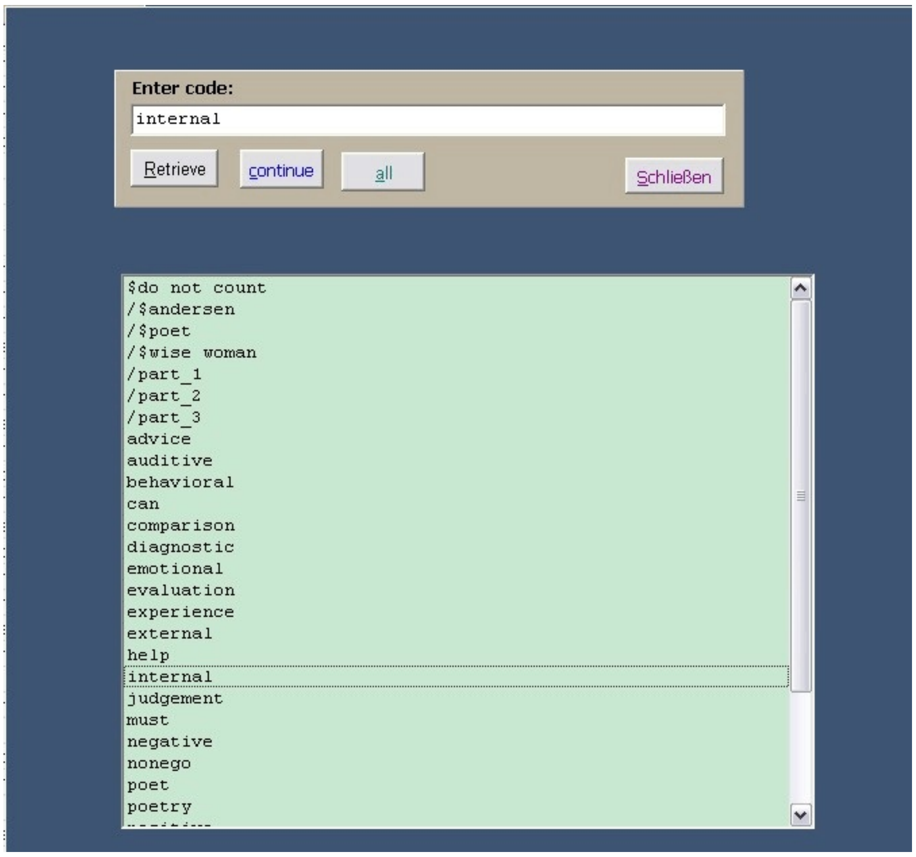


Text: Content analysis (Project: poet)

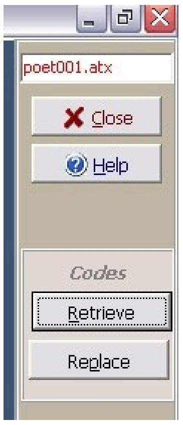
Memo	from	to	Code		
0	1	1	/part_1		
0	1	2	$do not count		
0	4	7	internal		
0	4	9	poetry		
0	4	15	/$andersen		
0	4	47	poet		
0	8	8	must	247	40
0	8	9	rule	247	72
0	8	9	think	247	72
0	8	10	internal	247	120

Code: internal
first line: 4
last line: 7

However, deleting codes with this function is a somewhat annoying work, if you have to delete several codes. Therefore, there is a field "*Codes*" on he right margin of the window, containing two buttons "*Retrieve*" and "*Replace.*" The "*Retrieve*"-button is what we need now. Click on it and the following blue window will appear:



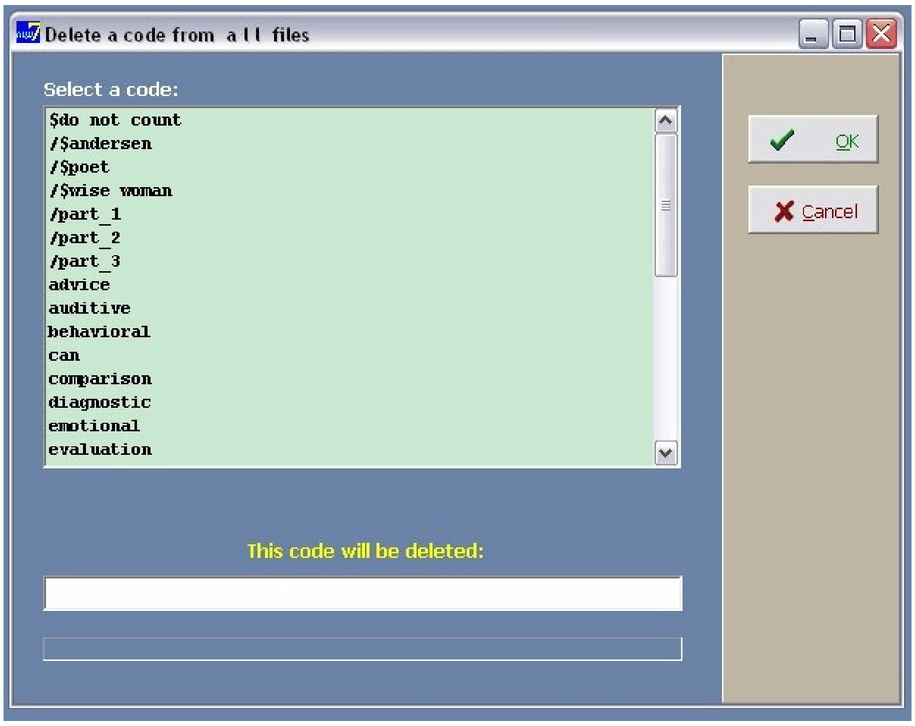
```
$do not count
/$andersen
/$poet
/$wise woman
/part_1
/part_2
/part_3
advice
auditive
behavioral
can
comparison
diagnostic
emotional
evaluation
experience
external
help
internal
judgement
must
negative
nonego
poet
poetry
```



Here, too, you do not have to write the code into the entry slot above, but you double click into the slot and again the master code list shows up (see above). Here you select the critical code as criterion of a retrieval run by double clicking on it. In this moment the code will appear in the entry slot and all that remains to do is clicking on the "*Retrieve*"-button below the code and start the retrieval.

You will be led automatically from one text segment to the next to which the critical code was attached. Then you do what was shown above: Click into the cell in front of the code showing the number of the first or the last text line and click on "*Delete*" in the small window you already know (see above in 7.14). This procedure is safe, because you can control it step by step. However, it takes time.

More elegant is the function "*Delete code in all files,*" which you find in the main menu in he group "*Working on codes*" – but the function does exactly what it says, so be careful. It is a good idea, to keep always an actual copy of your code files (content of sub-directory ..\cod) in a different directory.



Select a code:

```
$do not count
/$andersen
/$poet
/$wise woman
/part_1
/part_2
/part_3
advice
auditive
behavioral
can
comparison
diagnostic
emotional
evaluation
```

This code will be deleted:

Of course, you can also replace a code by a different one – step by step or automatically in all files. Here first the less dangerous option:

- Click in the field "*Code*" on the right on the button "*Replace*."
- Double click into the empty entry slot below the word "Retrieve" and select from the master code list the code you want to replace.
- Into the empty slot "Replace by" you write a new code or you select a code frm the master code list.
- On clicking on the "*Replace*"-button AQUAD does what you expect it to do: It looks for the first code in the actual code table and replaces it by the second code until the end of the table is reached. That is, within the actual file all critical codes ("Retrieve") are replaced.
- Faster but more risky – because replacing takes place in *all code files of the actual project* – is the function "*Replace code in all files*" in the group "*Working on codes*" in the main menu.

7.1.6 How to get an overview on codes?

There are a number of reasons why a better overview than the presentation in the code table may be necessary. AQUAD offers four options:

Overview on all codes used in a project

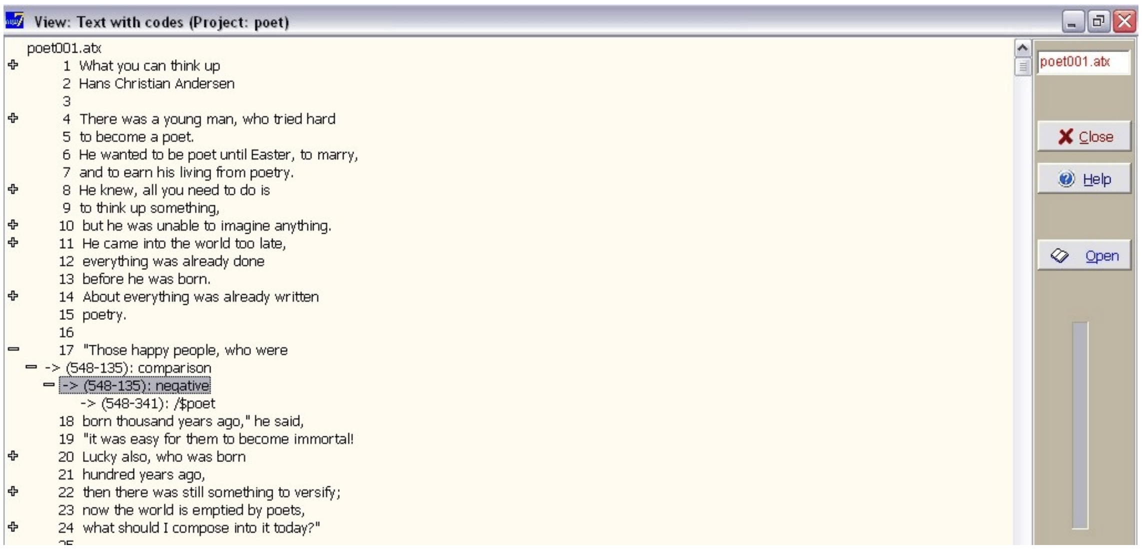
As already mentioned, AQUAD does a sort of bookkeeping during coding and offers at any time during your work the actual state in form of a master code file. There, all codes are registered and sorted alphabetically in a list, from which you may select any code by clicking on it.

Overview on all codes attached to a particular line

The yellow window shows the actual text file. Segments to which already one or more codes were attached are marked, if you click into the corresponding rows of the gray code table (see above). The line numbers, in which the text segments begin, are listed in the column "*from*." Just click on all cells containing the same line number for segment's first line.

Overview on all codes attached to a particular text file

The button "*View*" at the right margin gives an overview on a text file together with its codes in an outline format:



```
poet001.atx
+     1  What you can think up
      2  Hans Christian Andersen
      3
+     4  There was a young man, who tried hard
      5  to become a poet.
      6  He wanted to be poet until Easter, to marry,
      7  and to earn his living from poetry.
+     8  He knew, all you need to do is
      9  to think up something,
+    10  but he was unable to imagine anything.
+    11  He came into the world too late,
     12  everything was already done
     13  before he was born.
+    14  About everything was already written
     15  poetry.
     16
-    17  "Those happy people, who were
   - -> (548-135): comparison
     - -> (548-135): negative
         -> (548-341): /$poet
     18  born thousand years ago," he said,
     19  "it was easy for them to become immortal!
+    20  Lucky also, who was born
     21  hundred years ago,
+    22  then there was still something to versify;
     23  now the world is emptied by poets,
+    24  what should I compose into it today?"
```

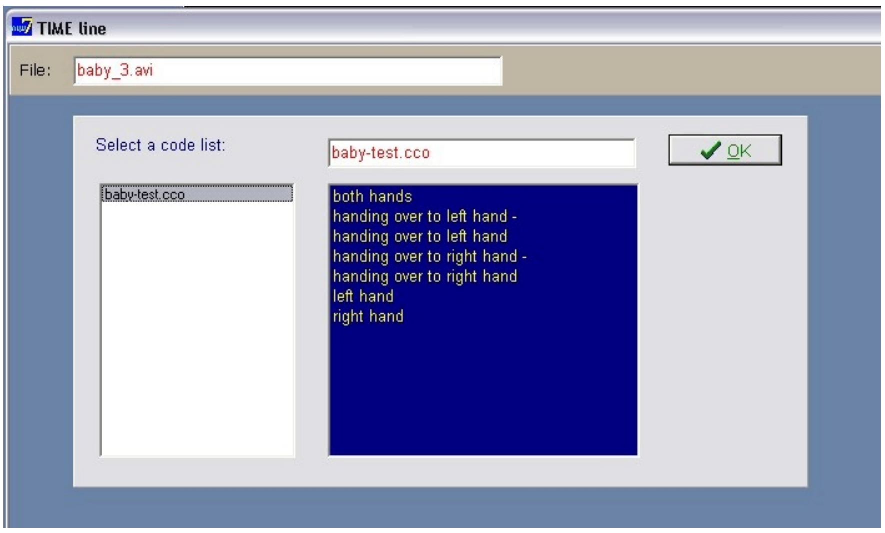
An attached code is shown whenever you click on the "+" symbol. All codes at once are shown after clicking on the button "*Open*" on the right.

Overview on the codes of a file in sequential order

A selection of interesting codes is collected in a code catalog (see -> "*Retrieval*" -> "*Code catalog*" -> "*Create code catalog*") and shown in their alternating sequence within the window of text-, audio- or video-projects after clicking on the button "*Timeline*".

Since the particular purpose of this function is to show clearly a sequence of codings, the number of codes available at the same time is limited to 50. However, you should in any case apply a much smaller number of codes when working with this function.

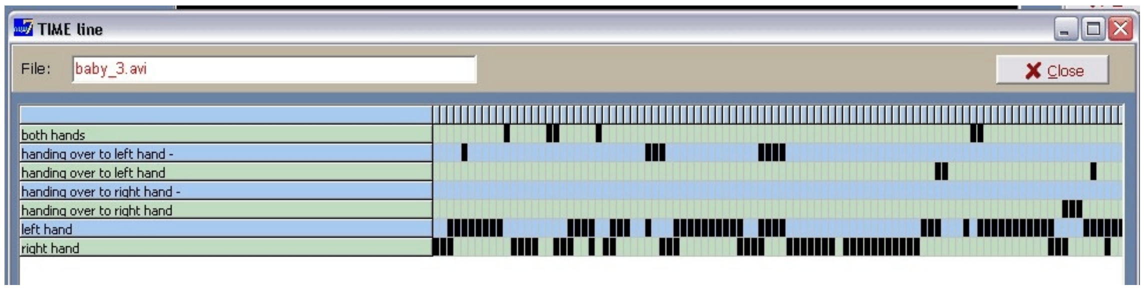
Within the project "Baby" we have created a code catalog "baby-test.cco," which we now select for a sequential overview:



both hands
handing over to left hand -
handing over to left hand
handing over to right hand -
handing over to right hand
left hand
right hand

After you click on "*OK*" in this window, AQUAD creates a table with a row for each code in the code catalog. The appearance of a coded data segment is represented and related to the other codes by tiny black blocks. In our example, the unit of representation are frames, each unit corresponding to 25 frames (one second). In case of audio recordings the units are seconds, in case of texts a unit corresponds to one textline. Of course, the concrete scores (number of text lines, seconds, frames) underlying the representation of units often are rounded, therefore you will see tiny gaps or overlaps of blocks. However, the purpose of visualizing the sequence of codes is not impaired – on the contrary, the position of each coded segment to each other in the selection of codes is clearly visible.

The figure below shows as an example the resulting sequential representation of codes in the code catalog "baby-test.cco" within the video file "baby_3.avi":

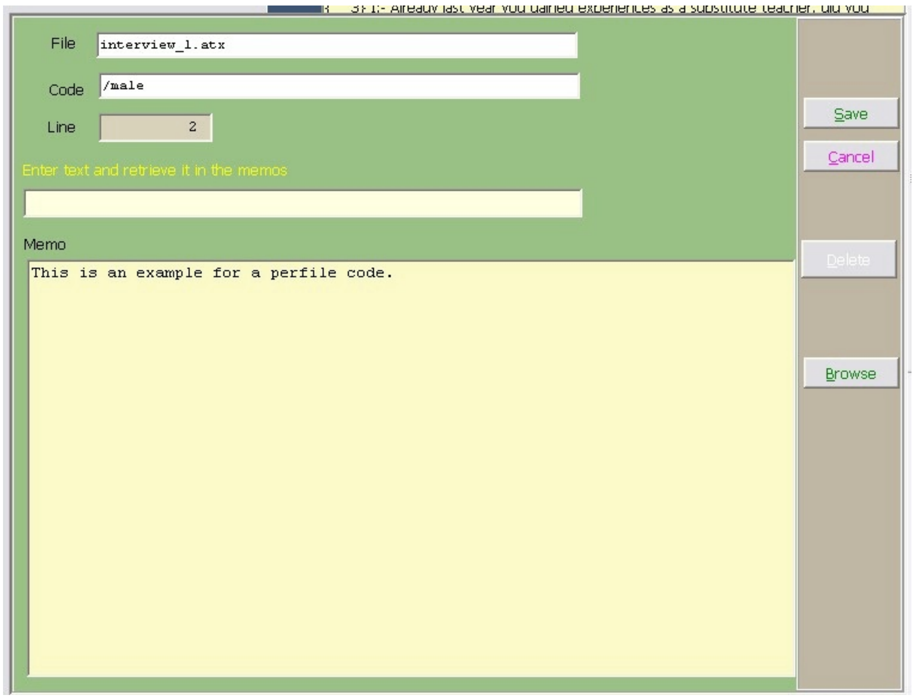

TIME line
File:
baby_3.avi
Close
both hands
handing over to left hand -
handing over to left hand
handing over to right hand -
handing over to right hand
left hand
right hand

The last two rows show how the baby again and again hands the object she is playing with, from the right hand to the left hand and back again, however, not always successfully (dropping the object!), as can be seen in row 2 ("handing over to left hand -").

7.1.7 How to add some comments to your codes?

During coding of text files or during analyzing pictures, audio- or video-recordings you add codes to your data files. While doing so, you may activate the memo-function and add comments to any code by clicking either in the green "Memo" column on the left side of the window or on the "*Memo*" option in the main menu.

In those rows of the code table to which no memo was attached until now, you will find in the "Memo"-column a "0", the numbers in other rows may show that here are already attached one or more memos. Clicking on any cell in this column opens the memo window with the first of potentially many memos attached to the data segment specified in this row of the code table. Clicking on a "0" cell in the "Memo" column opens an empty window for memo input.



File interview_1.atx
Code /male
Line 2
Enter text and retrieve it in the memos
Memo
This is an example for a perfile code.

If you try to retrieve memos later or if you activate the "*Memo*" option in the main menu (see chap. 10) you may define three characteristics for each of the memos:

- The name of the file and
- the code to which the memo is attached;
- the number of the text line, where the memo was inserted.

Additionally you can enter a keyword or part of a sentence into the slot below the yellow headline "*Enter text and retrieve it in the memo.*" AQUAD will then try to retrieve this text in your memos. You could, for instance, write code definitions as memos and enter always as first line "Code definition." Applying this keyword you assemble rapidly all code definitions after clicking on the button "Browse."

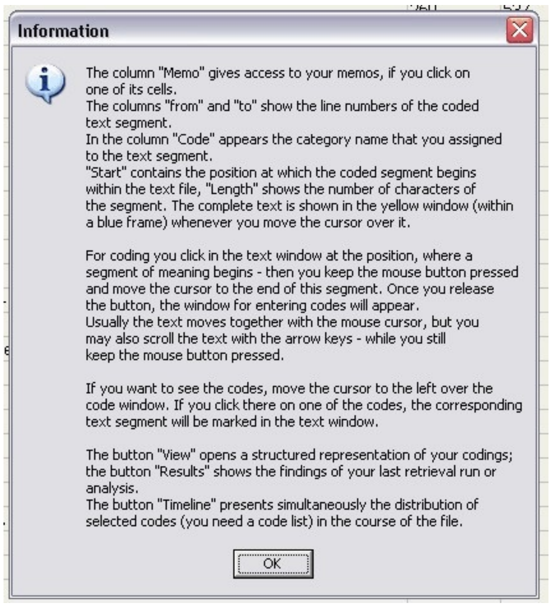
The big yellow window does not need explanations: Here you write (or import) your memo. The right mouse button opens access to the functions "Copy," "Cut," "Paste" etc. The three buttons above on the right side do exactly what they say: You can enter new memos, cancel the memo function or delete already existing memos.

If you decide to "Browse" through your memos and combine this command with particular characteristics of your memos – like, for instance, retrieving a particular keyword (see above) or retrieving all memos for a particular code – you should preferably activate the memo function from the main menu.

7.1.8 In case a little help is necessary

In paragraph 7.1.2 we have already described how you get help by activating the "Help"-function in the main menu. Of course, there is also context-sensitive help during coding and during other operations. Just click on the "Help"-button on the right margin of the windows. A small help window will present you the most important informations.

Here is an example. If you ask for context-sensitive help while the main window of "*Text: Content analys*is" is open, you will see the following text:



The column "Memo" gives access to your memos, if you click on one of its cells.
The columns "from" and "to" show the line numbers of the coded text segment.
In the column "Code" appears the category name that you assigned to the text segment.
"Start" contains the position at which the coded segment begins within the text file, "Length" shows the number of characters of the segment. The complete text is shown in the yellow window (within a blue frame) whenever you move the cursor over it.

For coding you click in the text window at the position, where a segment of meaning begins - then you keep the mouse button pressed and move the cursor to the end of this segment. Once you release the button, the window for entering codes will appear.
Usually the text moves together with the mouse cursor, but you may also scroll the text with the arrow keys - while you still keep the mouse button pressed.

If you want to see the codes, move the cursor to the left over the code window. If you click there on one of the codes, the corresponding text segment will be marked in the text window.

The button "View" opens a structured representation of your codings; the button "Results" shows the findings of your last retrieval run or analysis.
The button "Timeline" presents simultaneously the distribution of selected codes (you need a code list) in the course of the file.

7.1.9 Why not have AQUAD do more work: semi-automatic coding

AQUAD's retrieval functions offer opportunities to run computer-assisted retrievals of keywords as an approach to semi-automatic coding. We click on the button "*Keyword*" on the right side, type a keyword (or a sequence of words or part of a word) into the entry field of the pop-up window in the upper right corner – and have AQUAD scan the text.

For example, we might be interested where in our sample texts "Interview" something is told about *parents*. We would enter the key word "parent" and click on the button "*Retrieve*". Often a simple procedure like this will help us to find quickly critical text segments. But we have to be careful: The retrieval procedure does not only register the isolated appearances of "parent", but also where this sequence of characters is located within other words (for instance, in "grandparents"), and the procedure is not case-sensitive, that is, it does not distinguish capital letters and small letters. Remember:

We always have to check and to interpret the findings!

What can you do, if more than one key word is relevant for a search run or if several words define a field of meaning relevant for your analysis? Let us assume we want to find out in all our texts poet.xxx, whether there are mentioned more perceptions than thoughts and imaginations. We could compose a catalog of relevant words or word stems like *listen*, *hear*, *see*, *look*, *ear*, *eye*, etc. Unfortunately, using the keyword option forces us to apply the keywords one after the other to our text file. A smarter way to retrieve the locations of keywords is possible if you use the keyword routine in the module "R*etrieval*." If you want just a fast overview on one or a series of keywords in all of your project's files (that is, without loading one file after the other), you can apply the option "*Count words*" from the module "Text: QUANtitative content analysis." You read more about these possibilities in paragraph 7.5 of this chapter and in chapter 9.

7.1.10 Reading codes from a print-out

The code table on the left in the window of "QUALitative content analysis" gives an overview on your codes, however, only on the screen while your computer is running. How to get a print-out of the codings?

- Activate the function "*QUALitative content analysis*";
- select the text file for which you want to have a print-out of codes;
- click with the *right* mouse button somewhere into the column "*Codes*": A small window saying "*Print*" will pop up – click on this option.
- If you want to select a particular printer, fonts or layout (margins!) of the print-out, you activate in the next window the option "*Layout*."
- Then you click on the option "*Print*" – that is, the printer icon in the upper margin of the window – and your printer will do its work in case you have already switched it on.

7.1.11 What is a master code file – and why would you want to delete it?

Let us introduce some terminological differentiations:

- Codings are composed of the name of a data file, of the numbers of first and last line of a coded text segment and of the *code*.
- Codings are stored in *code files*.
- When we talk here about a master code file, we refer to the list of codes used in all files of your project (the "master code list").

There are several occasions when AQUAD wants to know which of your many codes you would like to have included in an analysis. The most simple case is attaching codes to a text file. If you want to select code names by clicking instead of always typing them again and again, AQUAD has to keep book on your code names. Another case is the situation when you want to find out which codes were used in which texts how many times. As a prerequisite for an analysis of frequencies of codes, AQUAD has to know which codes to retrieve and to count – all codes or just a selection of particular interest for your research question? Even if you want to have all codes counted, AQUAD has to learn which codes you have used until now – otherwise you would not find out whether some codes were never applied in some of the texts.

That is, you need a catalog of code names. During coding AQUAD does a sort of book-keeping and registers all newly introduced code names. Whenever you use a code for the first name, it will be sorted alphabetically into the *master code list*. For many analyses you may select codes from this list. Depending on the type of analyses only a few codes may be meaningful; in this case you select codes by clicking on them in the master code list. If in principle all codes could be included together for an analysis, as for instance when counting their frequency, you just delete those codes which you want to be excluded. Of course, your master code list remains undamaged by this operation; you are always working with a copy of the original catalog.

AQUAD will construct and maintain this master code list automatically, so why should you want to delete it? Well, there may be an inconvenience: When you play with the opportunities of replacing codes by meta-codes – and you should play with your ideas!! – as costs of the time-saving strategy of automatically checking the master codes it may happen that some code entries appear twice or even more often in the master list. The same may be true if you edit your code files manually outside of AQUAD. From the options in "*Working on codes*" you select "*Delete the master code file*" in this case.

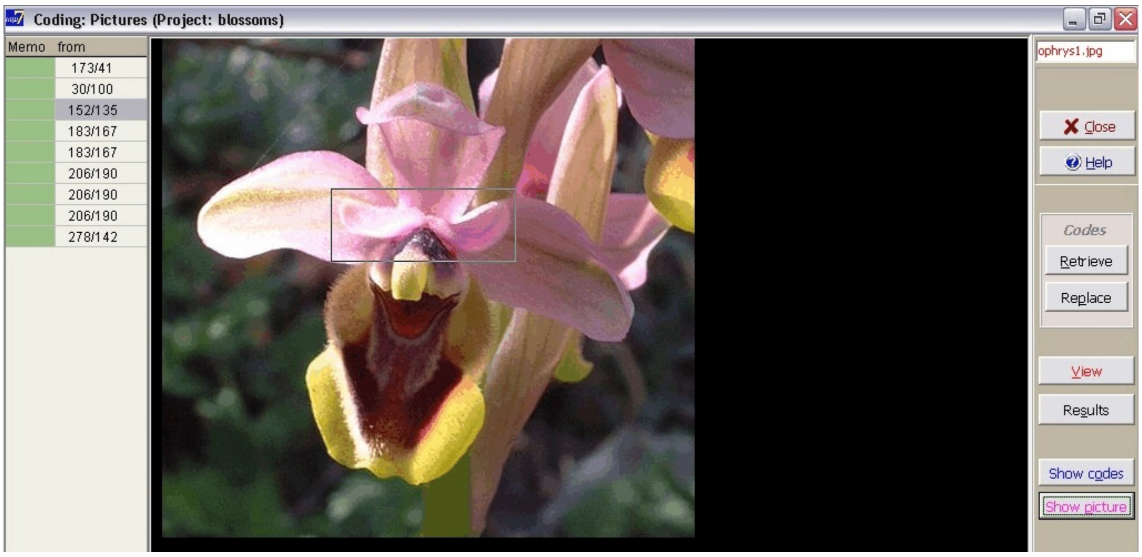
While this will not cause any problems of functionality, it is against the logic and the aesthetics of a master list. *To repair the master code entries just delete them and have them created again.* You get a new, refined code list easily, if you afterwards go to "*Working on codes*" again, where you now click on "*Reconstruct the master code file.*" The new master code file will be created from all code files attached to the text files listed in your project.

7.1.12 How to code graphic data

The windows and functions of AQUAD for coding interesting segments of graphic data correspond to the functions that were described above, when we explained how to code text segments.

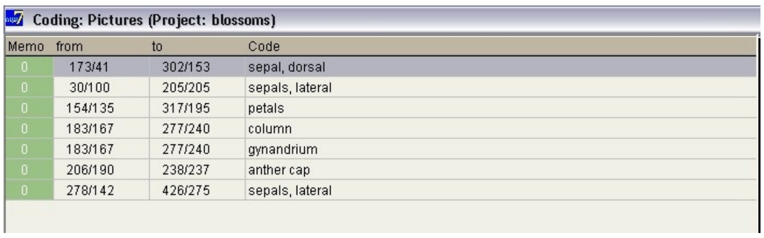
Principles

The screen shot below shows on the left hand the table of codes, on the right hand a photo we are just about to code. We mark relevant data segments with the mouse (clicking with the left mouse button on the upper left or lower right corner of the area we want to code, keeping the button pressed, drawing a frame around the critical area, and releasing the button): In this photograph of an ophrys blossom we just marked the petals (see gray frame below); AQUAD shows the coordinates (x1/y1 and x2/y2) underneath the photograph. Once we click on the button "*Coding*", these data are transferred into the code table (see left side of the screen shot).



Memo	from
🟩	173/41
🟩	30/100
🟩	152/135
🟩	183/167
🟩	183/167
🟩	206/190
🟩	206/190
🟩	206/190
🟩	278/142

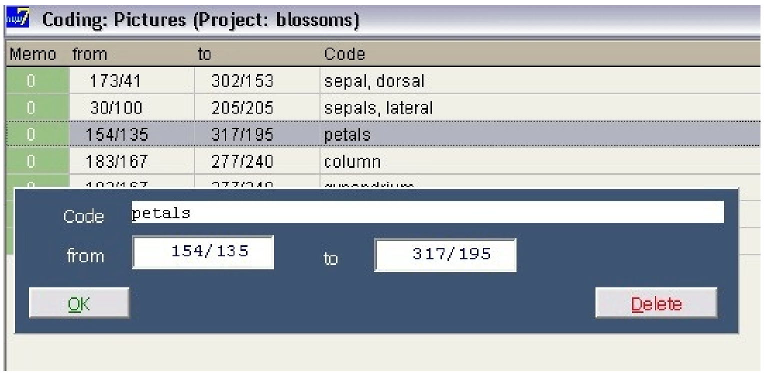
If you move the mouse cursor to the left over the code table, this table will be put into the foreground:



Coding: Pictures (Project: blossoms)

Memo	from	to	Code
0	173/41	302/153	sepal, dorsal
0	30/100	205/205	sepals, lateral
0	154/135	317/195	petals
0	183/167	277/240	column
0	183/167	277/240	gynandrium
0	206/190	238/237	anther cap
0	278/142	426/275	sepals, lateral

If you click on a code name in the code table, the corresponding segment of the image is marked on the right side. Deleting codes follows he same rules as you know from working with text data. After clicking in one of the cells, where the coordinates of the coded segment are shown, you will see the small blue window for deleting codes.



Coding: Pictures (Project: blossoms)

Memo	from	to	Code
0	173/41	302/153	sepal, dorsal
0	30/100	205/205	sepals, lateral
0	154/135	317/195	petals
0	183/167	277/240	column

Code: petals

from: 154/135 to: 317/195

Unlike other types of data, graphic data do not permit to edit codifications manually because of their somewhat complex localization of a data segment by determining the x/y-coordinates of its upper left and lower right corner. Also it is not possible to retrieve directly coding structures (see chapter 8.1) or linkages of codifications (see chapter 8,2). Until now, users of AQUAD did not complain about it. As mostly, there is a work-around that allows to retrieve structures of coding and test linkages even of graphic data, however, this trick seems to make sense only, if we analyze pictures showing texts, for instance, scanned handwritings (see next paragraph).

Since all picture files, which exceed the space reserved on the screen, are reduced in AQUAD to this size, the software offers two "magnifying" functions, if details are of interest: Parts of small picture files (originally smaller than the fixed size) can be zoomed (button "*ZOOM*" at the right margin), larger pictures will be shown in parts of original size in an extra window (button "*Original size*" at the right margin) and can be scrolled in this window.

In some research projects it may be interesting to work with quantitative data from graphic files, for instance children's drawings: How big was the father drawn as compared to mother and siblings? Which part of the area is covered by a specific picture element in different children's drawings? We can answer questions like these by the function "*Add height/width codes*" (option "*Picture codes*" in the group "*Tools*" in the main menu), which does exactly what it says: The function computes height and width of selected segments (= coded areas) of the pictures as difference of pixel coordinates and adds new codes to our code files. You find a detailed description in paragraph 13.8.2.

Coding texts available in graphic format (JPG)

In some projects researchers have to analyze printed or handwritten texts (for instance, protocols of conferences, political speeches, etc. or letters, diary notes, written answers to open questions in questionnaires, etc.). In these cases we cannot profit from the analytic possibilities described above: Because we do not have text files, it is impossible to mark units of meaning in the text on the screen.

There are, however, two possibilities to localize text segments even in graphic files:

(1) In case you have scanned printed texts, you can convert the resulting graphic files (*.jpg) with OCR-programs (for instance, with "Omniscan") into regular text files and work on them like described above.
(2) In case of handwritten texts saved jpg-format you start to code them like other graphic data. If you want later analyze later structures of codes and hypothetical linkages between codes, there is a function "*Convert into text codes*" (see option "*Picture codes*" in the group "*Tools*"). According to the internal logic of working with pictures, however, each text page is normally saved in a separate file.

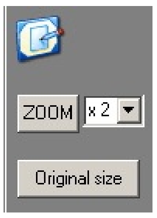
Subsequently, we describe the necessary procedures referring to a concrete project (Huber & Roth, 2004), in which we analyzed among other data handwritten "learning diaries" of teachers participating in an in-service training. To comment their classroom experiences, the teachers received every week a page A4 with some standardized topics (for instance: What went well this week: ...). We scanned these diaries and saved the files in JPG-format. However, it is impossible to read the handwritten entries on the screen after AQUAD makes them smaller and puts them into its picture frame. That is why we insert relevant parts in original size (button "*Original size*") – or we just read the original pages. On the screen we enclose those text segments, which we want to code, in a rectangle as described above ("Principles").

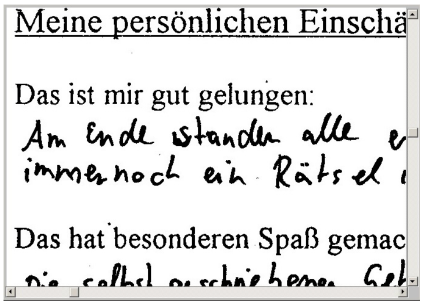
Later, if we want to retrieve code structures or test linkage hypotheses, we convert the original picture codes into text codes. You find
-> a group of functions "*Tools*" in the main menue;
-> there, after moving the cursor on "*Picture codes*"
-> appears an option "*Convert into text codes*".
This option eliminates in the coordinates "from" (upper left corner of the marked areas) and "to" (lower right corner) the x-components. What is left are the scores of the y-components, that is information about beginning and end of the marked segment on the vertical axis – a functional analogy of line numbers in true text files. AQUAD saves the original codes and applies them again automatically if necessary (for instance, if we decide at some point to return to the phase of coding, to modify our categories, etc.). We have to initiate the conversion into text codes ourselves if we need them again after a phase in which AQUAD activated the original codes again. An example can be found in paragraph 13.7.1.

The function "zoom" vs. viewing pictures in original size

Below the icon button, which opens a RTF editor, you see at the right margin two buttons, which give access to detailed viewing of graphic files. These functions are mutually exclusive and work only depending on determined characteristics of graphic files:

(1) ZOOM: Graphic files, which do not exceed the size of the picture area on the screen, that is which are not downsized to match the picture area of AQUAD, can be ZOOMed.

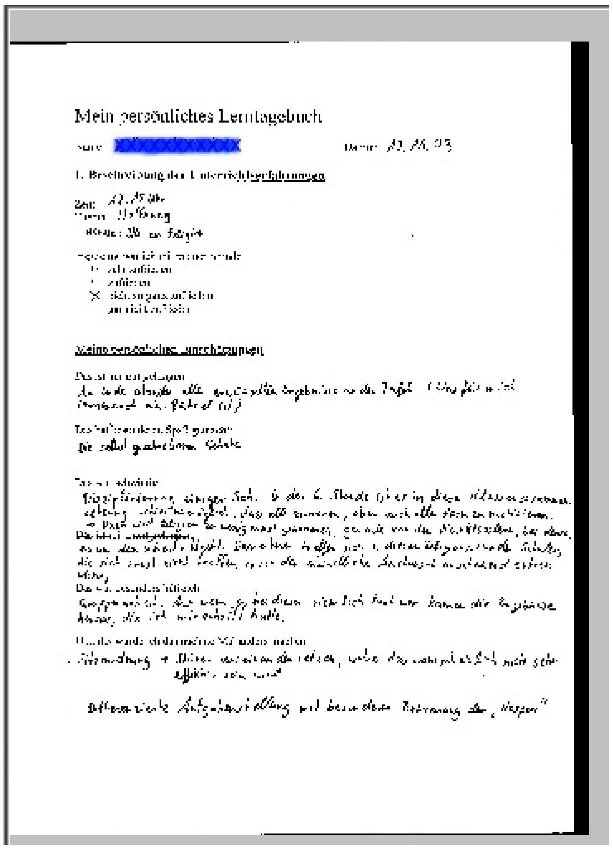




Meine persönlichen Einschä

Das ist mir gut gelungen:
Am Ende standen alle e
immer noch ein Rätsel

Das hat besonderen Spaß gemac
Die selbst geschriebenen Geh



Mein persönliches Lerntagebuch

Name: [redacted] Datum: [illegible]

1. Beschreibung der Unterrichtserfahrungen

[illegible]

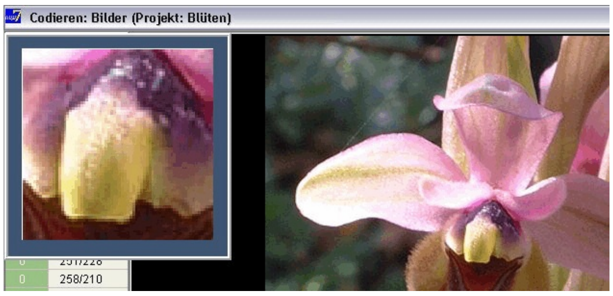
Meine persönlichen Einschätzungen

[illegible]

Clicking on "*ZOOM*" opens a small additional window (you can drag it around) and shows a magnified part. Its size depends on the zoom factor selected.

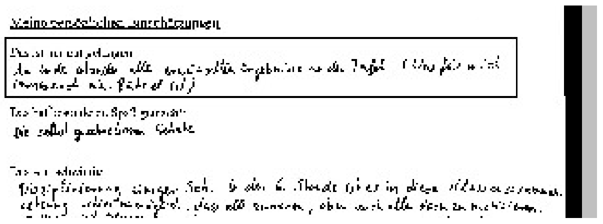
The content changes corresponding to movements of the mouse cursor in the field of the original picture. Clicking a second time on the "*ZOOM*" button closes the magnification again.

Under this condition, the button "*Original size*" is *not* in operation.



(2) Original size: If at least one dimension of the original file exceeds the size of the picture area, the graphic is downsized into the determined screen area. As a consequence, we will not be able to decipher any finer details, specifically it will be impossible to read printed or handwritten text (see the following example of a "learning diary" from the study described above).

The button "*Original size*" inserts an additional window (moveable) that shows part of the file – as the button announces – in original size. Horizontal and vertical scroll bars permit to view exactly the relevant area of the picture:



[illegible]

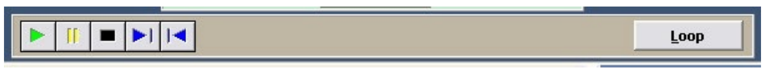
The rectangular frame necessary for determining a data segment for coding (see above) is drawn in the standard window for graphic files; otherwise the y-coordinates would not be comparable. To close the additional window, you click as usual on the "X" icon in the upper left corner of the window frame.

Under this condition, the button "*ZOOM*" is *not* in operation, because enlarging parts of the compressed file would not contribute to better readability.

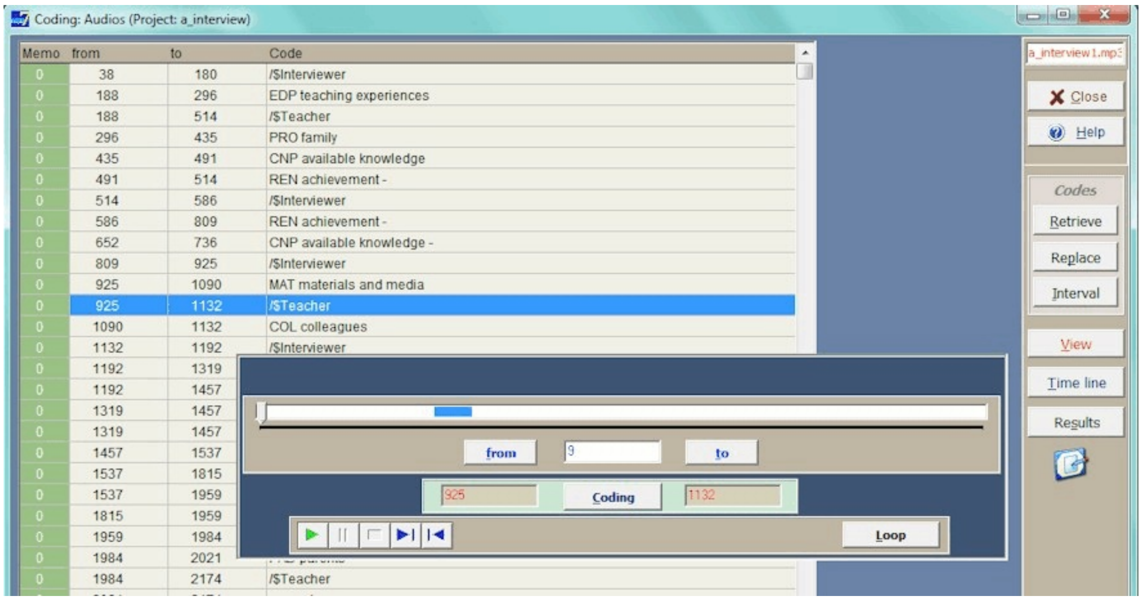
7.1.13 How to code audio data

Managing the "media player"

Windows and functions available for coding audio files (in WAV or MP3 format) correspond to what you know already about coding of text files (see above). However, there is no print-out next to your keyboard, but you have to maneuver through the file with the media player's control board.



On the left side in the screen shot on the following page you see the columns, where your codifications are listed: "M" stands for "memos" (wee see only "0" entries, there are no memos yet), "from" - "to" contain locations of coded segments within the file, and finally we see the column for code names. The unit for numeric entries into the two location columns is 1/10 of a second; internally the media player works with milli-seconds. The resulting values for beginning and end of an audio segment would be very long and clumsy. Therefore, we round the actual settings and show only 1/10 of seconds – accepting tiny aberrations of numerical demarcations and real beginning and end of a data segment. This, however, will go unnoticed in usual interview recordings. The screen shot below shows a highlighted speaker code "/$Teacher" with an assigned segment between "925" and "1132" (tenths of seconds).


Coding: Audios (Project: a_interview)

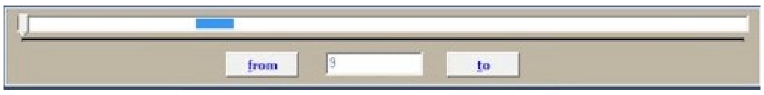
Memo	from	to	Code
0	38	180	/$Interviewer
0	188	296	EDP teaching experiences
0	188	514	/$Teacher
0	296	435	PRO family
0	435	491	CNP available knowledge
0	491	514	REN achievement -
0	514	586	/$Interviewer
0	586	809	REN achievement -
0	652	736	CNP available knowledge -
0	809	925	/$Interviewer
0	925	1090	MAT materials and media
0	925	1132	/$Teacher
0	1090	1132	COL colleagues
0	1132	1192	/$Interviewer
0	1192	1319	
0	1192	1457	
0	1319	1457	
0	1319	1457	
0	1457	1537	
0	1537	1815	
0	1537	1959	
0	1815	1959	
0	1959	1984	
0	1984	2021	[illegible]
0	1984	2174	/$Teacher

When the teacher started to speak in this segment, 92.5 seconds after the beginning of the interview, the button "*Pos1*" (-> Position 1 of the mediavplayer) was pressed. That is what can be seen on the right side. At the end of the teacher's turn, after s/he talks about her colleagues ("COL colleagues"), the button "*Pos2*" was pressed. Both positions – beginning and end – of the data segment within the whole recording are shown in red numbers below the "*Pos*"-buttons. The actual position of the player is visible in blue between these two buttons.

Clicking on the button "*Coding*" between the two slots for numerical information about beginning and end of a segment will open the usual window for entering codes. How to assign a code (or multiple codes) to a data segment corresponds exactly to what you have read about handling text data (see above).

At the very bottom of the panel of coding facilities you see a green bar with the familiar symbols of audio players. You use it to start the recording ("Play"), to pause, to stop, and to jump to the beginning or the end of the recording. Marked data segments will be repeated whenever you click the button "*Loop*", that is the data segment between positions 1 and 2 as shown in red numbers will be played again. Fine tuning of positions is possible with the arrow keys "Up" and "Down" on your keyboard (the "meter reading" in blue numbers will change automatically).

Rapid movement through the recording is achieved by clicking on the pointer in the slide controller on top of the coding panel and pushing it forward or backward with the mouse key still pressed:



If you try to retrieve coded data segments (cf. the corresponding paragraph on coding text files), you click on the appropriate codification in the code column on the left side. AQUAD automatically inserts the border positions of the selected segment into the coding panel to the right. Clicking the button "*Loop*" repeats the content of this segment, and of course, you may interrupt with "*Pause*" or "*Stop*" the reproduction.

We suggest, you start the project example "a_inter" (four fictitious interviews based on the interview transcriptions of the project "Interview") and play with all these possibilities.

A general strategy for coding audio data

In contrast to transcribed texts it is hard if not impossible to skim through the content of an audio file. Unless you are familiar with the content and/or have coded it already, you will need a lot of time and patience to retrieve particular data segments. Therefore we recommend a strategy that proved to be helpful in our own projects:

Start coding (first run) by formally structuring the files, for instance, code the turn-taking of speakers first. In the project "a_inter" we assigned firstly speaker codes (/$Interviewer; /$Teacher) to the data segments spoken by the interviewer and the interviewee.

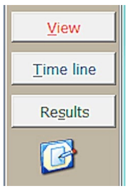
Already during this formal codification we should give careful attention to interesting topics and take them down either in memos or in provisional codes.

As a technical trick we recommend to enter immediately at the beginning of each new data segment an appropriate speaker code – even if we do not know at this point of time where this segment will end. Otherwise the start position of this segment will be lost, if we come upon a thematic issue afterwards for which we want to enter a conceptual code. At the end of the segment then, we would have wasted much time to retrieve the beginning of the segment again. Therefore you should proceed as follows:

- At the beginning of each new segment (next speaker, new question, etc.) you push immediately the stop icon and determine the start position (click "*Pos 1*") and the (preliminary and false) end position (click "*Pos* 2"). If necessary, you may enter or modify the exact position of the beginning of this segment manually (red numbers!) and control its fit by clicking the button "*Loop*." Then you click on "Coding" and insert an appropriate (speaker) code.
- At the end of this segment, that as when the speakers (or questions, topics, etc.) take turns again you learn about its exact position within the audio file: click on the "Stop" icon and read the actual (blue) number. Now we are able to correct the initial error: Clicking on the code name in the matching row on the left opens the small blue window, where you confirm, change or erase codifications. "Change" is what we have to do now: We overwrite the wrong number in the slot labeled "to" with the correct number, which we found (in blue) in the coding panel below. Alternatively you may transfer the number with copy and paste functions (try the right mouse button). Finally, we must not forget to click on the button "*Change*" – this will move the corrected codification to AQUAD's code file.

Maybe the following feedback of an AQUAD user is helpful for you: If you click the buttons "*Pos1*" or "*Pos2*" (the same is valid for "Pause" or "Stop" on the green audio-player panel) and the sound does not stop immediately – intervals of up to 3 seconds between clicking and end of the reproduction were reported – please, do *not* suspect AQUAD to be the cause of your trouble, but the system settings of your computer. The most probable cause, above all in case your computer came with "sound on board," is a mismatch of the hardware's sound components and the installed sound driver. The most convenient solution is to identify the manufacturer of the "sound on board" component (see your computer manual) and download the actual driver version from the manufacturer's internet pages.

Transcribing important data segments



Coding audio recordings directly without firstly transcribing them saves a lot of time, however, there is the disadvantage that it is impossible to copy essential examples from your interviews into your research report. Therefore we recommend to transcribe critical or essential passages (including each data segments border positions; see above) and to save these texts in a separate file.

AQUAD contains a tiny text processor to support this work. You open this text program by clicking on the "paper-and-pencil" icon on the right margin (see screen shot on the right). With this program you may open available (RTF) texts, save new texts (also in RTF), copy and paste texts, select fonts, and so on – familiar functions, which you will need for partial transcriptions "on the fly."

7.1.14 How to code video data

If you are coding video recordings, you can apply everything by analogy what you read in paragraph 7.1.13 about audio recordings. The big difference is, of course, a small video screen above the media-player panel. In case only part of the video is visible in this screen, there was a problem or you missed to adapt the size of your video frames to the dimensions of the screen in AQUAD while converting your data.

However, the code liste and the video screen are to big to appear at the same time next to each other. Therefore, AQUAD inserts for video data two additional buttons at the right margin: "*Show video*" and "*Show codes*." With these buttons you switch between two display modes: Clicking the button "*Show codes*" cuts out the video screen and inserts the list of codifications, while the button "*Show video*" effects the contrary.

The recommendation to start the coding of audio files with formal codifications (see 7.1.13) is valid analogously for video files.

7.2 How to edit texts and codings?

7.2.1 How to edit texts?

This question can be answered quickly: *Within* AQUAD you *never* edit a text! AQUAD is a program dedicated to the analysis of texts; you should refrain from any changes of texts, which form the basis of your analysis, during the process of analyzing them.

However, there may be good reasons not to treat the data base too orthodoxly! Maybe you detect one or another typing error, which you prefer to correct instead of preserving it for the final report. Sometimes you may feel that some additional information would support the interpretation, for instance, some hints of nonverbal behavior in a transcription of a videotaped discussion (well, with AQUAD 7 you will transcribe at the most crucial scenes of a video for quotation in your report). Or you would like to exclude some text segments from further analysis, for instance questions or comments of an interviewer, data describing the speaker, the site, etc., (see chapter 5). In all of these cases you have to edit the text base. This can be done *outside* of AQUAD without any limitations. Please, follow the guidelines for the preparation of texts in chapter 5 and take care to save your texts in ANSI format or in RTF again after editing them in your text processor

If your texts are already coded *and* if you are going to insert or to delete something, you will get a problem, however! By editing your text you have probably changed its length and the indicators of beginning and length of the coded segments will no longer correspond with the modified text. From the position of the first change in your texts on the codes will appear misplaced. Therefore, you should prepare your texts carefully for coding and not change them after you started to attach codes to them. Of course, replacing one or several letters by the same number of different letters will not interfere with your prior work.

7.2.1 How to edit codings?

Remember, codings are composed of code words and associated line numbers (or their equivalents: 1/10 of seconds, frames, pixel coordinates). Maybe there was a typing error during coding, maybe you want to differentiate a very general code like "emotion" later and want to enter instead "joy" or "fear", etc. As was already described in chapter 7.1.5, these codings can be edited in case of texts as data as well as in case of audio, video or graphic data.

- Within all data modes ("*Text*","*Picture*", "*Audio,*" and "*Video*") you can delete codings and attach new ones – code entry by code entry.

- Within all data modes you will find a field labeled "*Codes*" on the right side of the window. In this field are two buttons, which give access to "*Retrieve*" and "*Replace*" functions (see chapter 7.1.5) , which will help you to edit all codings of a particular kind at once.

Please, keep in mind: AQUAD allows only editing of codings attached to data segments, not the content of these data segments! By the way: Most people would not think of "editing" a video recording while analyzing it in AQUAD, so do the same if texts are your data base.

It is important generally to remember that you can check each code entry within each data mode (text, picture, audio, video), if you click on the code name in the column "Codes" of the list of code entries (table on the left side of the screen). This opens a small window, where you confirm, edit, or erase the code entry.

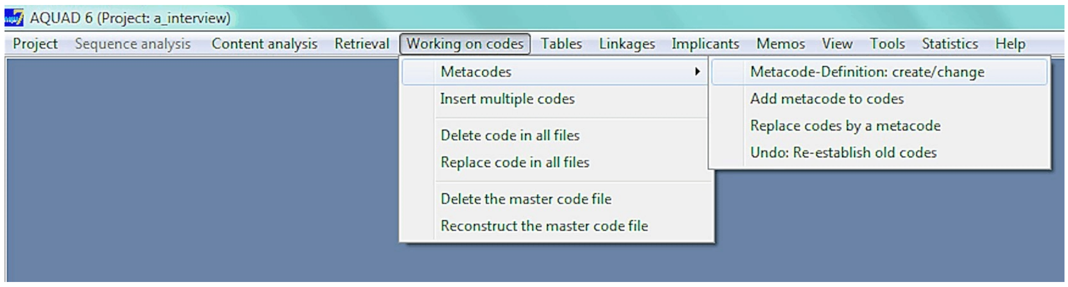
Additionally – in case you are working with texts – a text window opens that allows to compare easily the code entry and the text segment, to which this code was attached. In case you are analyzing audios or videos, you click the button "*Loop*" below in the green operation panel of the audio player or video player and listen to or watch a particular scene of the recording again. In graphic presentations, for instance drawings, clicking on a code name will show a frame around the coded part of the presentation.

7.3 What are meta-codes good for?

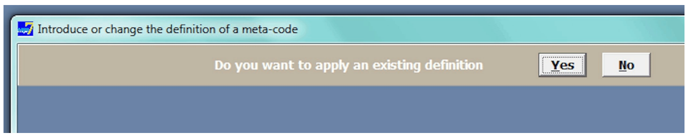
If you apply the strategy of generalization described in chapter 6, you will invent many different codes while reading and interpreting your data for the first time. Soon you will reach a point where you have to compare your various codes and their corresponding data segments to find commonalities and fundamental differences. This point is reached very soon, if you start your work without any particular strategy, but just openness for the meanings hidden in the data. From the very beginning you will find many interesting data segments, and you will mark them with meaningful codes. However, without permanent comparison the danger would be too great to get lost in details and to apply different codes for similar units of meaning in the data files. Therefore, we need to alternate between analysis of details and putting details back into their context – and consequently re-organize our system of codes again and again.

An efficient approach ordering codes is comparing them permanently (cf. the principle of "permanent comparison") and generalizing them tentatively. We check each code, whether it represents similar phenomena as some other codes. If so, we can perhaps understand these other codes as sub-codes, or we can subsume all of them under a new super-ordinate code. This type of super-ordinate codes is addressed in AQUAD as *meta-code.* If you get a hunch that meta-coding may support your overview and structure your work, you should print the code catalog (click on it with the right mouse button!), that is the list of all codes introduced in your work until now (see chapter 7.1.4), and mark all those codes, maybe with different colors, which belong together under some meaningful concept. Then you should invent a name, that is a meta-code for each of these new groups of codes.

After these preparations you choose from the sub-menu "*Working on codes*" the option for creating (or later: modifying) meta-codes: "*Metacodes*" -> "*Metacode-Definition: create/change*".

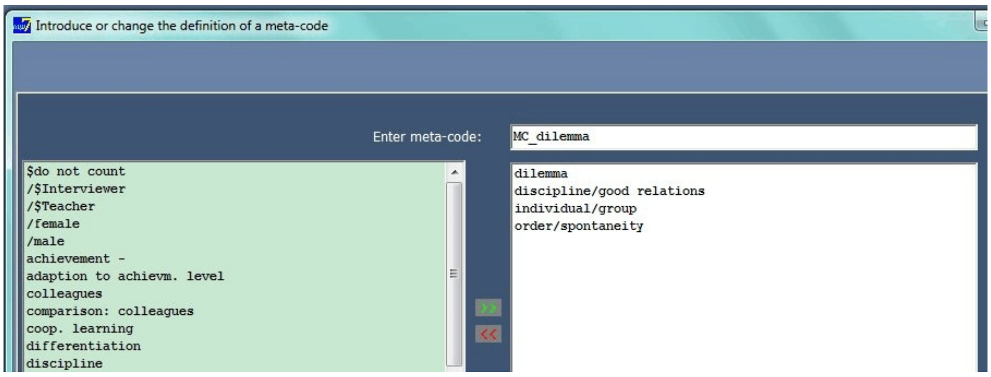

AQUAD 6 (Project: a_interview)
Project
Sequence analysis
Content analysis
Retrieval
Working on codes
Tables
Linkages
Implicants
Memos
View
Tools
Statistics
Help
Metacodes
Insert multiple codes
Delete code in all files
Replace code in all files
Delete the master code file
Reconstruct the master code file
Metacode-Definition: create/change
Add metacode to codes
Replace codes by a metacode
Undo: Re-establish old codes

Clicking on this function opens the following window, where you are asked whether you want to apply an existing definition (to add or delete some sub-codes from the definition). Since we want to create a new meta-code, the answer is "No":


Introduce or change the definition of a meta-code
Do you want to apply an existing definition
Yes
No

After answering the question the actual window for a meta-code definition is shown. As regards the slot labeled "Enter meta-code" you do exactly that: Write a new code name – a ***meta-code*** – into this slot. Later this new code will replace some existing codes or be added to their data segments. Then you select these codes by clicking on these codes in the light-blue box on the left (master code list). Of course, you can undo any selection by clicking on a code in the white selection box. Before any selection will be accepted, that is the selected code will be transported from the master-code file left into the meta-code box on the right side, you have to enter the name of the new meta-code.

Here we introduced the code-name "MC_dilemma" (MC for meta-code) in order to unite the teachers' specific dilemmas whether discipline vs. good relations, individual work vs. group work and order vs. spontaneity should be preferred in the classroom. In some cases the general code "dilemma" was used without differentiating its nature, therefore it is also included in the definition:

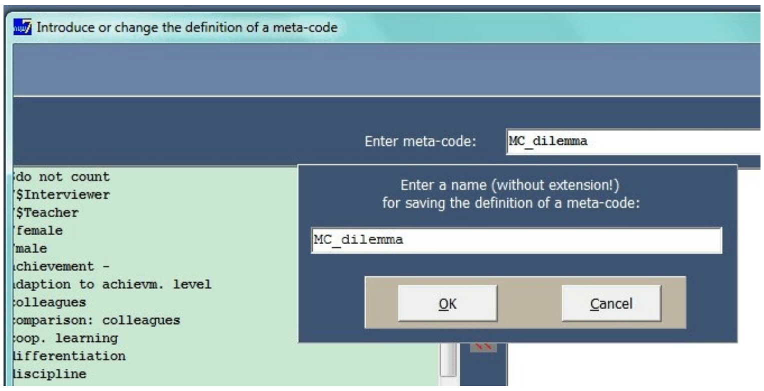


```
MC_dilemma
```

```
$do not count
/$Interviewer
/$Teacher
/female
/male
achievement -
adaption to achievm. level
colleagues
comparison: colleagues
coop. learning
differentiation
discipline
```

```
dilemma
discipline/good relations
individual/group
order/spontaneity
```

After clicking on the button "*OK*" on the right margin you are asked (see next figure) to write a file name (without extension!) for the meta-code definition. It is a good idea to insert the meta-code or an abbreviation (in case of longer codes) as file name for saving the definition.


Introduce or change the definition of a meta-code
Enter meta-code:
MC_dilemma
Enter a name (without extension!)
for saving the definition of a meta-code:
MC_dilemma
OK
Cancel

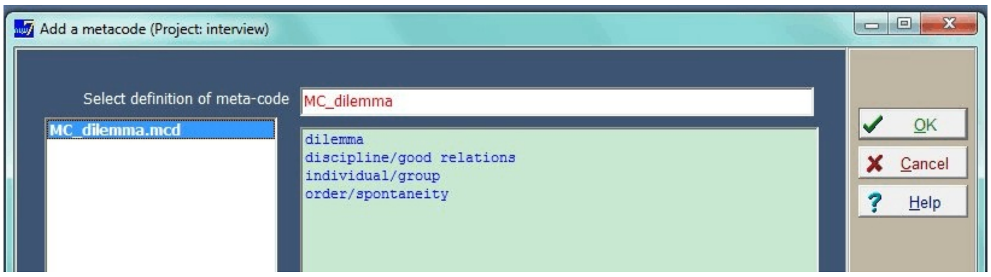
```
do not count
$Interviewer
$Teacher
female
male
chievement -
daption to achievm. level
olleagues
omparison: colleagues
oop. learning
ifferentiation
iscipline
```

After clicking on the button "*OK*" the definition of your new meta-code will be saved. That is, AQUAD will create a file containing the new code name and the list of sub-ordinate old codes for later application.

7.3.1 Modify code files: Add meta-codes

At this point you should be familiar with the difference between *replacing* existing codes by a newly introduced meta-code and *adding* a meta-code to your existing codes. To add a meta-code, you select in the group "*Working on codes*" the option "*Metacodes*", then "*Add metacode to codes.*"

In the box on the left we see the list of available metacode definitions (compare the extension: "mcd"). Supposing we want to apply the newly saved definition "MC_dilemma.mcd", we click on this name – and the file name appears in read on top of the blue box, were the content is shown, that is, the list of sub-ordinate codes.


Add a metacode (Project: interview)
Select definition of meta-code
MC_dilemma.mcd
MC_dilemma
OK
Cancel
Help

```
dilemma
discipline/good relations
individual/group
order/spontaneity
```

After clicking on the "*OK*" button, the new meta-code is *added* to each of the old, grouped codes. Thus, your original codes are not lost. However, your code files get longer and longer, particularly if you play with the possibility of generalizing by introducing meta-codes – what you should try to do! Therefore, if you intend to use this option frequently, you better apply the function "*Replace codes by a metacode*", which also keeps care that your original code files are saved for subsequent analyses.

7.3.2 Modify files: replace codes by meta-codes

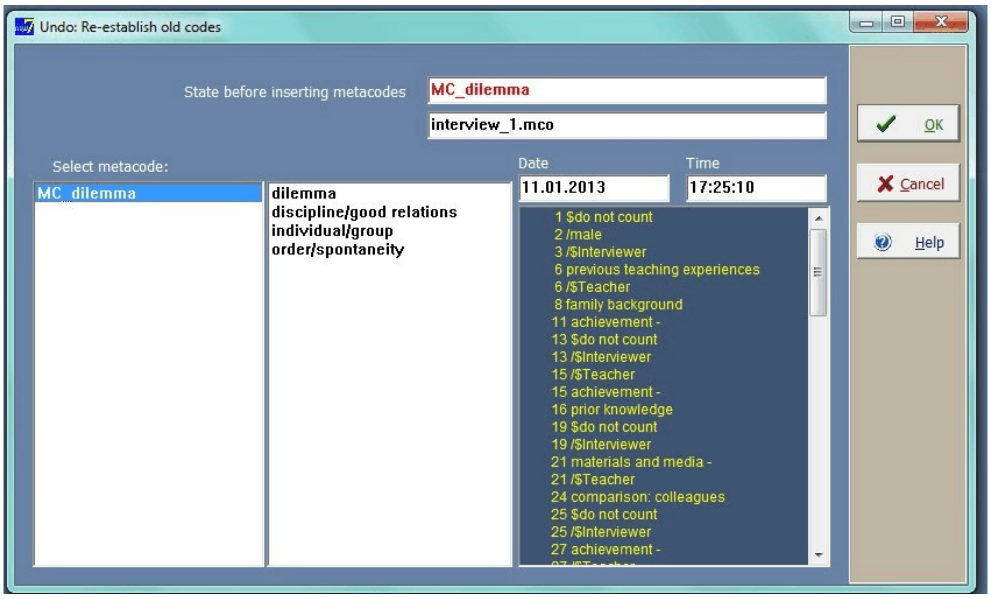
The procedure of selecting a particular meta-code is exactly the same as shown above in paragraph 7.3.1. However, after clicking on the "*OK*" button, the sub-ordinate codes disappear from the code files and listings, and are *replaced* by the meta-code you selected. Don't worry: Your old code files are not erased or overwritten, but saved separately in the directory ..\mco\ together with information about date and time of their construction. Thus your initial codes are not touched and may be retrieved for later use.

Remember: AQUAD stores only up to 100 modifications of your initial code files. From modification no. 101 on already stored files are overwritten, beginning with the oldest file (*-001.cod). AQUAD will show a warning in this case, so you can decide about saving all your previous code files manually in a different directory.

7.3.3 Reconstruct previous codings

If you decide to return to a former state of codification, select in the menu-group "*Working on codes*" -> "*Metacodes*" the option "*Undo: Re-establish old codes:*" (see next page).

You click in the box on the left on the name of a meta-code, which you want to undo in your code files. If you applied more than one version of the same meta-code, there will be a list if identical meta-code names. To make sure you find the meta-code you are looking for, the box in the middle shows the components (sub-ordinate codes) of the selected meta-code. (In the example we have only one meta-code definition to select ...), and in the blue box on the right you find the beginning of the code file attached to the first data file (here: "interview_1") in your project (here: "interview"). Above this box you see date and hour, when you replaced codes in the then actual code files by the selected meta-code. Now you may restore the former state by clicking on "*OK*."



State before inserting metacodes: MC_dilemma

interview_1.mco

Select metacode:
MC_dilemma

dilemma
discipline/good relations
individual/group
order/spontaneity

Date: 11.01.2013
Time: 17:25:10

```
 1 $do not count
 2 /male
 3 /$Interviewer
 6 previous teaching experiences
 6 /$Teacher
 8 family background
11 achievement -
13 $do not count
13 /$Interviewer
15 /$Teacher
15 achievement -
16 prior knowledge
19 $do not count
19 /$Interviewer
21 materials and media -
21 /$Teacher
24 comparison: colleagues
25 $do not count
25 /$Interviewer
27 achievement -
```

It is a good idea to use the memo function, when you apply meta-codes. Particularly you should create a sort of research diary and note exactly, at what day and time you implemented which meta-code in your system of codification. Memos of this kind will make it easy to retrieve the critical state of coding among up to 100 different states of modification.

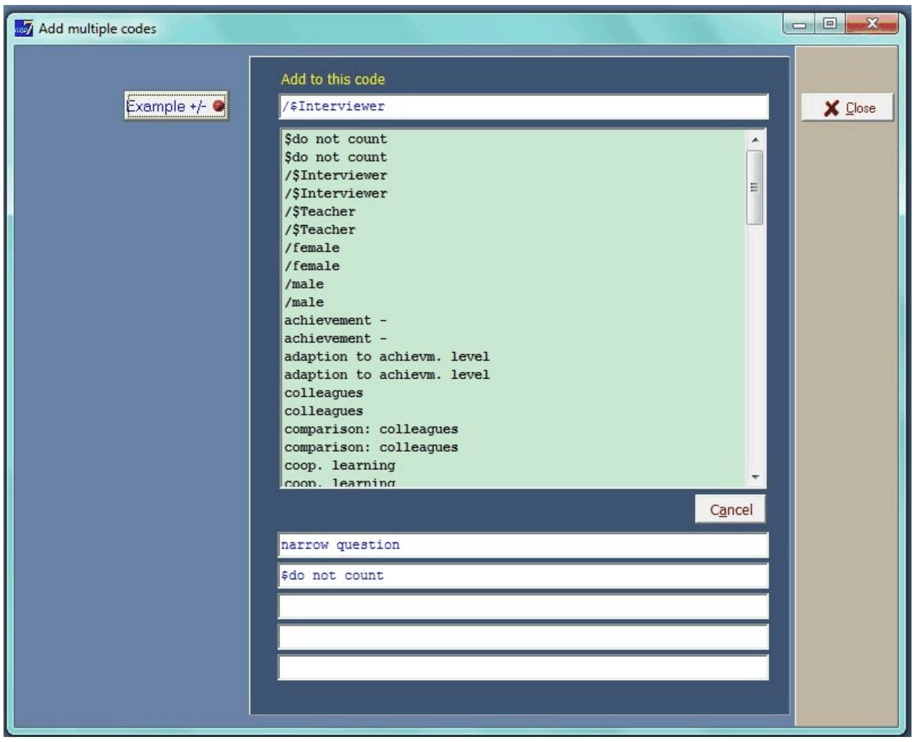
7.4 How to enter multiple codes?

During coding you may add up to five codes at the same time to a selected data segment (marked lines in text, a scene in a video, etc.). This is very helpful, for instance, if you want to code simultaneously the parts of a particular speaker with a speaker code (e.g., /$Interviewer), with three different conceptual codes, and exclude the content from counting on the level of words ($do not count).

However, sometimes you are not aware from the very beginning of coding that particular data segments need multiple codes, but only later you are confronted with the necessity to add in many files to many segments of a particular code some additional codes. Of course, the retrieval functions for codes would give rapid access to all critical data segments – but then you would have to enter the codes manually and start the function for entering codes again and again. A more convenient possibility would be to apply the function "*Add metacode to codes*" (see above) creatively to insert automatically additional codes at defined locations within the data files. From a point of view of conceptual clarity we preferred to add a specific function "*Insert multiple codes,*" which does exactly and straightforwardly what we just described.

First we select (double click) from the master code list (light blue window) a code to which we want to add multiple codes. That is precisely, these multiple codes are added to the data segments already coded with the selected, "leading" code. Then we enter the additional codes by selecting or writing them into the entry slots at the bottom of the window. Clicking on the green button "*Insert code/s*" starts the operation.

This button appears to the left of the button "*Cancel*," but is not visible at the moment in the screen shot, because we activated the "*Example* +/-" function: Clicking on this button inserts an example of leading code ("/$Interviewer") and multiple codes ("narrow question", "$do not count"); clicking a second time clears the entry slots again for your particular codes.



```
/$Interviewer
```

```
$do not count
$do not count
/$Interviewer
/$Interviewer
/$Teacher
/$Teacher
/female
/female
/male
/male
achievement -
achievement -
adaption to achievm. level
adaption to achievm. level
colleagues
colleagues
comparison: colleagues
comparison: colleagues
coop. learning
coop. learning
```

```
narrow question
$do not count
```

7.5 How to use keywords?

7.5.1 Looking for words in data texts

One possibility was already explained above (see chap. 7.1.9). Remember the button "*Retrieve*" in the box "*Keyword*" at the right margin of the window for coding "*Text*". While reading a text file, you may activate a retrieval of keywords.

You click on the button "*Retrieve*" and a particular blue window pops. Here you enter the critical word or sequence of words. You start the retrieval by pushing the "*Retrieve*" button in this window, then proceeding with the "*continue*" button from location to location where the key word(s) are hidden in your text. Or you click on the "*all*" button, which displays the results of a retrieval run in one glimpse. Let us assume we go step by step ("*Retrieve*" and repeatedly "*continue*"): In case the word is found, the text display changes so that the line containing the critical word is highlighted. Maybe you are wondering about the result, because there are some specific rules to remember:

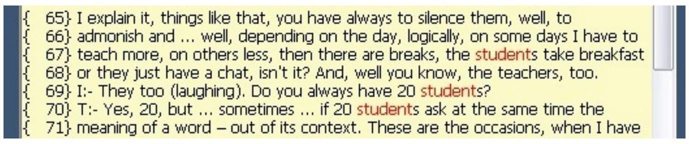
- The search is case-sensitive, that is, AQUAD differentiates between capital letters and small letters when retrieving key words in your data text (as opposed to your memos!). If you try to find, for example, the word friend, the program will not find "Friend" at the beginning of a sentence (and vice versa).
- The retrieval algorithm is unable to put hyphenated words together again. If you expect that you will use word searches frequently in your analyses, you should switch off any automatic hyphenation when transcribing the texts.
- If you choose the setting "*part of words*," then AQUAD runs a "wild card" search, locating your string of characters wherever it occurs, including embedded in other words. If you try to find, for example, a string "he was", the program will also stop and highlight these words as part of "...the wasting of ..." or . If you want to avoid this effect, you should keep the default setting to "*whole words*."

This last feature deserves some comments. Let us assume you want to retrieve all text segments where something is said about "advantage" or "disadvantage." If you enter simply "dvantage" as criterion for a word search (with the setting "*part of words*"), AQUAD will find everything you are looking for in one run – given that the words really appear in your texts. Or another problem: You are searching for "friend" and "friends," but you do not want to have reported segments where something is characterized as "friendly" or as "friendship." So add a blank after the critical word (represented by "_" in the following examples) and have AQUAD retrieve text segments containing "friend_" and then in a second run "friends_". Now you will find only these complete words, however, not if they appear at the end of a sentence, followed by a period instead of a blank! Therefore, you should use the opportunity to create a whole retrieval dictionary, that is a word catalog (more about this in the next section).

7.5.2 Looking for keywords in context

For some research tasks it may be helpful to get a fast overview of words in the data that you assume are indicators of critical concepts. In fact, you might even want to use this feature to begin defining your codes. It might be important to know in which data file these words appear and in which context. Word frequencies alone may be insufficient, because often the specific meaning of a word can only be understood from its context. Imagine, you have counted the word "high", but the context of *high* may be in one case "During this time I always felt at high tension ... ," and in another case "After my friend left, I really was high and dry." More information about this procedure (key-word-in-context or "KWIC" lists) can be found in Weber (1985).

Looking for keywords in context is activated from the main menue: "*Retrieval*" -> "*Keywords*". The example below shows the result of a "KWIC" search for the word "student" in the example "interview_1". Wherever AQUAD retrieves the key word, whether as whole word or as a part of a word, for instance "students", it will mark its finding in red color:



```
{   65} I explain it, things like that, you have always to silence them, well, to
{   66} admonish and ... well, depending on the day, logically, on some days I have to
{   67} teach more, on others less, then there are breaks, the students take breakfast
{   68} or they just have a chat, isn't it? And, well you know, the teachers, too.
{   69} I:- They too (laughing). Do you always have 20 students?
{   70} T:- Yes, 20, but ... sometimes ... if 20 students ask at the same time the
{   71} meaning of a word – out of its context. These are the occasions, when I have
```

The function starts by opening a window where you can choose between

- entering key words by typing them *or*
- selecting the name of a (key-) word catalog from a list of available catalogs (in case there are any!).

If you decide to type your keyword(s), these words have to be typed anew whenever you activate this function. If you are sure to use a list of words for several retrieval runs, you should create a word catalog.

To do so you choose within the option "*Content analysis*" -> "*QUANtitative analysis*" -> "*Word catalog*" the option "*write a word catalog*." Then you define a name for saving the catalog you are going to write; the extension ".cwo" (catalog of words) is added automatically. Then a window is opened into which you can write as many key words as you find useful for your retrieval run. Please, remember to put only one word or one sequence of related words on a line. You should be already familiar with creating catalogs from writing your first file catalog (see chapter 5).

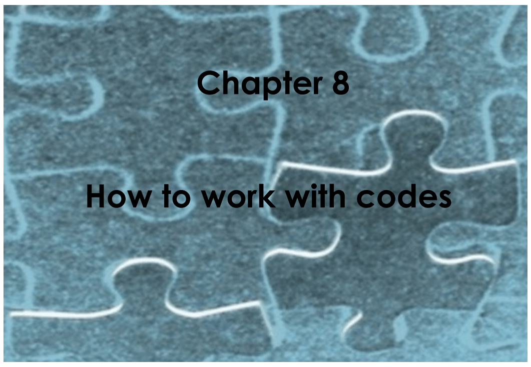
In order to find, for example, the independent word "old" at the beginning of sentences, at the end of sentences, somewhere within sentences, but not as part of other words (for instance, "gold"), a word catalog should contain at least the following entries. In this catalog, the underlining characters "_" represent spaces in front of and after the key word; do not type this character instead of a blank (using the space bar on your keyboard) when creating a catalog!

old
Old_
_old,
_old;
_old.
_old!
_old?

Besides, the rules already mentioned in section 7.5.1 are valid here, too. Remember above all, that the search is case-sensitive! Once you have created a word catalog, you can use it again and again by simply selecting its name via the option "*Load a word catalog*".

A particularly interesting possibility to apply word catalogs should be mentioned: You can use a word catalog to define a dictionary of synonyms or a complete list of all concepts defining a particular field of meaning. Thus, you get an opportunity for semantic retrieval of critical text segments. Let us assume you want to find all text segments where your interview partners made remarks about their social relations. You could type a word catalog containing all relevant elements of this field of meaning, for instance, friend(s), acquaintance, family, mother, father, brother(s), sister(s), partner, wife, husband, etc.

Beyond just retrieving important text segments without reading all the texts again, you may apply your findings in a time-saving strategy: Why not use them for *semi-automatic coding* (see also section 7.1.9)? Mark the unit of meaning that contains your keyword by clicking into the text and highlight the complete relevant data segment – at the same time you will get the window for code entries on the screen additionally. If you are not yet familiar with entering codes, read more about how to enter codes in section 7.1.4.



Chapter 8

How to work with codes

8.1 How to retrieve codes

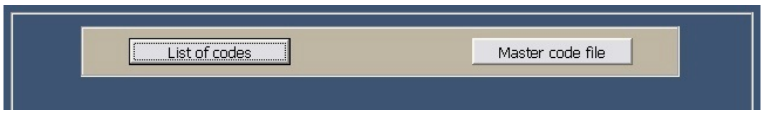
8.1.1 Retrieving coded text segments and particular codes

There are many good reasons why a researcher now and then may want to get an overview on all those text segments to which the same code was attached. One of these reasons, which is prominent in descriptive/ interpretative studies (see Tesch, 1990), is to check commonalities across all data texts. Another reason is to control the consistency of coding. All codes should be supplied always according to the same principles. Still other reasons are to look for locations in the texts where a code was applied correctly vs. erroneously, where similar meanings can be found expressed in different formulations, etc.

AQUAD extracts coded text segments from all texts following the sequence in which they are listed in the file catalog. Because one text after the other is checked sequentially, we call this search for matching text segments a linear or one-dimensional analysis. It is to be distinguished from two-dimensional analyses, called table analyses or matrix analyses (see below, section 8.2.2) in AQUAD, and from analyses of complex linkages, also called exploration of linkages (see below, section 8.3). The results can be brought to the screen, to a printer or they can be stored on the hard disk.

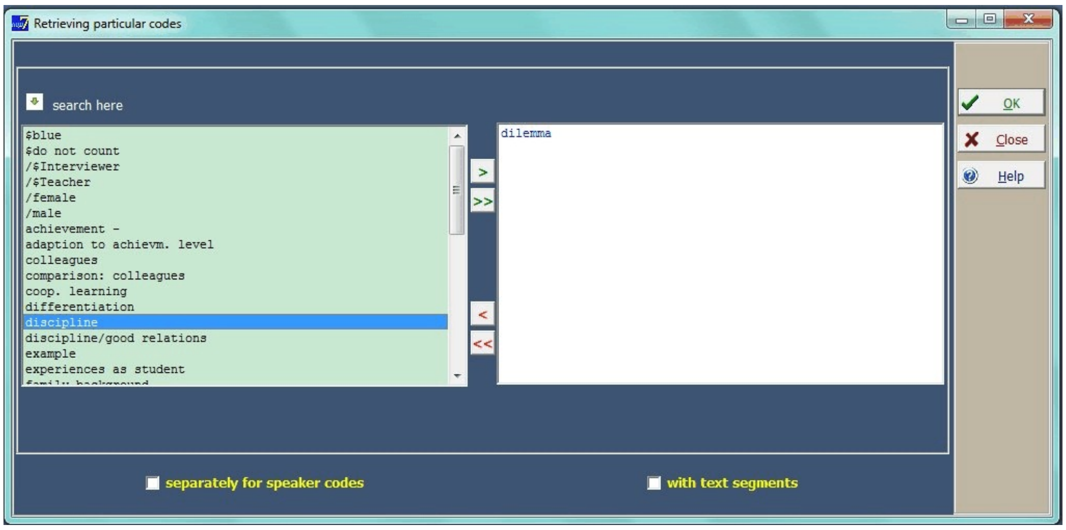
You start a linear search for coded text segments by choosing the option "*Particular codes*" within the module "*Retrieval*." A window opens from which you can decide

- to load one of your already existing code catalogs (i.e., a list of codes) *or*
- to construct a temporary list of codes by selecting the content from the master code file.



If you select critical codes from the master code list, the codes have to be selected anew whenever you activate this function. In the following example (see screen shot on top of the next page) we select the code "dilemma" from the master code list. Please, pay attention to three check-boxes labeled in red at the bottom of the selection window and mark those options you wish to apply:

- "separately for speaker codes:" The code/s will be searched for and listed within each datafile separately for different speakers – of course, only if you marked their data segments by speaker codes.
- "show text segments" attaches the coded text segment directly to each retrieved code when the results are presented.



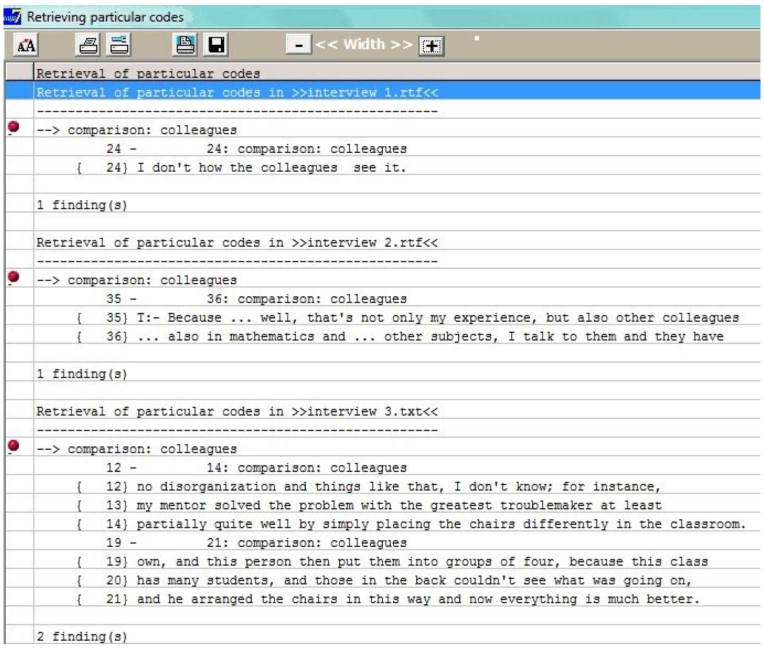
```
$blue
$do not count
/$Interviewer
/$Teacher
/female
/male
achievement -
adaption to achievm. level
colleagues
comparison: colleagues
coop. learning
differentiation
discipline
discipline/good relations
example
experiences as student
```

```
dilemma
```

If you are sure to use a list of codes for several retrieval runs, you should create a code catalog. To do so you choose – also within the option "*Retrieval*" – from "*Code catalog*" the option "*Create a code catalog.*" Then you define a name for saving the catalog you are going to construct. Just enter a name; the extension ".cco" (catalog of codes) is added automatically. Then a window is opened, where you can create the catalog by selecting critical codes from the master code list. Meanwhile, you should be already familiar with creating catalogs from writing your first file catalog (see chapter 5).

As an example we start a search for all text segments coded as "comparison: colleagues" in our sample texts interview_1, ..., interview_4. The screen shot on the right shows only part of the results.

We will not discuss these findings here, but pay attention to its form: You see that AQUAD shows in which texts (interview_1 through interview_4) it tried to find which the code "comparison: colleagues," and that matching text segments are shown together with their line numbers. In case you should want to include these findings later in your research report, you should not forget to click on the diskette icon in the tool bar on top of the window, where the results of the retrieval run are shown.

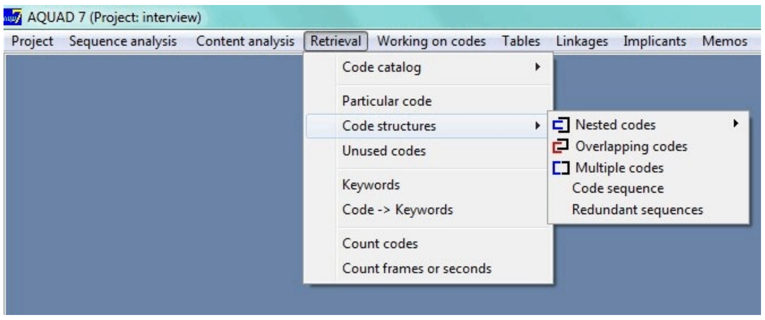


```
Retrieval of particular codes
Retrieval of particular codes in >>interview 1.rtf<<
-----------------------------------------------------
--> comparison: colleagues
         24 -          24: comparison: colleagues
     {   24} I don't how the colleagues  see it.

1 finding(s)

Retrieval of particular codes in >>interview 2.rtf<<
-----------------------------------------------------
--> comparison: colleagues
         35 -          36: comparison: colleagues
     {   35} T:- Because ... well, that's not only my experience, but also other colleagues
     {   36} ... also in mathematics and ... other subjects, I talk to them and they have

1 finding(s)

Retrieval of particular codes in >>interview 3.txt<<
-----------------------------------------------------
--> comparison: colleagues
         12 -          14: comparison: colleagues
     {   12} no disorganization and things like that, I don't know; for instance,
     {   13} my mentor solved the problem with the greatest troublemaker at least
     {   14} partially quite well by simply placing the chairs differently in the classroom.
         19 -          21: comparison: colleagues
     {   19} own, and this person then put them into groups of four, because this class
     {   20} has many students, and those in the back couldn't see what was going on,
     {   21} and he arranged the chairs in this way and now everything is much better.

2 finding(s)
```

AQUAD supports permanent comparison of your analytic decisions not only by text retrieval, but by various options for retrieval of codings. We have just seen the first of these options: If you run a search for "*Particular codes*," the program informs you both about the location of text segments to which a particular code was attached and about these segments.

8.1.2 How to retrieve specific structures of coding

Let us assume a researcher has some doubts about several text segments and is not sure which category to apply for coding. Therefore, he or she attached all those codes that appeared to be meaningful to each of these text segments. Now imagine further that in another text two more categories could be applied to a critical text segment. Whenever in doubt, this researcher makes use of AQUAD's possibilities of multiple coding. At some point the researcher gains the insight necessary to clarify the ambiguous codings. It would be very useful now to retrieve all critical text segments quickly, that is without searching for single codes and pick those segments from the numerous findings that were attributed to other categories, too. A problem like that could be easily solved by running a search for coding structures. The researcher would choose "*Retrieval*" from the main menu, select the option "*Coding structures*" and activate "*Multiple codes*" from the next sub-menu.



This sub-menu, appearing after the selection of "*Coding structures*" offers a choice between several abstract structures of coding. After determining *additionally,* which codes you are trying to retrieve in a particular pattern, you can make AQUAD search for

- nested codes,
- overlapping codes,
- multiple codes,
- sequence of codes, and
- redundant (i.e., repeated) sequences

in your code files. Here is in more detail what AQUAD will do:

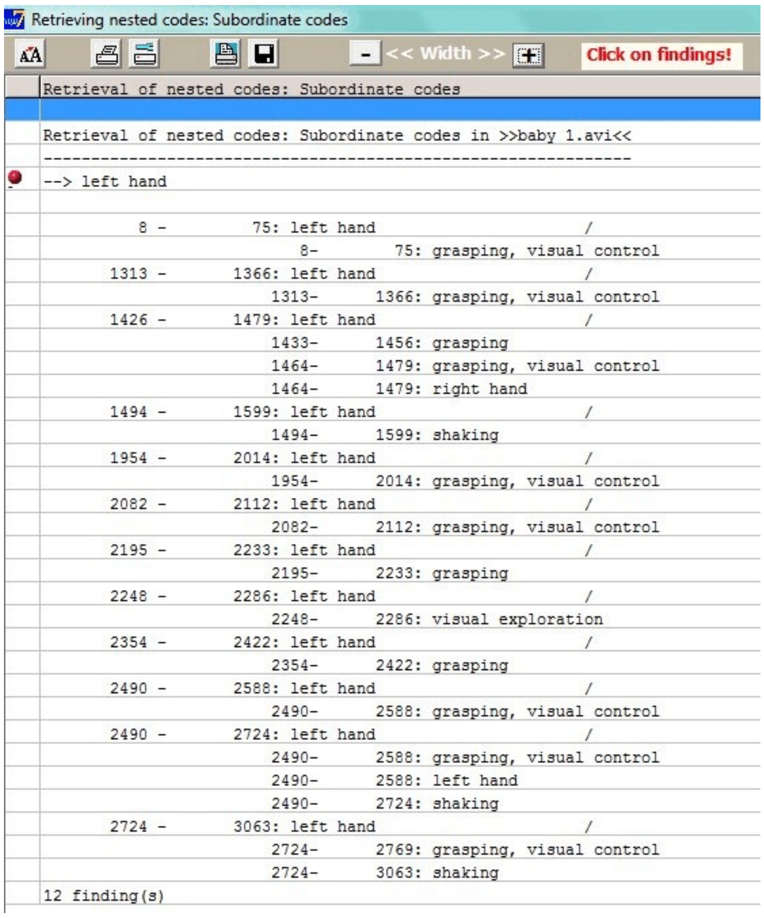
Nested codes

With this option the search is aimed for data segments which contain another data segment within its boundaries. Or formulated differently: all those codings are found, which include within their text area another coding. This option therefore is useful, if you are looking for hierarchically structured sequences of codes. You will not only find codes and sub-codes, but also codings with identical line numbers, as long as their text areas do not overlap (see below).

Example: In our video file "baby_1" some segments were coded as related to "left hand". Later the coders began to attend to more specific processes like grasping, oral exploration, etc. Now we want to merge the broader and the narrower perspectives. A search could show, for instance, that particular video scenes were coded as "left hand", while within this segment the code "grasping" was attached to a smaller part of the scene.

AQUAD will report this finding as shown in the screen shot on the next page. We see, for instance: Within the super-ordinate segment from frame 1426-1479, grasping was coded and additionally involving the right hand (between the frames 1464-1479). Also between the frames 1464 and 1479 "visual control" of the movement was noticed.

By the way, if you click on one of the findings, the coded scene will be shown in the video window – as signaled in red at the top margin.

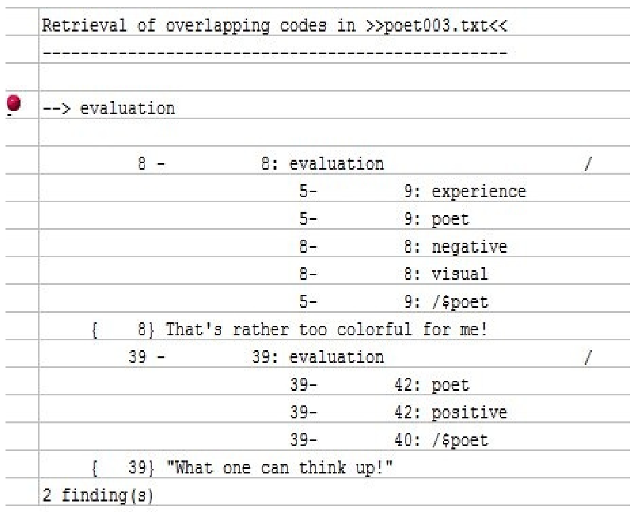


```
Retrieval of nested codes: Subordinate codes

Retrieval of nested codes: Subordinate codes in >>baby 1.avi<<
---------------------------------------------------------------
--> left hand

         8 -         75: left hand                   /
                         8-        75: grasping, visual control
      1313 -       1366: left hand                   /
                      1313-      1366: grasping, visual control
      1426 -       1479: left hand                   /
                      1433-      1456: grasping
                      1464-      1479: grasping, visual control
                      1464-      1479: right hand
      1494 -       1599: left hand                   /
                      1494-      1599: shaking
      1954 -       2014: left hand                   /
                      1954-      2014: grasping, visual control
      2082 -       2112: left hand                   /
                      2082-      2112: grasping, visual control
      2195 -       2233: left hand                   /
                      2195-      2233: grasping
      2248 -       2286: left hand                   /
                      2248-      2286: visual exploration
      2354 -       2422: left hand                   /
                      2354-      2422: grasping
      2490 -       2588: left hand                   /
                      2490-      2588: grasping, visual control
      2490 -       2724: left hand                   /
                      2490-      2588: grasping, visual control
                      2490-      2588: left hand
                      2490-      2724: shaking
      2724 -       3063: left hand                   /
                      2724-      2769: grasping, visual control
                      2724-      3063: shaking
12 finding(s)
```

AQUAD is programed to analyze nested codes in two directions:

- You determine a code (or several codes in a code catalog) as superordinated code and AQUAD will retrieve all sub-ordinated codes within the same data segment (as shown in the example here).
- You determine a sub-ordinated code (or several codes in a code catalog) and AQUAD will retrieve all codes attached to data segments that cover the data segments of this sub-ordinate code.

Overlapping codes



```
Retrieval of overlapping codes in >>poet003.txt<<
-------------------------------------------------

--> evaluation

          8 -          8: evaluation                    /
                            5-        9: experience
                            5-        9: poet
                            8-        8: negative
                            8-        8: visual
                            5-        9: /$poet
      {    8} That's rather too colorful for me!
         39 -         39: evaluation                    /
                           39-       42: poet
                           39-       42: positive
                           39-       40: /$poet
      {   39} "What one can think up!"
2 finding(s)
```

The result of this search strategy shows all codes (as always together with their location within a text), which were attached to overlapping text segments. That is, we are informed about both text segments which overlap, not only about the overlapping area.

Example: We take our sample texts "Poet" and ask for text segments overlapping all text segments which were coded as "evaluation." We see in the screen shot below only part of the findings, and only from the data file poet003. The first finding of an evaluative remark ("That's rather too colorful for me") appears in line 8 of the text as part of the poet's (speaker code "/$poet") utterances in lines 5-9 and was also coded as a "negative" "visual" "experience."

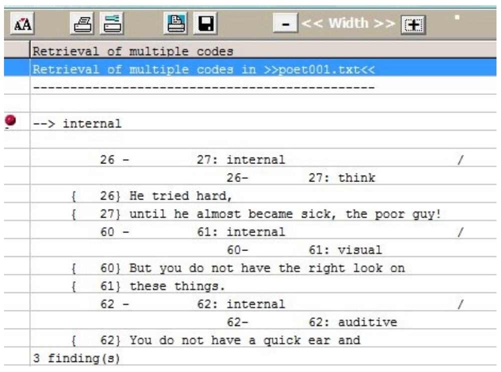
Please, keep in mind: "Overlapping" is defined merely in terms of physical overlap of text areas, without any semantic connotations.

The result shows not only codifications, but also the coded data segments, because the check box "show text segments" was marked in the selection window. But you should be careful with this option: If you try to retrieve some very comprehensive codes overlapping with many other units of meaning, we would get very long print-outs! Per definition, speaker codes overarch everything within an actual data segment of this speaker!

Multiple codes

This option was already described at the beginning of this chapter. It is used to retrieve all data segments to which more than one code was attached.

Example: We assume now that the researcher was not sure whether to code some data segments in the project "poet" generally as dealing with "internal" processes or more specifically as thinking, reflecting, etc. In this situation the researcher decided to apply both categories and to check their validity later. Searching for "multiple codes" including "internal" will produce the following result (here shown only for the data text "poet001" – "With text segments" was marked again):

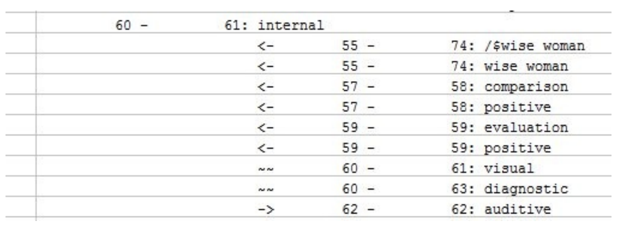


```
Retrieval of multiple codes
Retrieval of multiple codes in >>poet001.txt<<
---------------------------------------------

--> internal

         26 -         27: internal                    /
                           26-        27: think
      {   26} He tried hard,
      {   27} until he almost became sick, the poor guy!
         60 -         61: internal                    /
                           60-        61: visual
      {   60} But you do not have the right look on
      {   61} these things.
         62 -         62: internal                    /
                           62-        62: auditive
      {   62} You do not have a quick ear and
3 finding(s)
```

Sequence of codes

A very useful first step to approach the reconstruction of linked meanings in a data file is to look for statements, scenes, etc. frequently found in the neighborhood of other units of meaning. We specify a critical code as focus of a search for "sequences of codes." AQUAD will report all combinations of codings used for all data segments which are within a maximal distance – which we determine additionally – of our focus code. "Nested codes" and "overlapping codes" will be reported, too.

Example: We want to find out quickly, which other codes were used in the project "poet" in the neighborhood of text segments, to which the code "internal" was attached. As maximal distance we determine 3 lines before or after the critical text segments. That is, AQUAD will report all codes attached to text segments that end within maximally 3 lines before a text segment coded as "internal" begins, and all codes attached to text segments that start within maximally 3 lines after the last line of this critical segment (here you see only part of a very large result file):



```
     60 -        61: internal
                      <-        55 -        74: /$wise woman
                      <-        55 -        74: wise woman
                      <-        57 -        58: comparison
                      <-        57 -        58: positive
                      <-        59 -        59: evaluation
                      <-        59 -        59: positive
                      ~~        60 -        61: visual
                      ~~        60 -        63: diagnostic
                      ->        62 -        62: auditive
```

The characters in front of the line numbers indicate units of meaning before (<-), after (->) and within (~~) the critical text segment.

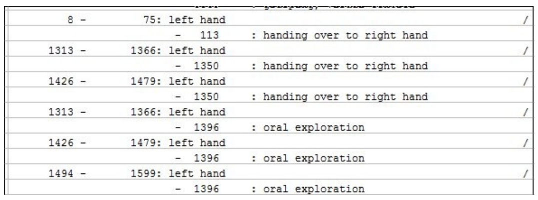
Repetition of sequences

The main function of the qualitative analysis in AQUAD is to support the endeavors of researchers to find and check relations of meanings in their data files. The program does this with algorithms which apply the principle of "backward deduction." That is, AQUAD takes your assumptions about how codes could be associated and checks all data files to find whether the critical codes can be found in matching sequences (and distances).

Each coded data segment can be conceived of as part of a proof for the corresponding category in your data. While reading your texts, structuring them, comparing them and having now and then a look into your memos you will soon start to develop assumptions about relations between some of the categories used in your analysis. Emerging relations or associations may give rise to the formulation of hypotheses about latent contents of your data.

The retrieval of *redundant (or repeated) sequences* is a heuristic approach supporting the detection of associations. You enter a code (or several codes) together with the units of measurement (number of lines, seconds, frames) determining a particular area data around its data segments. Whenever a data segment coded with a particular other code name appears *more than once* within this area, AQUAD will report it.

Example: We want to find out quickly, whether there are redundant structures including data segments in the project "baby," to which the code "left hand" was attached. As maximal distance we determine 3x25 =75 frames before or after the critical data segments. That is, AQUAD will report all codes attached *more often than once* to data segments that end maximally within 75 frames before a video scene coded as "left hand" begins, and all codes attached to data segments that start within maximally 75 frames after the last frame of this critical segment (here you see only part of a large result file):



```
   8 -      75: left hand                                    /
                 -   113    : handing over to right hand
1313 -    1366: left hand                                    /
                 -  1350    : handing over to right hand
1426 -    1479: left hand                                    /
                 -  1350    : handing over to right hand
1313 -    1366: left hand                                    /
                 -  1396    : oral exploration
1426 -    1479: left hand                                    /
                 -  1396    : oral exploration
1494 -    1599: left hand                                    /
                 -  1396    : oral exploration
```

8.1.3 Digression: Distance between data segments

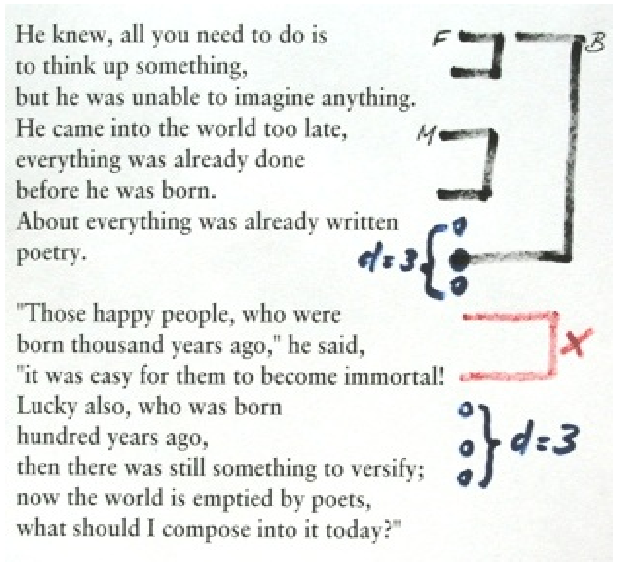
Hypotheses about linkages of codes (or more precisely: linkages of coded data segments) make sense in most cases only if we include a formulation like "... under the condition of a specific maximal distance between them." Take, for instance, the example of the project "interview" (on your hard disk) and suppose we got a hunch that some speakers (teachers) begin to reflect about themselves in these interviews after they talked about a typical dilemma in their classrooms. In terms of a linkage hypothesis: We observe that a speaker talks in a particular data segment about a "dilemma" – and now we expect that he/she talks in the following data segment about himself/herself (coded as "SIM self" in the example).

Since the program is unable to analyze semantically, whether there is a relation of meaning or not, we have to rely as approximative solution to limit the distance between critical segments in the stream of words. Most probably there is no direct relation between two instances of our critical codes, if a teacher talks about conflicting tendencies of reacting to students, for instance, allowing space for spontaneity vs. insisting strictly on rules, and ten minutes (or four pages in the transcription) later he/she talks about being somewhat frustrated. If there is a direct relation, most speakers will refer explicitly again to an idea (in our example: a cause) they expressed already earlier. We would have to code this – maybe very short – reference once more as "dilemma." The default distance is set to three "units", that is in case of text analysis three lines as maximal distance. Data segments "SIM self" beginning only four lines after the end of a segment coded as "dilemma" would not be registered as positive findings in our example. What follows is that we have to experiment with the distance settings and adapt it to the characteristics of our data.

The following screen shot uses an example from "two-step coding" the files of the project "poet" (was copied to your harddisk during installation): The data segment

```
He knew, all you need to do is
to think up something,
     ...
About everything was already written
poetry.
```

was coded as "B." After a blank line follows a data segment "Those happy people ..." coded as "X." The default setting d=3 (maximal distance = 3 lines) would report a positive result for a hypothetical linkage of B-X, however a linkage of F-X would not be reported in this case.



He knew, all you need to do is
to think up something,
but he was unable to imagine anything.
He came into the world too late,
everything was already done
before he was born.
About everything was already written
poetry.

"Those happy people, who were
born thousand years ago," he said,
"it was easy for them to become immortal!
Lucky also, who was born
hundred years ago,
then there was still something to versify;
now the world is emptied by poets,
what should I compose into it today?"

The meaning of "distance" between text segments should be clear: "d" stands for the maximal number of lines between the end of one segment and the beginning of another segment that are hypothetically linked together. But what is the meaning of "distance" in audio recordings or scenes in a video file?

Audio files:
The mediaplayer counts the audio stream in 1/10 of a second. The distance value in retrieval runs or linkage hypotheses is multiplied by 10, that is, distances are defined in seconds. The default setting d=3 causes then that audio segments beginning maximally 3 seconds after a previous segment are accepted as "related" to this segment.

Video files:
The mediaplayer counts the video stream in frames, that is, single pictures. The distance setting is multiplied by 25, that is, according to the PAL norm of 25 frames per second, the distance is checked in seconds again. The default setting d=3 causes then that video segments beginning maximally 3 seconds (or 75 frames) after a previous segment are accepted as "related" to this segment.

8.1.4 How to learn about unused codes

This option in the menu “*Retrieval*" starts a one-dimensional search. It is often quite helpful in a large project to learn which of the many codes were *not* used in connection with which texts. AQUAD takes the momentary content of the master code file or any selection of codes as criterion for this type of search.

8.1.5 How to retrieve codes, count them and enter their frequencies into tables

Information about the frequencies of selected codes of a project gives access to various quantitative analyses, which may be really meaningful – depending on the research question (see chap. 11). You may create frequency tables for codes in AQUAD and import them later into software for quantitative analysis like spreadsheets or statistical programs. This is true also for sequential codes, which are applied by AQUAD automatically during linkage analyses.

In any case you start by having AQUAD "*Count codes*," one of the options in "*Retrieval*" in the main menu. AQUAD summarizes the results in a simple frequency list (listing of counted codes and their frequencies separately for each data file in your project). You save this list under any name in the subdirectory ..\res by clicking on the symbol of a diskette in the tool bar on top of the result window.

Statistic programs are at a loss with listings of this type, therefore we have to convert them into tables structured by codes (=variables) filling the columns and cases (files) defining the rows. The cells of these tables then contain frequencies for each (selected) code in each of the cases of our project. There is a specific conversion routine within the options of "*Implicants*" or "Qualitative Comparative Analyse" (see chap. 13), and a general tool, of course, in the group "*Tools*" for converting lists into tables.

There are two alternatives for saving the resulting tables, which you can apply both for any table: (1) Code frequencies are stored in AQUAD's proprietary table format ".adt" (AQUAD data table) and can be opened directly for table analyses (see 8.2.2) – if the number of codes does not exceed the limits for this sort of analysis; (2) the code frequencies are separated by commas and saved line by line, resulting in a "csv" table (comma separated values), which can be opened directly in spreadsheet programs (for instance, Exel or QuattroPro) or in statistical software packages (for instance, SPSS). Details of how to apply this tool can be found in chapter 13.

8.2 How to assemble coded data segments and examine how codes are associated with each other

The main function of AQUAD is to support the endeavors of researchers to find and check relations of meanings in their data texts. The program does this with algorithms which apply the principle of "backward deduction." That is, AQUAD takes your assumptions about how codes could be associated and checks all data files to find whether the critical codes can be found in matching sequences and distances. According to Glaser and Strauss (1967) each of your many codes represents an analytic category or a conceptual element of an emerging theory. Each coded text segment can be conceived of as part of a proof for the corresponding category in your data. While reading your texts, structuring them, comparing them and having now and then a look into your memos you will soon start to develop assumptions about relations between some of the categories used in your analysis, for instance, "Whenever these people mention critical life events, they begin to talk about emotions ... ." Emerging relations or associations may give rise to the formulation of hypotheses about latent contents of your texts.

In chapter 6 we outlined how inductive and deductive processes are necessarily combined in text analysis. Thus, you could also start from the opposite end, that is from some hypotheses about possible associations of categories and try to find data supporting these hypotheses deductively in your texts. If you do not succeed or if you experience contradictions, you would try to apply an inductive approach, that is to detect categories and their relations emerging from your texts. Of course, this is a very abbreviated description of the interchange of inductive and deductive processes in qualitative analysis.

Anyway, you will reach a point where you want to formulate a statement about systematic relations in your text, which developed slowly from your growing insight into the texts or from some general assumptions about associated categories. In other words: you formulate a hypothesis. Within text analysis we understand a hypothesis principally as a statement that particularly coded text segments represent the categories underlying these codes appear systematically associated. Let us use an example from an analysis of biographical interviews: There is a linkage between "critical life event" and "emotional reaction."

To examine associations of codes is simple, at least in principle. You enter the codes which you assume to be linked somehow, and the computer starts to check whether there are some segments in your texts which were coded in this way. That is, the computer is searching for the codes you think they may be associated *and*

the computer is searching for indicators of the type of association you assume. The advantage of using a machine is, that it will find everything in your texts, every relevant code, every indicator. Unlike usual data bank software, AQUAD does not only compare fields within records, but it compares each element of its entries with each other.

To tailor this search principle to your needs, AQUAD offers a number of variations. Generally, there are four approaches examining associations between codes or to "test" your hypotheses, which are described subsequently according to different research strategies (see chapter 6, section 6.3). AQUAD examines

- associations within the *context of unconditioned categories*, that is, the software examines a particular area of text preceding and/or succeeding the text segments of a critical category. You determine this category as well as the size of the text area to be checked around its text segments. Within this area any other categories are registered without any further conditions.

- associations within the *context of conditioned categories*. You define the "condition" for reporting possible systematic links by naming a second category, which has to be valid for a given text segment together with the first category.

- associations in form of *simple sequences of codes*. A number of abstract sequences are already implemented in AQUAD. All you have to do is start a concrete search for one of these sequences by entering the codes you expect to be associated and the number of lines which you take as maximal distance within which you will accept the two critical codes as sequentially associated.

- associations in form of *complex relations of codes*. To examine assumptions of this type you formulate them as hypotheses, first in plain English, then in a format that can be more easily used by the software as a template for a close examination of your texts.

8.2.1 Associations within the context of unconditioned categories: Searching for codes

You are already familiar with the simplest form of non-linear search or examination of data files. You have already learned about the various possibilities of searching for coded data segments and codes by using AQUAD's strategies of code retrieval.

The simplest strategy is to have the computer register all codes within a defined distance around any other particular code. Somewhat more complex than reporting "sequences of codes" are strategies by which particular patterns of codings are searched, as for instance multiple codings of a data segment. For example, when you search for "coding structures," you restrict the search by determining particular codes as criteria. A "category" is here understood as a pattern or structure of coding.

What is *not* necessary to make a retrieval successful is an additional relation of a given text segment to at least a second category. Thus, for instance, the search algorithms described above are unable to distinguish between overlaps of codings in texts of younger vs. older speakers. A retrieval task of this type requires an algorithm for "conditioned" categories, the additional condition here being "age."

8.2.2 Associations within the context of conditioned categories: Table analysis or matrix analysis

Miles and Huberman (1994) recommend and use in their book matrix displays as an important strategy to structure the content of texts. Above all they intend to elaborate an overview on sequential configurations of statements, ideas, opinions expressed in the texts by transforming them into a simultaneous configuration. Matrices help, according to Miles and Huberman (1994, pp. 93 f.)

- to gain or to maintain an overview, because data and analysis are held together;
- to identify rapidly, where additional analyses are necessary;
- to compare data and interpretations of data,;
- to communicate the results of a study.

Further hints regarding how to organize verbal data in matrices can be found together with numerous examples in Miles and Huberman (1994, Chap. IV, V and VI). For computer assisted examinations of associations the researcher constructs code matrices (Miles & Huberman 1994) or code tables (Shelly & Sibert 1992). The double determination of the cells of a matrix structures your interpretative efforts much more then mere information about a contingency (in terms of close distance in the text) of otherwise unrelated categories. On the other hand, you will need advanced conceptual insights into your data texts to determine a matrix suited for your analysis.

> From an analysis of biographical interviews (Huber & Kenntner, 1988) we take two interviews to demonstrate the principle of table analysis. One of the speakers was a woman, the other was a man. The columns of our table are determined by gender as a "singular" characteristic. Therefore, we introduce the profile-codes "gender: female" and "gender: male". We want to examine the assumption, that there is a difference between female and male interviewees as regards emotionality (conceptual code "emotional reaction), experience of critical life events ("critcal life event"), and their social relations ("social relation"). Of course, we will examine all data texts in our study, but to demonstrate the principle of matrix analysis, two texts should be sufficient. The rows of our matrix are determined by "emotional reaction", "critical life event", and "social relation". When we start the matrix analysis, AQUAD will gather all text segments in one cell, where women talk about emotional reactions; in the neighboring cell (next column) we will find what men said about the same topic. In the row below we will find first reports of women about critical events in their life, and in the next cell to the right reports by men. In the last row the cells show statements about social relations, in the first cell statements by women, in the following cell statements by men.

Matrices are well suited to structure data, because great amounts of information can be organized simultaneously in their cells instead of sequentially as in the original data. If it is true that one picture tells more than 1000 words could do, then a matrix display substitutes at least 100 words – above all because it keeps data and analysis visibly together.

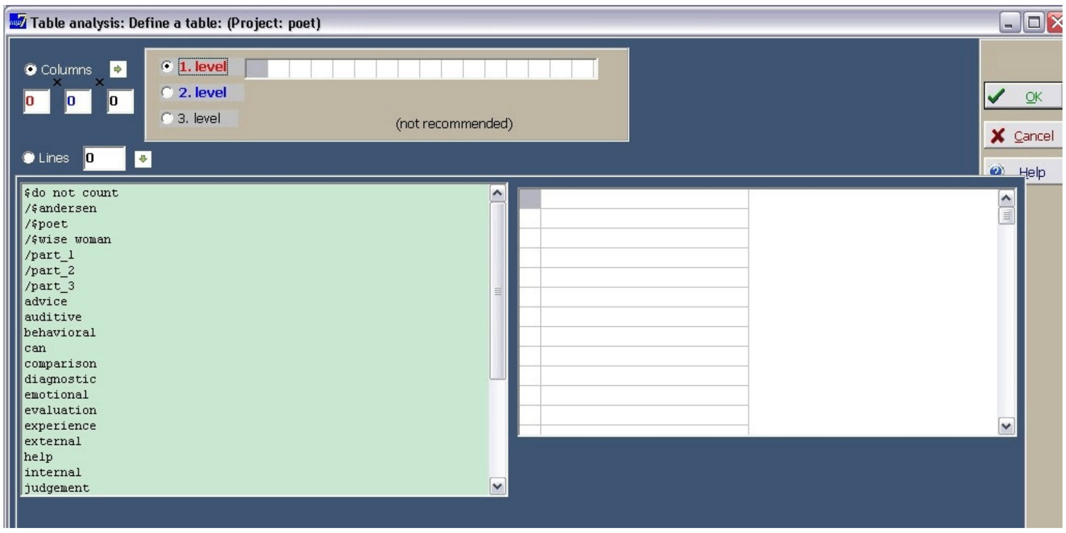
However, there is a restriction: Even a well differentiated matrix display of verbal data is nothing more than a valuable prerequisite for further conclusions; the display *is not* a conclusion. Besides, a matrix display is limited in the definition of column headers to categories representing singular characteristics like gender or school type, which are applied only once in each data file. If you intend to work with specific combinations of conceptual categories, which could be used for coding several times in a file, you have to apply more complex strategies.

You find the software functions for matrix or table analysis in the main menu under "*Tables.*" You should study the use of "*Create a table*" and "*Edit a table*" carefully. The next three options "*Analysis: text segments*" (of course available only, if you work with texts as data), "*Analysis: codings*," and "*Analysis: frequencies*" determine in which form you will get a report. As you may assume, you have a choice between a complete print-out of coded text segments in the cells of your matrix, only the codings, or just the frequency of matching cell contents.

How to design a table

To design a table for a meaningful two-dimensional search for data, you need of course at least two data files, which differ at least in one profile characteristic (or "singular" characteristic). And you should have used at least two codes in your work. One of these codes has to be a profile code for a whole data file. In the above example we introduced the category "gender" represented by the profile codes "gender: female" and "gender: male". Both characterize interview texts in total and could be used to define the columns in a table. In the sample project "Poet" we applied "/part_1," "/part_2" and "/part_3" as profile codes characterizing three files containing three different parts of the story.

The other codes (at least one!) defining the rows of the matrix will be conceptual codes, of course. Probably they were used several times within each text of your study. In our example above we took the codes "emotional reaction", "critical life event", and "social relation" as relevant for a table analysis. In our sample project "Poet" we could apply, for instance, the codes "behavioral", "auditive", "visual", "emotional", "taste", "think", etc. as row headers to find out, whether there are differences between the parts of the story as regards external action versus internal activities. Whenever AQUAD detects one of these conceptual codes in the code files related to our example texts, it will put the matching text segment into the appropriate cell (assuming we choose later the option "*Analysis: Text segments*"). First you have to find out how to enter these codes:



```
$do not count
/$andersen
/$poet
/$wise woman
/part_1
/part_2
/part_3
advice
auditive
behavioral
can
comparison
diagnostic
emotional
evaluation
experience
external
help
internal
judgement
```

After clicking on "*Create table*" the above window will appear. You see two small, round buttons in the upper left corner labeled "*Columns*" and "*Rows*." The screen shot shows that "*Columns*" is already activated (there is a little dot in the button), however, no column headers are determined yet. Therefore, almost everything else is still blank.

Right of these two buttons is a field, where another selection was already made: the button "*1st level*" was already pushed. Below, a window showing the master code list is opened, and on the right is an empty list, were your selections will be confirmed. If you click now on a code in the master code list, it will be determined as a first-level column header and transferred to the white box on the right side.

Please, remember: **As column headers, you determine socio-demographic codes only** (here also called profile codes or singular codes)!

In our sample project "Poet", we would click on "/part_1" and "/part_3" as the only relevant profile codes available (in the second part of the story the hero plays only a minor role). Both will be transferred to the white box on the right, and in the little cells following the button "*1st level*" above, the abbreviations "A" and "B" will be entered. As you may conclude from the number of these cells, you may work with maximally 12 first-level column headers. At the same time, the first of the three little counter boxes on the left will be filled by numbers, first "1", then "2" is appearing as we enter the first and the second column header (see first graphic on the next page).

In case we had a lot of text files, we could differentiate our column headers by a series (maximally six!) second-level headers. For example, we could differentiate in the example outlined above between male and female interview partners according to age groups (young, middle, old). Thus, we create 2x3 columns, and we need at least 6 data texts, each one of them matching one of the column determinants, to run an analysis. In our sample project "Poet" we have only three text files or cases, therefore, a differentiation would be impossible. In same cases, with very much data texts available, you may even consider a third level of differentiation – this is possible in AQUAD, however, we do not recommend it.

After we are finished with the column headers, we fill in some row headers. Click on the button "*Rows*". Now we select appropriate conceptual codes, for instance "behavioral", " auditive", etc. as suggested above. Again, the selection is transferred to the white box on the right side, and the number of rows determined is counted in the little box below the "*Rows*" button (see second graphic on the next page).

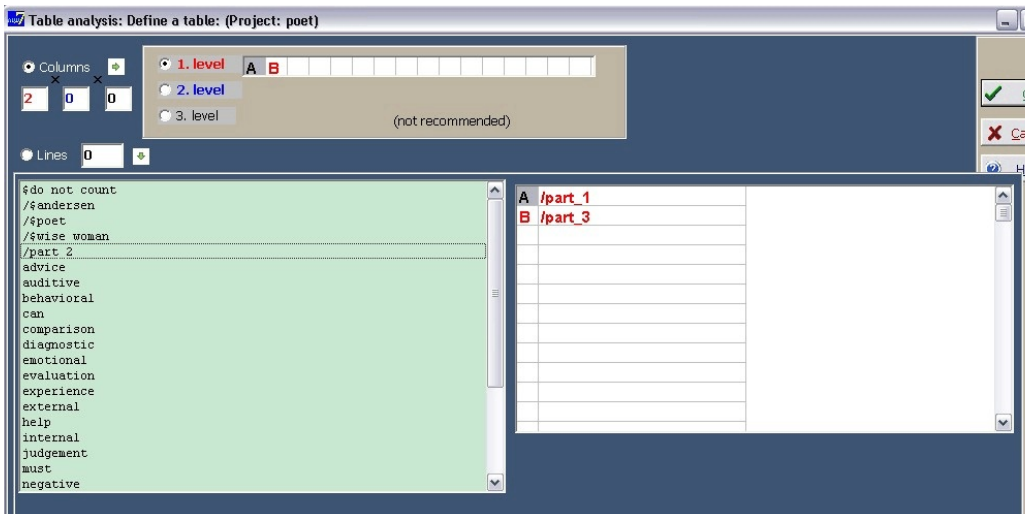


Table analysis: Define a table: (Project: poet)

Columns: 2 × 0 × 0

[✓ 1. level] A B

2. level

3. level (not recommended)

Lines: 0

```
$do not count
/$andersen
/$poet
/$wise woman
/part 2
advice
auditive
behavioral
can
comparison
diagnostic
emotional
evaluation
experience
external
help
internal
judgement
must
negative
```

A	/part_1
B	/part_3

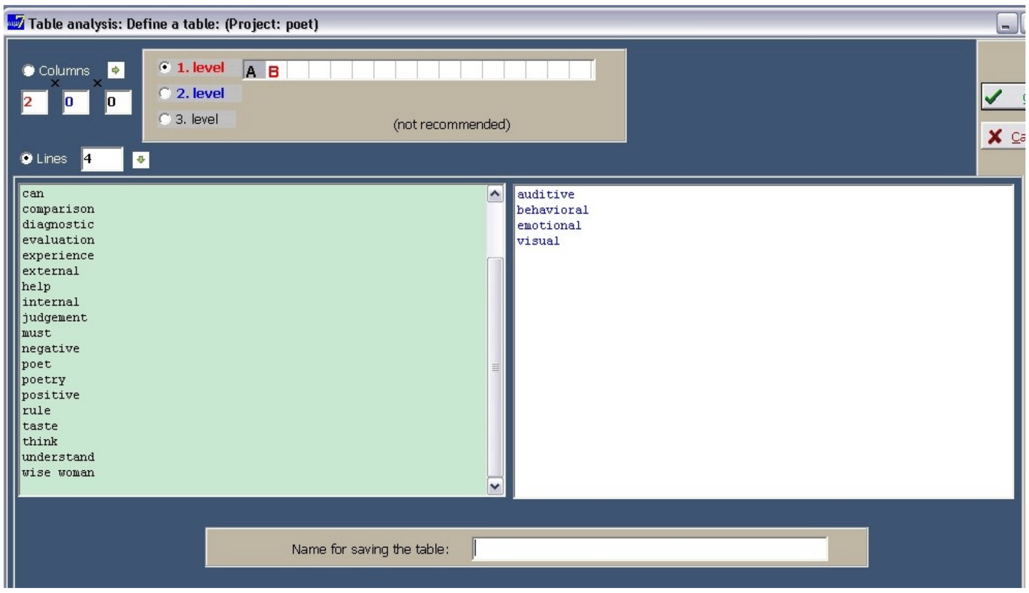


Table analysis: Define a table: (Project: poet)

Columns: 2 × 0 × 0

[✓ 1. level] A B

2. level

3. level (not recommended)

[✓ Lines]: 4

```
can
comparison
diagnostic
evaluation
experience
external
help
internal
judgement
must
negative
poet
poetry
positive
rule
taste
think
understand
wise woman
```

```
auditive
behavioral
emotional
visual
```

Name for saving the table:

Now we have set up the table. We are ready to click on the "*OK*" button. When we do so, a little entry field will appear at the bottom of the window. We are asked to enter a name (no extension!) for saving our table design. After another click on the "*OK*" button, these settings will be saved for later use in a table analysis. We suggest that you play with these suggestions for a sample table.

Summary:

① You click on the "*Columns*" button and select the first level of column headers.
② You fill in up to twelve socio-demographic (profile) codes as column headers.
③ If necessary for your analysis, you add additional levels of column headers (sub-ordinate profile codes).
④ You click on the "*Rows*" button and fill in conceptual codes as row headers.
⑤ After clicking on the "*OK*" button you enter a name for saving your settings and click on "*OK*" again.

How to edit a table

If you want to see how the above window looks like after you entered all column and row headers suggested above, follow the instructions, save the resulting table (for instance, as file "modes") and go then to "*Edit table*" in the sub-menu "*Tables*." Just switch between the options "*Columns*" and "*Rows*" to have a look.

You can easily modify the table entries. Clicking on one of the selected column headers or codes in the right window will move it back to the master code file shown in the window on the left side. Selecting a code from the master code file by clicking on it in the left window will make it part of the modified selection for a table analysis.

If you do not want to overwrite the new table setup which you just edited, please, enter a different name for saving into the little entry field at the bottom (appearing only after you clicked the "*OK*" button).

How to run a table analysis

There are three possibilities, which function according to the same principle, however, they produce results differing in information value.

- Available only while you are analyzing text files and most comprehensive is an "*Analysis: Text segments*". As you may assume, its resulting output contains all those text segments fulfilling the double conditions for cells of your table design. Please, be aware that a print-out of a text matrix may consume great amounts of paper and printing time. Neither printer nor monitor are generally able to display all the columns together. Therefore, AQUAD prints one cell after the other, column after column. On a blackboard, an empty wall or any other appropriate area you can afterwards put the printed cells back into their matrix pattern.

- Available with all sorts of data and more economic is an "*Analysis: Codes*". Its output contains in its cells only codings of data segments which fulfill the combined criteria of column and row headers. You will take these codings and go back to your data files, for instance, line-numbered data texts or frame-numbered video files to scrutinize the data segments.

- Most economic is an "*Analysis: Frequencies*"; however, it is least informative. You see in one glimpse how often conceptual codes appear in your data under the condition of profile codes. Where they appear and attached to which statements is *not* reported. Usually this option is used as a heuristic tool for getting some hints as to systematic relations (or non-relations) in the data files. Here are the frequency results for our example project "Poet" using the table setup described above:

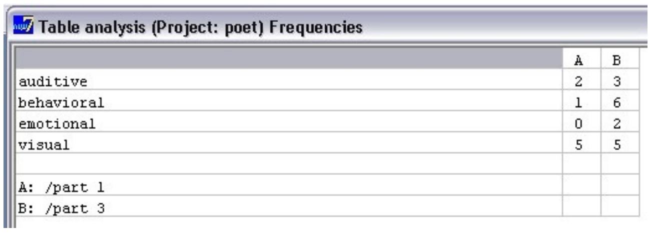


Table analysis (Project: poet) Frequencies

	A	B
auditive	2	3
behavioral	1	6
emotional	0	2
visual	5	5
A: /part 1		
B: /part 3		

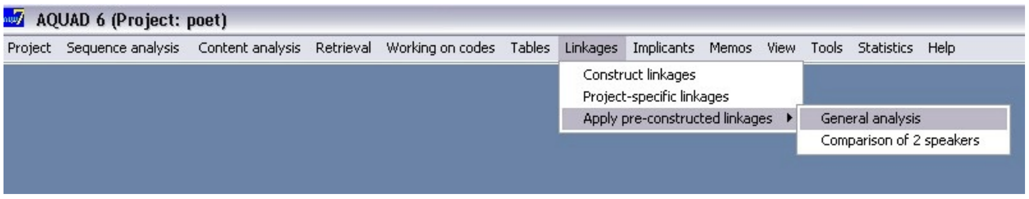
We see, that the expected difference between the texts seems to be true: There is only one overt action (conceptual code "behavioral") in "part_1" (abbreviated by "A" in the column header), but six segments coded as "behavioral" in "part_3" (column header ("B"). It would be a good idea now to go back to the "*Working on codes*" options, have AQUAD retrieve the text segments to which "behavioral" was attached, and to read and compare these segments carefully before we draw any further conclusions.

8.3 Associations in form of simple sequences of codes: Checking linkages

The program module "*Linkages*" is the most important for theory-generating or theory-reconstructing analyses of verbal data. This module applies deductive logic to check systematic linkages of text segments. Each one of your various codes is seen as representing a "category of analysis" in the sense of Glaser and Strauss' work: "A category stands by itself as the conceptual element of the theory." (Glaser & Strauss, 1967, p. 36). Each coded segment is evidence for the occurrence of that category in your data. As you work with your data and look at your conceptual categories, you may form hunches about relationships that seem to exist among some of these categories. Such relations or linkages could become the beginning of theoretical notions about what is going on in your data. Thus, you could also say, this module allows testing hypotheses about linkages. You formulate how you suppose text segments might be linked in your texts, and the computer checks all entries in your code files to confirm or disconfirm your assumption. The result is not just a report that your assumption could be confirmed, but you receive a list of all codings or combinations of codings that fulfil the conditions of your hypothetical linkages. With this information given, you can start well focused comparisons.

Within AQUAD all sorts of sequences of codes can be checked. Some patterns of sequences are implemented in the software for immediate use in the form of abstract structures of linked codes. When you run a linkage analysis, you enter some concrete codes from your study and transform thus the abstract structures into concrete assumptions about linked codes. To start the analysis you choose the option "*Linkages*" from the main menu, then you select "Apply preconstructed linkages" from the sub-menu and decide, which of two alternatives you want to apply: The option "*General analysis*" retrieves any sequence of hypothetically linked codes across all files of a project. If necessary, the output may be structured by speakers, that is findings within speaker segments (marked by specific speaker codes) are sorted and reported for each speaker within a file separately. The option "*Comparison of 2 speakers*" on the other hand analyzes particular linkages between subsequent data segments of two defined speakers, for instance: "If speaker 1 talks about 'A', then speaker 2 refers to 'B.' Or: "If a subject answers with 'A' in question 3 of a questionnaire, he/she will answer 'B' in question 5."

Of course, AQUAD also supports analyses of complex linkages. However, because of their complexity and variety, no abstract structures could be pre-constructed and implemented into the software. Instead, the option "*Construct linkages*" gives you an opportunity to implement your assumptions about complex linkages of code easily by clicking on code names and logic connectors. How to do so is described in detail in chapter 12. Because of pragmatic reasons the number of codes to be linked in your own constructions is limited to 5. The following screen shot from the main menu shows all possibilities how to check linkages in AQUAD:

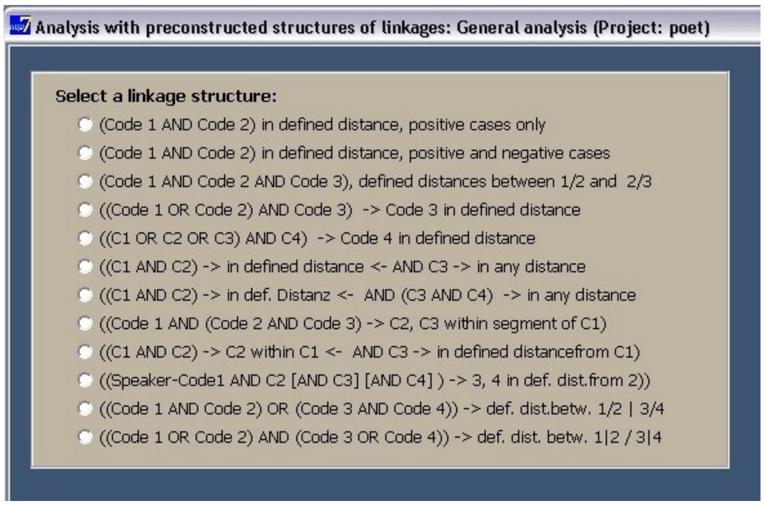


"*Project-specific linkages*" are not implemented into this version of AQUAD, but they are available on demand. If necessary the author will readily construct the linkage algorithm which you need for answering your research question – but in most cases the preconstructed linkage structures and the possibilities offered in "*Construct linkages*" will solve your problem.

8.3.1 Which linkage structures are available in AQUAD?

General sequences of codes

AQUAD provides 12 hypothesized "linkages" into which you may enter any of your codes. The hypotheses specify the relationships you want to test, and sometimes the distance between pieces of evidence you wish not to exceed. These preconstructed hypotheses were not set up abstractly, but have been developed during the programmer's own research as well as in the supervision of doctoral theses, and for introductory courses of doctoral candidates into qualitative methods. Below is a list of all hypotheses currently available in AQUAD. This list appears when you select "*Apply preconstructed linkages*" -> "*General analysis*" from the sub-menu of the "*Linkages*":


Analysis with preconstructed structures of linkages: General analysis (Project: poet)

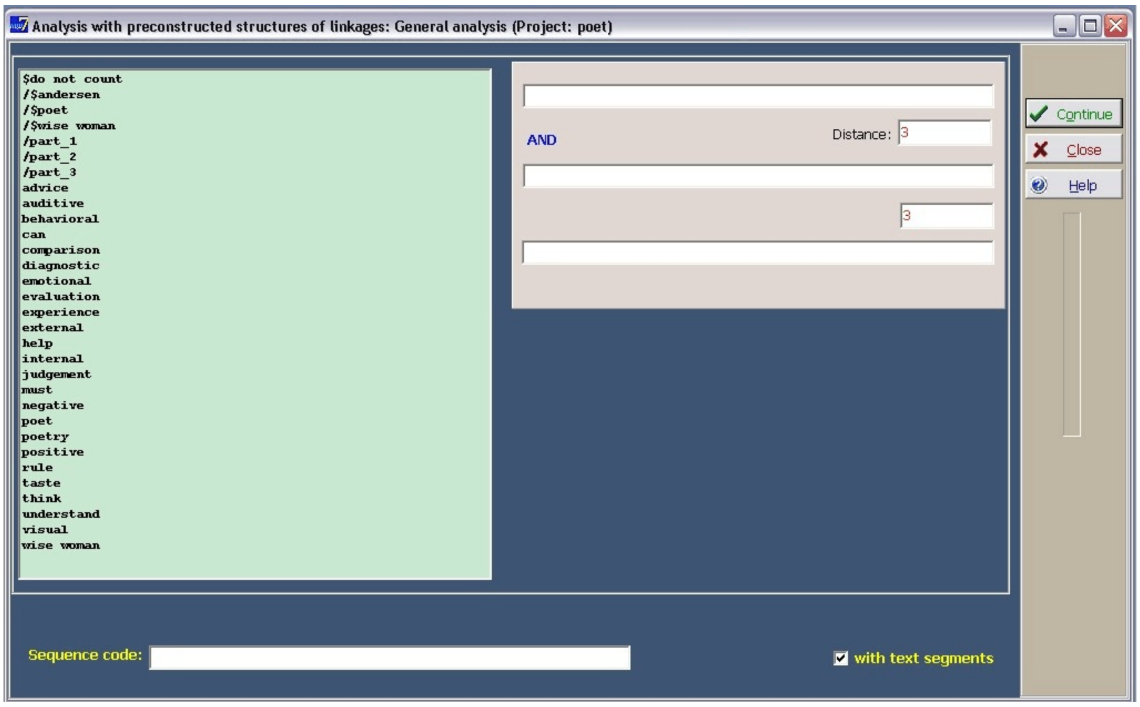
Select a linkage structure:

- (Code 1 AND Code 2) in defined distance, positive cases only
- (Code 1 AND Code 2) in defined distance, positive and negative cases
- (Code 1 AND Code 2 AND Code 3), defined distances between 1/2 and 2/3
- ((Code 1 OR Code 2) AND Code 3) -> Code 3 in defined distance
- ((C1 OR C2 OR C3) AND C4) -> Code 4 in defined distance
- ((C1 AND C2) -> in defined distance <- AND C3 -> in any distance
- ((C1 AND C2) -> in def. Distanz <- AND (C3 AND C4) -> in any distance
- ((Code 1 AND (Code 2 AND Code 3) -> C2, C3 within segment of C1)
- ((C1 AND C2) -> C2 within C1 <- AND C3 -> in defined distancefrom C1)
- ((Speaker-Code1 AND C2 [AND C3] [AND C4]) -> 3, 4 in def. dist.from 2))
- ((Code 1 AND Code 2) OR (Code 3 AND Code 4)) -> def. dist.betw. 1/2 | 3/4
- ((Code 1 OR Code 2) AND (Code 3 OR Code 4)) -> def. dist. betw. 1|2 / 3|4

Before we elaborate on these linkages and examine some examples, here is a short description of the meaning of the very abbreviated formulations on the computer screen. The first sentence repeats the hypothesized linkage, the second sentence describes what the computer is checking when you activate this particular linkage "hypothesis:"

Linkage 1:	Two codes appear in the same data file within a particular maximal distance from each other. In which files is this statement true?
Linkage 2:	Two codes appear in the same file within a particular maximal distance from each other. In which files is this statement true? In which texts is it false?
Linkage 3:	Three codes appear in the same file – code 2 within a particular maximal distance from code 1, while code 3 appears within a particular maximal distance form code 1. In which files is this statement true?
Linkage 4:	One or both of two codes appear in the same file together with a third code within a particular maximal distance. In which files is this statement true?
Linkage 5:	One, two or all of three codes appear in the same file together with a fourth code within a particular maximal distance. In which files is this statement true?
Linkage 6:	Three codes occur in a file; two of these codes always appear within a specified distance of each other in the file, while a particular third code is located somewhere in this file – this code is just a particular condition linked to the whole data document only. In which files is this statement true?
Linkage 7:	Four codes occur in a file; two of these codes occur within a specified distance of each other in a document, while a particular third and fourth code (just as additional global conditions) are located somewhere in the data document. In which files is this statement true?
Linkage 8:	Three codes occur in the same data document, with codes #2 and #3 being sub-codes of code #1 (= they occur within the data segment of code #1). In which files is this statement true?
Linkage 9:	Three codes occur in the same data document, with code #2 being a sub-code of code #1, and code #3 occurring within a particular maximal distance of code #1. In which files is this statement true?
Linkage 10:	One, two or three codes occur in the same data document *and* within the data segment(s) of a particular speaker code (code#1), with codes #3 and #4 occurring within a particular maximal distance of code #2. In which files is this statement true?
Linkage 11:	Two particular codes or a pair of two alternative codes occur in the same data document within a specified maximal distance of each other. In which files is this statement true?
Linkage 12:	A specific code#3 or an alternative code#4 occur(s) in a particular maximal distance of another code#1 or its alternative code#2. In which files is this statement true?

If you want to work with any of these hypotheses, you merely choose the appropriate formulation, then you enter the codes and in most cases additionally the maximal distances you are interested in. Let us assume, we are interested in the preconstructed pattern #3. For this purpose, a special entry window is opened:



In this screen shot, the distances between data segments coded by code #1 and those by code #2 as well as the distance between segments coded by code #2 and those by code #3 are already entered. You see the default distance of maximally three lines (or 3 seconds/75 frames in case of audio/video data) in the little boxes on the right. Just click into the boxes and enter a different distance, if necessary in your project. What we have to do now is enter the code names. To do so, you just click on the appropriate code name in the master code list on the left. It will be automatically transferred into the next empty entry field. Double-clicking on this field clears it again and brings its content back to the master code list.

Whenever a distance specification appears in a linkage structure, you may disregard it by entering a number that is at least as long as your entire data file (such as 9999). This way all occurrences of the two codes in the file will be captured. Also, note that the order in which the two codes are entered into the entry window is important, since this linkage formulation takes into account the sequence of the coded segments. The first code entered must occur in the data first. If the second code occurs first in a data file, and later the first one, there is no "match," i.e., for this particular case the linkage hypothesis is rejected.

Two yellow labels on the bottom of the window show

(1) that each finding of linked codes can be coded automatically by a sequence code, which you have to write into the empty slot behind the label.
(2) that in case of text data the program will attach the coded text segments to any confirmed linkage hypotheses – if you mark the little box in front of the label;

Examples

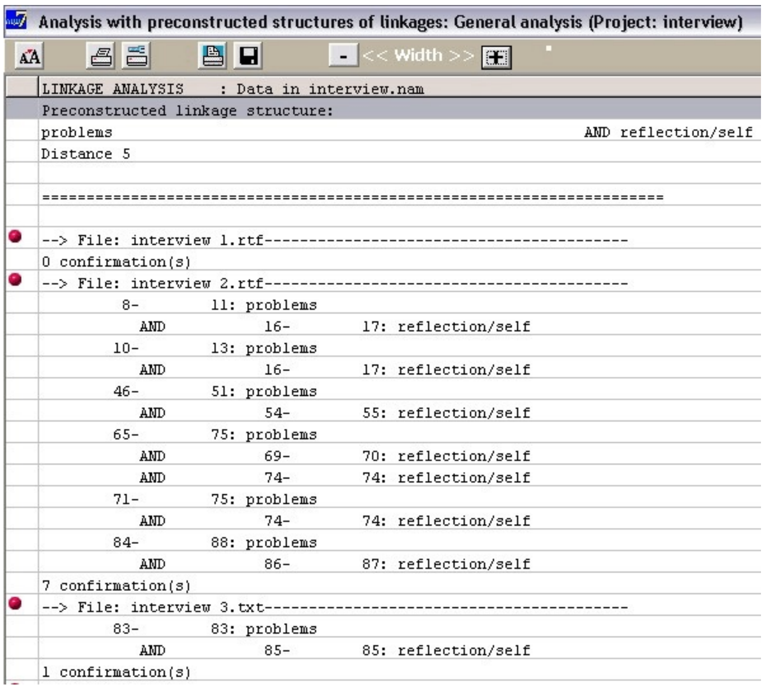
In the remainder of this chapter, the first six of the preconstructed linkages are applied and explained in detail. Since the linkage formulations are rather abstract, we will illustrate the usefulness by providing an empirical example.

Linkage 1

This hypothesis states that there is a linkage between two particular codes (categories), i.e., that these two codes appear regularly within a particular maximal distance from each other in the data files. When using this hypothesis formulation, we are interested only in getting a display of positive results, i.e., of all instances in which the combination of coded segments was indeed found within the specified distance from each other.

> *Example*: In the – somewhat fictitious – interview texts on the AQUAD CD, which were taken from a study of Marcelo (1991) and enriched with quotations found in Zabalza (1991), we found many short reflective references to the interviewees' mood state, competencies, etc. We got a hunch that these teachers started to make some remarks about their own state whenever they mentioned problems in their classrooms. As a hypothesis we formulated: "If the interview partners talk about critical situations in their classrooms (code: problems), then they talk about themselves and their (emotional) state (code: reflection/self) in close relation, i.e., within maximally 5 lines of text in the interview transcription."
>
> In order to check the interview transcriptions (or actually: their corresponding code-files) with this linkage formulation (no. 1), we entered the codes in the entry window. The box "show text segments" was not marked, because we did not want to fill the pages of this manual with text you have already on your hard disk (see project "interview").

We were presented with the following results: The header contains the file catalog used in this analysis. Additionally we are informed, that we tested a preconstructed linkage, here determining a linkage of two codes, "*problems*" followed by "*reflection/ self*" within a distance of maximally 5 lines of text. The remaining output displays the results for the first three interviews only:.


Analysis with preconstructed structures of linkages: General analysis (Project: interview)
<< Width >>

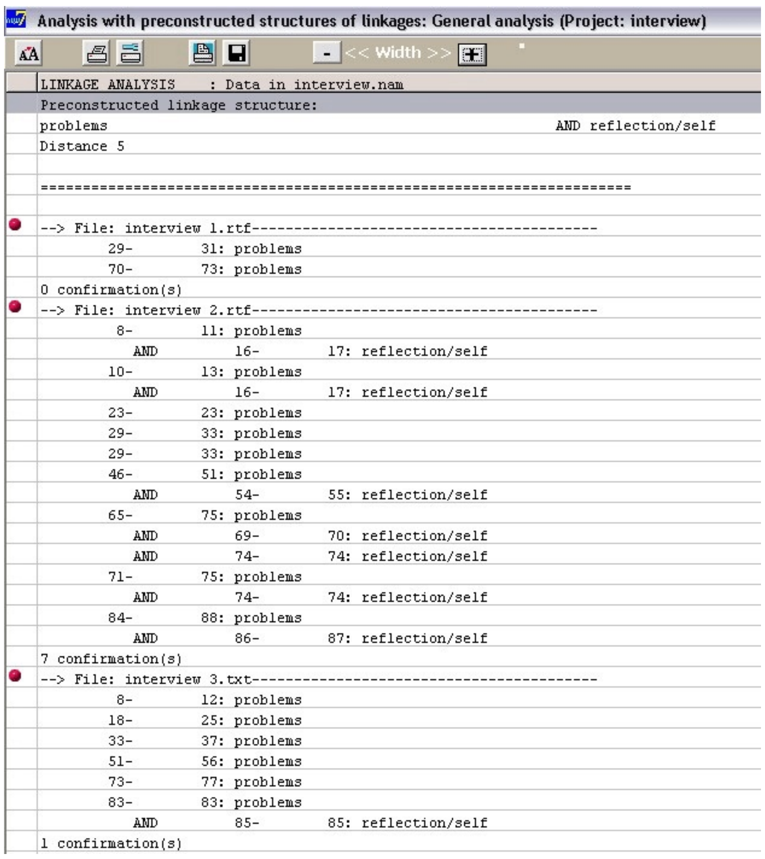
```
LINKAGE ANALYSIS    : Data in interview.nam
Preconstructed linkage structure:
problems                                              AND reflection/self
Distance 5

===================================================================

--> File: interview 1.rtf-----------------------------------------
0 confirmation(s)
--> File: interview 2.rtf-----------------------------------------
         8-        11: problems
           AND           16-        17: reflection/self
        10-        13: problems
           AND           16-        17: reflection/self
        46-        51: problems
           AND           54-        55: reflection/self
        65-        75: problems
           AND           69-        70: reflection/self
           AND           74-        74: reflection/self
        71-        75: problems
           AND           74-        74: reflection/self
        84-        88: problems
           AND           86-        87: reflection/self
7 confirmation(s)
--> File: interview 3.txt-----------------------------------------
        83-        83: problems
           AND           85-        85: reflection/self
1 confirmation(s)
```

The screen shot above shows, for instance, that no problems were related to reflections in interview_1, while in interview_2 six problems were linked with seven reflections, and so on.

Since we asked for a display of positive results only in linkage no. 1, it is difficult to decide whether interview_1 calls into question the postulated relationship of problems and reflections, or whether the interviewed person just either has not experienced any problems in the classroom yet or does not relate problematic situations to the own, personal state. A test run with hypothesis no. 2 could shed more light on the question.

Linkage 2

This linkage hypothesis works just like linkage 1. However, here the researcher receives a display of positive as well as negative results. Negative are not only results where neither of the codes was found, or one of them was not found, but also those where both were found, but the second one did not follow within the defined maximal distance.


Analysis with preconstructed structures of linkages: General analysis (Project: interview)
<< Width >>

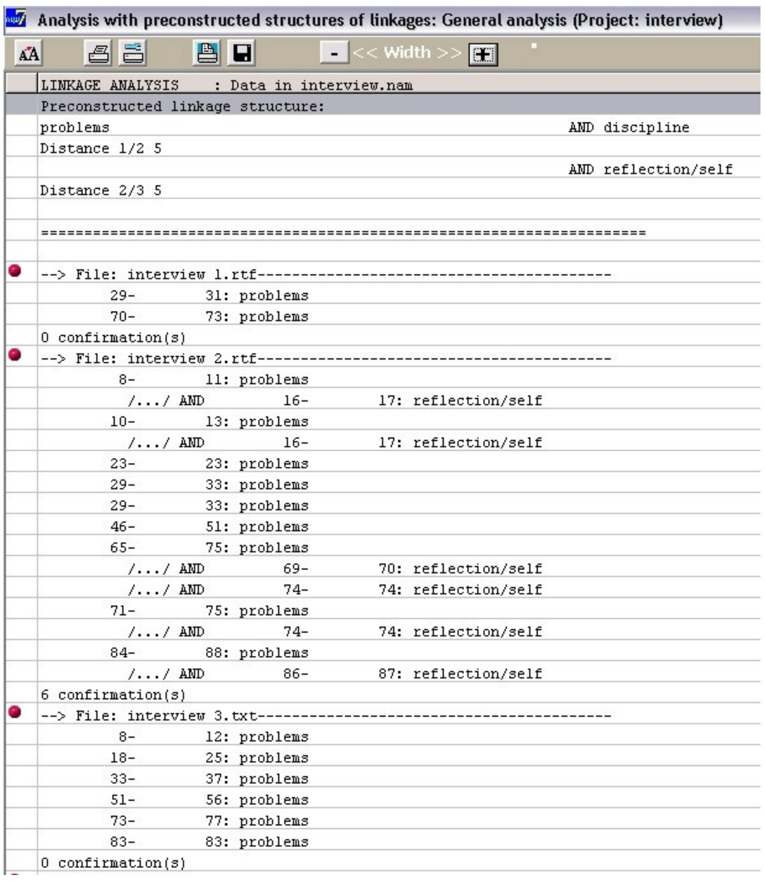
```
LINKAGE ANALYSIS    : Data in interview.nam
Preconstructed linkage structure:
problems                                                      AND reflection/self
Distance 5

======================================================================

--> File: interview 1.rtf-----------------------------------------
        29-        31: problems
        70-        73: problems
0 confirmation(s)
--> File: interview 2.rtf-----------------------------------------
         8-        11: problems
           AND         16-        17: reflection/self
        10-        13: problems
           AND         16-        17: reflection/self
        23-        23: problems
        29-        33: problems
        29-        33: problems
        46-        51: problems
           AND         54-        55: reflection/self
        65-        75: problems
           AND         69-        70: reflection/self
           AND         74-        74: reflection/self
        71-        75: problems
           AND         74-        74: reflection/self
        84-        88: problems
           AND         86-        87: reflection/self
7 confirmation(s)
--> File: interview 3.txt-----------------------------------------
         8-        12: problems
        18-        25: problems
        33-        37: problems
        51-        56: problems
        73-        77: problems
        83-        83: problems
           AND         85-        85: reflection/self
1 confirmation(s)
```

As you can see, the teacher in interview_1 talked twice about classroom problems, but none of these statements was followed by any self-reflective remarks. In interview_2, six of nine statements about class-room problems are followed by reflections. On the other hand, in interview_3 the teacher mentions six times classroom problems, but connects them only once with reflections.

Linkage 3

This hypothesis deals with three coding categories. As in linkage 1, we are concerned only with cases in which the relationship was found. Again, the proximity of the three different codes is of importance. Maximal distances may be defined for the codes 2 and 3 in relation to code 1.

Example: We extended the hypothesis from example 1. Now we assume that the interview participants specifically talk about classroom "*discipline*" when they relate "*reflection/self*" to "*problems*." Such statements should be found within maximally five lines after a statement about classroom problems is made. Here are our results:


Analysis with preconstructed structures of linkages: General analysis (Project: interview)
<< Width >>

```
LINKAGE ANALYSIS    : Data in interview.nam
Preconstructed linkage structure:
problems                                                    AND discipline
Distance 1/2 5
                                                            AND reflection/self
Distance 2/3 5

======================================================================

--> File: interview 1.rtf-----------------------------------------
        29-        31: problems
        70-        73: problems
0 confirmation(s)
--> File: interview 2.rtf-----------------------------------------
         8-        11: problems
          /.../ AND         16-        17: reflection/self
        10-        13: problems
          /.../ AND         16-        17: reflection/self
        23-        23: problems
        29-        33: problems
        29-        33: problems
        46-        51: problems
        65-        75: problems
          /.../ AND         69-        70: reflection/self
          /.../ AND         74-        74: reflection/self
        71-        75: problems
          /.../ AND         74-        74: reflection/self
        84-        88: problems
          /.../ AND         86-        87: reflection/self
6 confirmation(s)
--> File: interview 3.txt-----------------------------------------
         8-        12: problems
        18-        25: problems
        33-        37: problems
        51-        56: problems
        73-        77: problems
        83-        83: problems
0 confirmation(s)
```

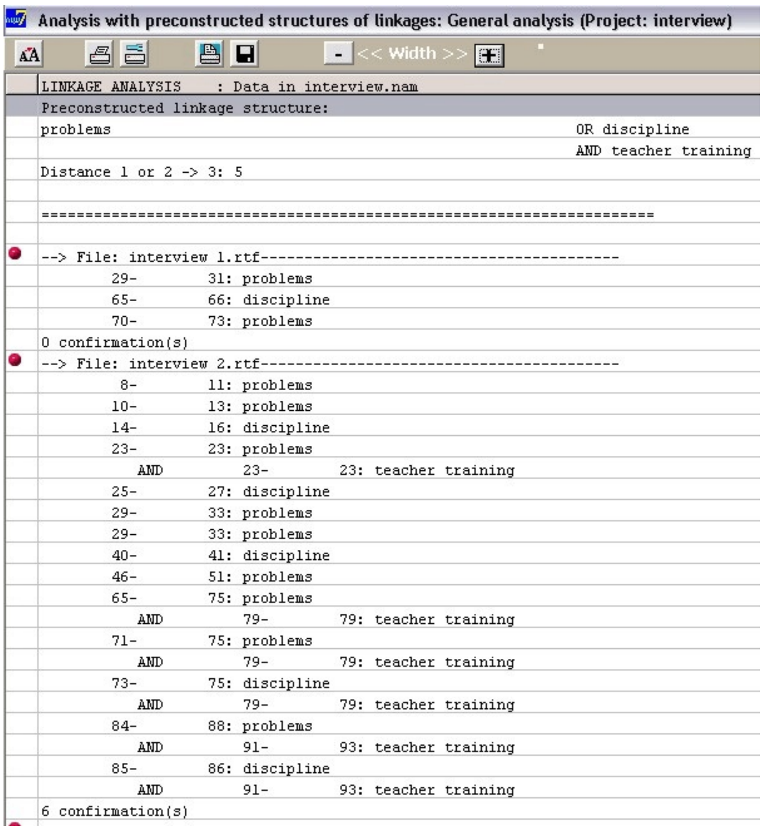
It is obvious that this hypothesis seems to be true in interview_2 (in five of nine cases in which "problems" were mentioned, but not at all in case 3. Now, of course, we should have a closer look into the original data file – however, these examples do not claim to unveil any deeper insights into these not completely authentic data (see introduction to linkage 1 above), but are only meant to illustrate how you apply and what you can learn from linkage analyses!

Linkage 4

In this case, the researcher is not interested as much in the co-occurrence of coded segments in the data, as in the questions where one or the other of two (perhaps similar) codes can be found. A search like this may also provide an overview of whether certain codes occur in the data, and where. In Boolean terms, they are logically related by the "inclusive OR;" i.e., we want information on where code 1 or code 2 or both codes have been used. The sequence in which the coded segments occur in the data files is of no consequence for this search; in other words, it does not matter which code you enter into the entry window first.

Example: We ask, whether the interviewees relate either "*discipline*" or "*teaching methods*" or both of them to their "*teacher training.*" That's what we get as answers (because of space requirements shown only for the first two interviews):

In interview_2 (and not shown here, in interview_3) we find – as expected – that teachers talk about their training, when they mentioned classroom problems and discipline. However, in which respect do they relate actual teaching experience and previous preparation? We are at a loss to find an answer in the results below, but have to go back to the original data – and take more seriously what was recommended: Neutral codes like "*teacher training*" do not allow to differentiate linkages; we have to code the quality of this training, too, as seen by the former student teachers. For instance, we could replace the neutral expression by "teacher training -" wherever critique is expressed.


Analysis with preconstructed structures of linkages: General analysis (Project: interview)
<< Width >>

```
LINKAGE ANALYSIS    : Data in interview.nam
Preconstructed linkage structure:
problems                                                      OR discipline
                                                              AND teacher training
Distance 1 or 2 -> 3: 5

======================================================================

--> File: interview 1.rtf-----------------------------------------
        29-        31: problems
        65-        66: discipline
        70-        73: problems
0 confirmation(s)
--> File: interview 2.rtf-----------------------------------------
         8-        11: problems
        10-        13: problems
        14-        16: discipline
        23-        23: problems
           AND         23-        23: teacher training
        25-        27: discipline
        29-        33: problems
        29-        33: problems
        40-        41: discipline
        46-        51: problems
        65-        75: problems
           AND         79-        79: teacher training
        71-        75: problems
           AND         79-        79: teacher training
        73-        75: discipline
           AND         79-        79: teacher training
        84-        88: problems
           AND         91-        93: teacher training
        85-        86: discipline
           AND         91-        93: teacher training
6 confirmation(s)
```

Linkage 5

This hypothesis is much like the previous one, but now we assume that three codes are related by the inclusive OR, i.e. one or two or all three of them occur in a given data file. The search results would look much like the ones for hypothesis 4, with the occurrences for the third code added.

How to compare codes of two speakers

Various speakers can be differentiated by individual "speaker codes"in a data file. We use speaker codes normally to attribute data segments – for instance, parts of an audio-taped group discussion – to the corresponding speakers. Another possibility to apply this type of code would be open questions in a questionnaire: in this case our "speaker codes" are used as "question codes:" /$question_1, /$question_2, etc. Thus we could analyze each question separately and compare easily the answers within the data files of our subjects.

The option "*Apply preconstructed linkages*" -> "*Comparison of 2 speakers*" is useful, if we want to find specific associations in successive data segments of two particular "speakers" (or questions; see above). Typical research questions that can be answered with this option are, for instance:

Is it true that students answer only in fragmentary sentences, if teachers ask closed questions?
Is it true that a person B always claims the contrary of what a person A has just said?
Is it true that in this questionnaire an answer X to question 3 is mostly followed by an answer Y to question 7?

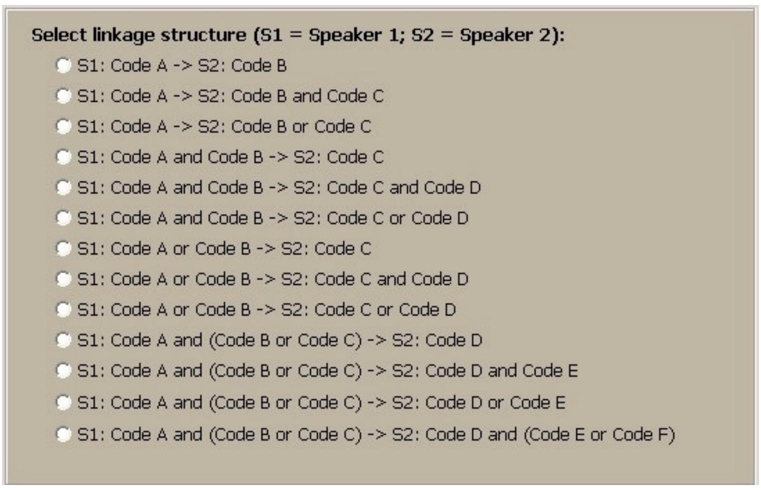
Particular linkages

Unlike within option "*General analysis*" the criterion of hypothesis testing is the fact that particular codes occur in data segments of two different speakers (or questions).

The labels A, B, C, ... attached to codes in the following descriptions imply only that we are looking for concrete data segments in the sections of different speakers – but not that A, B, C, etc. stand for different meanings in different sections. Or to put it another way, according to hypothesis 1 (see below) we should observe that speaker 1 says A and then speaker 2 says B; it is not necessary that speaker 2 states something different or even the contrary of what speaker 1 had said, but speaker 2 could simply confirm a statement of speaker 1, for instance:

A: "What a blue sky!"
B: "What a blue sky!"

Subsequently we show a list of all linkages for comparing two speakers actually available in AQUAD. The linkages are activated by selecting the option "*Apply preconstructed linkages*" -> "*Comparison of 2 speakers*" in the menu section "*Linkages*:"


Select linkage structure (S1 = Speaker 1; S2 = Speaker 2):
- S1: Code A -> S2: Code B
- S1: Code A -> S2: Code B and Code C
- S1: Code A -> S2: Code B or Code C
- S1: Code A and Code B -> S2: Code C
- S1: Code A and Code B -> S2: Code C and Code D
- S1: Code A and Code B -> S2: Code C or Code D
- S1: Code A or Code B -> S2: Code C
- S1: Code A or Code B -> S2: Code C and Code D
- S1: Code A or Code B -> S2: Code C or Code D
- S1: Code A and (Code B or Code C) -> S2: Code D
- S1: Code A and (Code B or Code C) -> S2: Code D and Code E
- S1: Code A and (Code B or Code C) -> S2: Code D or Code E
- S1: Code A and (Code B or Code C) -> S2: Code D and (Code E or Code F)

Here is a short description of the meaning of the very abbreviated formulations on the computer screen. The first sentence repeats the hypothesized linkage, the second sentence describes what the computer is checking when you activate this particular linkage "hypothesis:"

Linkage 1:	Speaker 1 says A, then speaker 2 says B in his/her next data segment. In which files is this statement true?
Linkage 2:	Speaker 1 says A, then speaker 2 says B AND C in his/her next data segment. In which files is this statement true? (We could test, for instance, the linkage: "If speaker 1 tells something about X, then speaker 2 refers also to X and describes a concrete example.")
Linkage 3:	Speaker 1 says A, then speaker 2 says B OR C (inclusive OR) in his/her next data segment. In which files is this statement true?

Linkage 4:	Speaker 1 says A AND B, then speaker 2 says C in his/her next data segment. In which files is this statement true?
Linkage 5:	Speaker 1 says A AND B, then speaker 2 says C AND D in his/her next data segment. In which files is this statement true?
Linkage 6:	Speaker 1 says A AND B, then speaker 2 says C OR D in his/her next data segment. In which files is this statement true?
Linkage 7:	Speaker 1 says A OR B (inclusive OR), then speaker 2 says C in his/her next data segment. In which files is this statement true?
Linkage 8:	Speaker 1 says A OR B, then speaker 2 says C AND D in his/her next data segment. In which files is this statement true?
Linkage 9:	Speaker 1 says A OR B, then speaker 2 says C OR D in his/her next data segment. In which files is this statement true?
Linkage 10:	Speaker 1 says A AND (B OR C), then speaker 2 says D in his/her next data segment. In which files is this statement true? (For instance: "Speaker talks about a weekend at the beach and describes heavy winds or enormous waves, then speaker 2 also describes experiences with big waves.)
Linkage 11:	Speaker 1 says A AND (B OR C), then speaker 2 says D AND E in his/her next data segment. In which files is this statement true?
Linkage 12:	Speaker 1 says A AND (B OR C), then speaker 2 says D OR E in his/her next data segment. In which files is this statement true?
Linkage 13:	Speaker 1 says A AND (B OR C), then speaker 2 says D AND (E OR F) in his/her next data segment. In which files is this statement true?

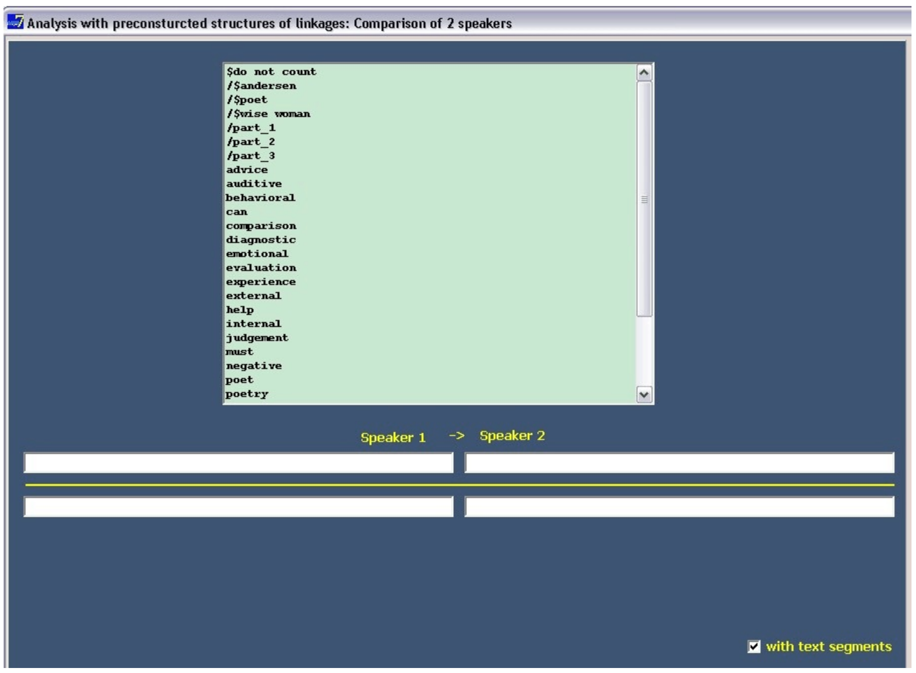
If you want to use one of these linkage hypotheses, you just click on the button in front of it (see screen shot on the previous page).

Examples

How to apply preconstructed linkages was demonstrated extensively for general linkages (see above). Therefore, we describe here only two more examples and concentrate on differences between a general linkage analysis and a comparison of two speakers.

Linkage 1

In all parts of our example project "poet" a wise woman advices a young man how to become a poet. The question is, does he follow her advices? Formulated positively and as linkage hypothesis we test the statement: "If the wise woman (speaker 1) gives an advice (conceptual code A), then the young man (speaker 2) evaluates it (conceptual code B)." We select the first linkage structure by clicking on the button in front of it and the following window will occur:



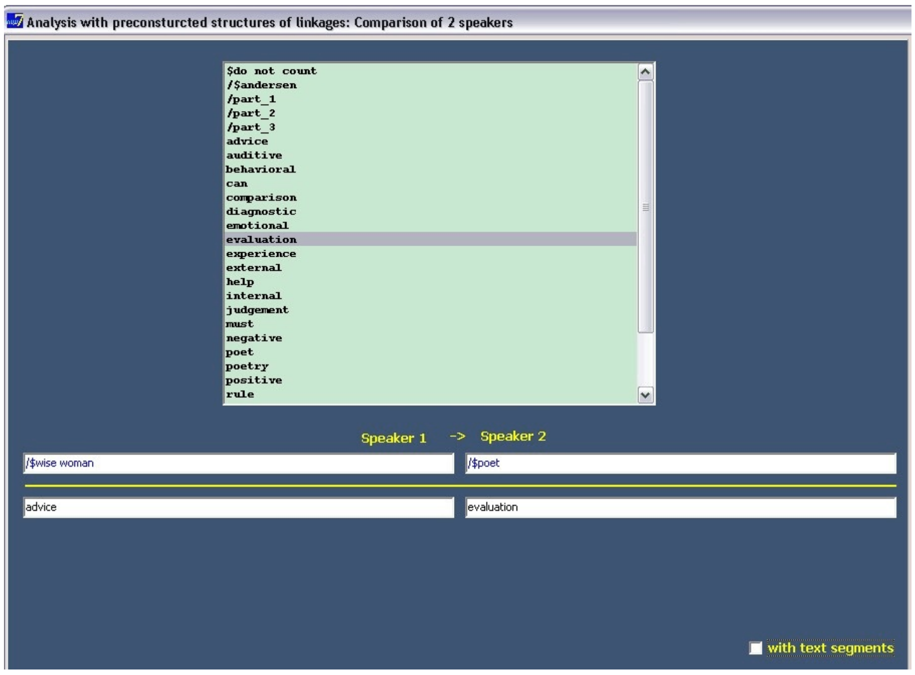
Analysis with preconsturcted structures of linkages: Comparison of 2 speakers

```
$do not count
/$andersen
/$poet
/$wise woman
/part_1
/part_2
/part_3
advice
auditive
behavioral
can
comparison
diagnostic
emotional
evaluation
experience
external
help
internal
judgement
must
negative
poet
poetry
```

Speaker 1 -> Speaker 2

☑ with text segments

The following screen shot shows how to enter this hypothesis: We click on the appropriate speaker codes in the master code list on the left side, logically speaker 1 is selected before speaker 2. The conceptual codes follow; again we begin with a code representing the content of the first speaker's talk – in our example the wise woman's advice – and select then code (here: "evaluation") for the second speaker:



Analysis with preconsturcted structures of linkages: Comparison of 2 speakers

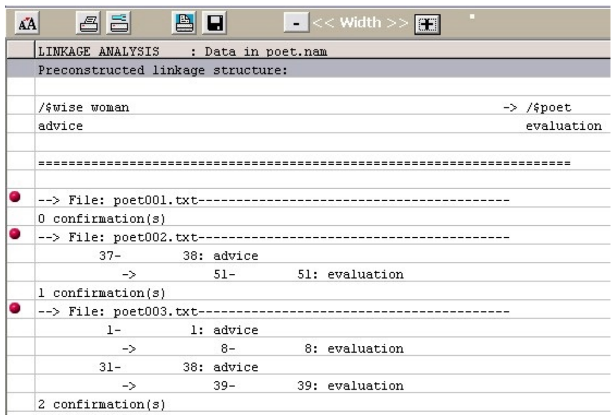
```
$do not count
/$andersen
/part_1
/part_2
/part_3
advice
auditive
behavioral
can
comparison
diagnostic
emotional
evaluation
experience
external
help
internal
judgement
must
negative
poet
poetry
positive
rule
```

Speaker 1 -> Speaker 2

Speaker 1	Speaker 2
/$wise woman	/$poet
advice	evaluation

☐ with text segments

By clicking again on the check-box we rejected the option to "with text segments." Finally we click on the button "*Continue*" and are presented with the findings:



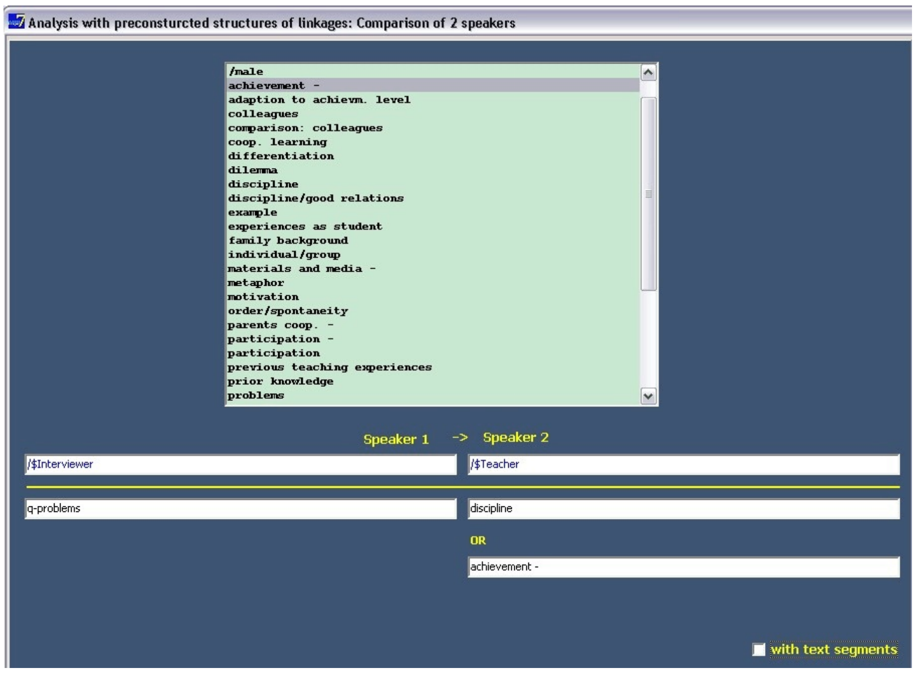
```
LINKAGE ANALYSIS     : Data in poet.nam
Preconstructed linkage structure:

/$wise woman                                              -> /$poet
advice                                                       evaluation

======================================================================

--> File: poet001.txt-----------------------------------------
0 confirmation(s)
--> File: poet002.txt-----------------------------------------
        37-        38: advice
            ->         51-        51: evaluation
1 confirmation(s)
--> File: poet003.txt-----------------------------------------
         1-         1: advice
            ->          8-         8: evaluation
        31-        38: advice
            ->         39-        39: evaluation
2 confirmation(s)
```

Whenever the wise women offers an advice, the would-be poet does not act (code: "behavioral"), but evaluates it afterwards. Whether these evaluations at least reflect the spirit of the advice should be checked by reading the critical data segments.

Linkage 3

As last example we will apply the third of these linkage structures to the project "interview" and fill in appropriate codes. Clicking on the third button opens the following window: The structure implies that we select two speaker codes (speaker 1, speaker 2), one conceptual code for speaker 1, and two conceptual codes (linked by the inclusive logical OR) for speaker 2. To construct an example we return to the files of the project "interview." Let us suppose we want to find out whether there is a relation between specific interviewer questions and answers of teachers. For instance, if the interviewer asks explicitly about problems (conceptual code: "*q-problems*") in the classroom, we assume that the interview partners talk about low discipline ("*discipline*") and/or achievement ("*achievement -*").

The following screen shot shows how to enter this hypothesis: We click on the appropriate speaker codes in the master code list on the left side, logically speaker 1 is selected before speaker 2. The conceptual codes follow; again we begin with codes (or a code) representing the content of the first speaker's talk – in our example the interviewer's question – and select then codes for the second speaker:



Analysis with preconsturcted structures of linkages: Comparison of 2 speakers

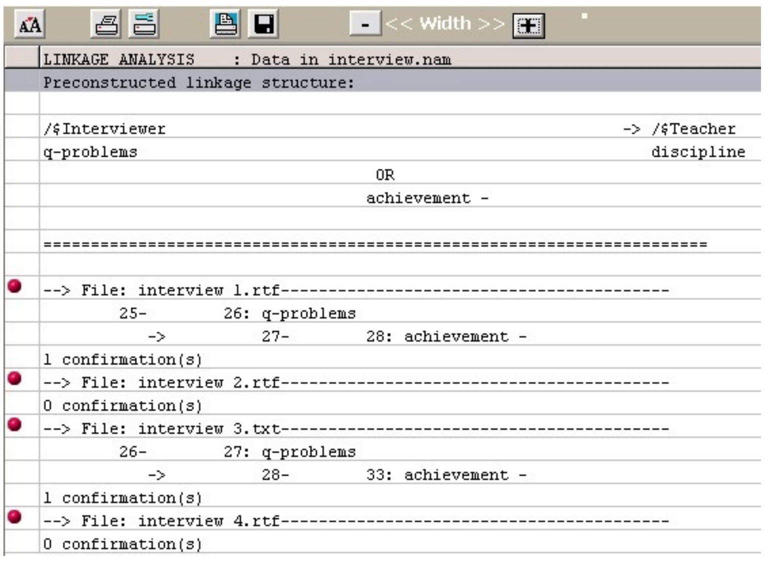
```
/male
achievement -
adaption to achievm. level
colleagues
comparison: colleagues
coop. learning
differentiation
dilemma
discipline
discipline/good relations
example
experiences as student
family background
individual/group
materials and media -
metaphor
motivation
order/spontaneity
parents coop. -
participation -
participation
previous teaching experiences
prior knowledge
problems
```

Speaker 1 -> Speaker 2

Speaker 1	Speaker 2
/$Interviewer	/$Teacher
q-problems	discipline
	OR
	achievement -

☐ with text segments

Finally we click on "*Continue*" and are presented with the following findings:



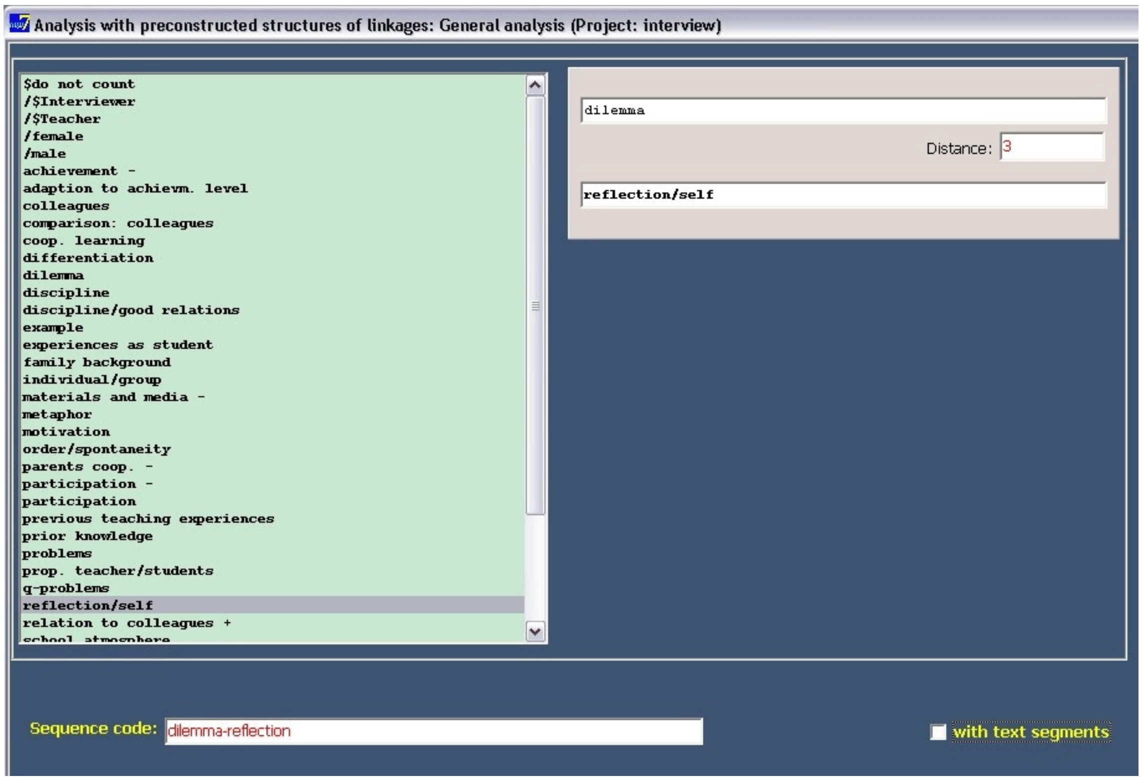
```
LINKAGE ANALYSIS    : Data in interview.nam
Preconstructed linkage structure:

/$Interviewer                                          -> /$Teacher
q-problems                                                discipline
                                  OR
                                 achievement -

====================================================================

--> File: interview 1.rtf-----------------------------------------
        25-        26: q-problems
           ->          27-        28: achievement -
1 confirmation(s)
--> File: interview 2.rtf-----------------------------------------
0 confirmation(s)
--> File: interview 3.txt-----------------------------------------
        26-        27: q-problems
           ->          28-        33: achievement -
1 confirmation(s)
--> File: interview 4.rtf-----------------------------------------
0 confirmation(s)
```

We may wonder that the interviewer asked only twice directly about problems in the classroom, and maybe even more that the teachers then worried about their students' low achievement level, but did not mention disruptions or other discipline problems. However, if we analyze the questions and answers within the full context of the interviews, we note rapidly that the teachers mostly complain about discipline without being asked explicitly to describe problems in their classrooms. This example should warn us once more against conclusions from code analyses without substantiating the findings by studying relevant segments of the original data.

8.3.2 How to create and apply sequence codes

If we look for linked codes, we generate hypotheses about linked units of meaning and attribute a common meaning to much larger data segments than we did until now. The idea suggests itself to mark this newly defined sequence of (already separately coded) data segments with a new code – that is a sequence code. Let us assume we find in our sample interviews with teachers sequences of data segments, in which a teacher describes a dilemma and then mentions his/her feelings, worries, or something else related to the own person. Now we could formulate a hypothesis according to which our interview partners relate problematic experiences in their classrooms to thoughts about themselves. There are already separate codes for both content domains: "*dilemma*" and "*reflection/self*", so what we are doing now is linking both domains to one unit of meaning: "*problem-reflection*."

During each linkage analysis AQUAD offers the opportunity to mark all positive findings by a sequence code. If we want to use this opportunity, we write a sequence code into the slot labeled "Sequence code:" blinking in red and black at the bottom of the window:


Analysis with preconstructed structures of linkages: General analysis (Project: interview)

```
$do not count
/$Interviewer
/$Teacher
/female
/male
achievement -
adaption to achievm. level
colleagues
comparison: colleagues
coop. learning
differentiation
dilemma
discipline
discipline/good relations
example
experiences as student
family background
individual/group
materials and media -
metaphor
motivation
order/spontaneity
parents coop. -
participation -
participation
previous teaching experiences
prior knowledge
problems
prop. teacher/students
q-problems
reflection/self
relation to colleagues +
school atmosphere
```

dilemma

Distance: 3

reflection/self

Sequence code: dilemma-reflection

After clicking the button "*Continue*" we will be asked additionally, whether we really want to add this code to our code files in case of positive results – and that is what AQUAD will do if we confirm the question.

Of course, sequence codes can be analyzed also quantitatively after counting their frequencies. You may add sequence codes for linkage findings in all types of linkage analyses, that is within the options "*Construct linkages*" and "*Apply preconstructed linkages*" (both versions: "*General analysis*" and "*Comparison of 2 speakers*"). In section 8.3.5, Leo Gürtler will share experiences from the work on his doctoral thesis with you and describe some possibilities how to analyze more complex problems and research questions applying sequence codes.

8.3.3 How to construct linkages

Since the hypotheses pre-formulated in AQUAD cover only a small number of possible options, the program provides you with the opportunity to formulate your own. You have to learn a few special mouse clicks, but they are quite easy to understand and to use. As already described in the previous chapter, AQUAD allows you to analyze complex relations between data. For this purpose,

- you have to draw conclusions including several different codes;
- you have to *state* that those relations do exist within your data, and then,
- you have to test whether the hypothetical statement is true or false. This is done for your own hypotheses within the module "*Linkages*", using "*Construct linkages*" from the sub-menu.

Unlike usual database programs, AQUAD compares all the elements of its database with each other on its own, applying a principle of deduction. You only need to formulate a statement, and then you can be sure that AQUAD will not forget any combination of code entries when it tests the specific conclusion. The only problem is that you need to translate your hypothesis from everyday language into a particular sequence of code names when you want to test your own hypotheses. To this end, however, you only need to be familiar with one of the windows in AQUAD and how to make use of its possibilities.

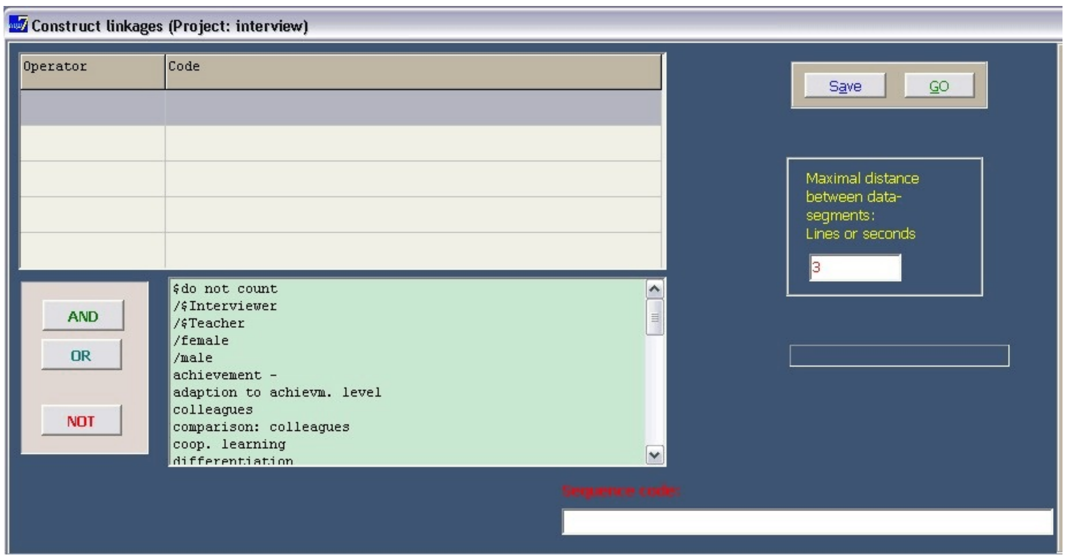
The main problem is, how to translate your own hypotheses into a language which AQUAD is able to understand? Let us look at the procedure with the help of a simple example, for which we will use the data of the interviews with beginners in the teaching profession we referred to in earlier chapters (see files "interview_1" ... "interview_4" on your program CD). We suppose there is a (sequential and/or causal) relation between descriptions of dilemmas and typical controversies particularly novice teachers experience in their classroom: How to maintain a balance between order and spontaneity or between discipline and positive social relations to students? Consequently we claim that we can find linkages between three codes: "*dilemma*", "*order/spontaneity*", and "*discipline/good relations*." Let us first formulate the hypothesis in everyday language:

The hypothesis is true, if
among the codes attached to an interview transcription
a code "*dilemma*" can be found,
and shortly afterwards a code "*order/spontaneity*" or a code "*discipline/good relations*" will show up.

Often, the problems with constructing specific hypotheses have less to do with the principles of AQUAD, which are rather simple, than with unclear conceptions of the critical relations between the coded segments in the texts. Or to say it more directly: only if a hypothesis is formulated clearly and well structured in regular English, can it be translated into a logically connected sequence of codes!

To start we click in the menu "*Linkages*" on the option "*Construct linkages*" and select the alternative "*Create*" in the window occurring afterwards:

Here is the window for the construction of your own hypotheses. You can manage it almost intuitively, nevertheless we will describe the necessary procedures step by step:


Construct linkages (Project: interview)

Operator	Code

```
$do not count
/$Interviewer
/$Teacher
/female
/male
achievement -
adaption to achievm. level
colleagues
comparison: colleagues
coop. learning
differentiation
```

Maximal distance between data-segments: Lines or seconds: 3

What we see in the lower part on the left are three buttons for entering the logical connections or operators "AND", "OR", and "NOT" as well as (in the middle) the familiar blue window containing the master code list of our project. Each linkage construction has to begin with a code that is included by "AND." The notation of linkages demands that an operator always precedes a code included in the construction. In the upper part we see on the left a box containing the real construction table (two columns: operator and code), on the right a box, where we determine the maximal distance between coded data segments. On top of the right side are two buttons to "*Save*" our construction for later application and/or edition, and to "*GO*", that is to start the analysis. At the bottom we find the meanwhile familiar entry slot for sequence codes.

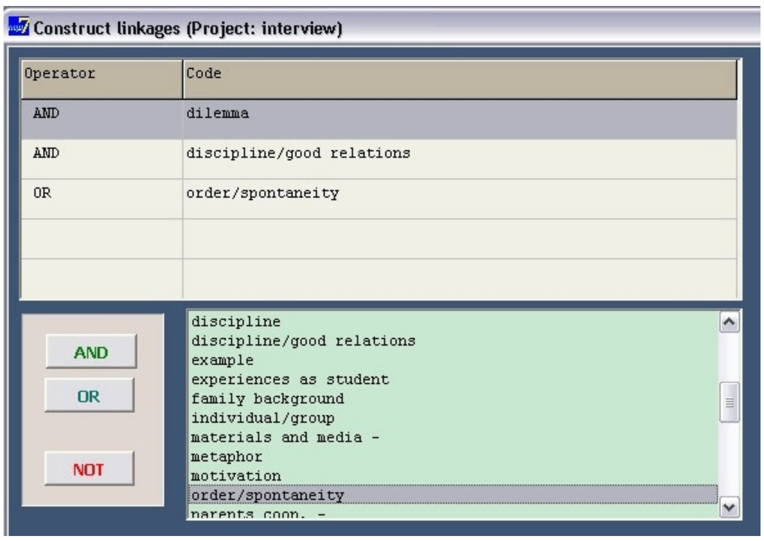
Just to make sure there are no doubts as regards the functions of the operators: If we link two or more codes by "AND", the resulting linkage hypotheses will be confirmed as true only if all of its components appear in your data within the defined maximal distance of each other. If you attach a code together with the operator "NO" the resulting linkage hypotheses will be confirmed as true only if this particular code does not occur at all in a data file.

Besides entering operators and codes we have to decide which maximal distance should be accepted between codes, or better their data segments, under the conditions of our hypothesis – except "NO"-operated codes (see above). Distances are expressed in the amount of lines between end of a preceeding and begin of a succeeding data segment in case of text documents. If we work with audio or video documents, the number you enter is automatically converted into the appropriate unit: seconds in case of audio files (1 unit -> 1 second -> 10 units of segment limitations in your code files, where positions are counted in 1/10th of seconds), frames in case of video files (1 unit -> 1 second -> 25 frames).

Now back to the linkage hypothesis we suggested in the beginning: Our hypothesis will be confirmed, whenever the program can find in a text file of the project a code "*dilemma*" AND within a distance maximally five lines from the corresponding data segments another segment coded "*order/spontaneity*" OR "*discipline/good relations.*" How do we enter operators and codes? Let us proceed step by step:

We click into the first empty cell for operators and "activate" it for an entry. At the beginning of a linkage construction, AQUAD fills the cell in this moment automatically with an "AND," whereas in the following operator cells you have to click on the appropriate operator (AND, OR, NOT). Then enter the code in the second cell in this row, the code cell: select a code from the master code list (by clicking, as usual).

In our example we select "*dilemma*" to fill the uppermost code cell. The next steps are just repetitions with variations: we fill the second operator cell with "AND," then we select "*discipline/good relations*" from the master code list. And a third time: We fill the next empty operator cell with "OR", and select the corresponding code cell with "*order/spontaneity*" from the master code file. Here is what we produced:



Operator	Code
AND	dilemma
AND	discipline/good relations
OR	order/spontaneity

```
discipline
discipline/good relations
example
experiences as student
family background
individual/group
materials and media -
metaphor
motivation
order/spontaneity
parents coop. -
```

Concluding from the number of rows in the construction table we can compose linkage hypotheses from up to five codes. Principally, we have to follow these **rules**:

- If we want to differentiate a linkage hypothesis by speakers, only one speaker code at a time must be placed into a linkage construction **and** this speaker code must be the first of all linked codes.

- A linkage construction may contain maximally two profile codes; they must be placed either as first codes or following a speaker code.

- A linkage construction may contain maximally two "NOT" operators. They function like general retrieval conditions, that is, AQUAD starts a linkage analysis by checking for codes that are expected NOT to occur in a file and does not proceed (in the actual file), if one if these codes is unexpectedly found anywhere in the file.

The sequence of codes (and connected operators) within a linkage construction may be easily modified by "drag-and-drop" movements within the construction table. Of course, this modifies also the meaning of a linkage hypothesis. Supposing we would exchange the positions of the first two elements in our example construction, we would formulate the expectation that novice teachers talking about experiences with order vs. spontaneity in their classrooms will continue by describing this situation as dilemma OR refer to a related problem of balancing discipline and positive social relations. Of course, we may also modify the critical distance between data segments. A double click into a cell or entry slot erases its content.

Finally, we have AQUAD test our linkage and click on the green "*GO*" button. Maybe you would like to save your construction first – to apply it again later, maybe with some modifications. Just click on the button "*Save*," enter a file name afterwards, and start the storage routine.

8.3.4 Linkage construction step by step

In this section we will deal with a specific application of AQUAD during the analysis of qualitative data. It is taken from a study by Carlos Marcelo Garcia, University of Seville (see Huber and Marcelo Garcia, 1990). The data presented here are only a small portion of the huge amount of qualitative and quantitative data combined in this study. Originally, version 3 of AQUAD was used for this analysis, so we have no screen shots for the last step of applying the final linkage construction.

Theoretical background

The goal of this study was to develop a more practice-oriented program for teacher education by means of analyzing the subjective experiences, especially expectations, convictions, sorrows, emotions, implicit theories, etc., of novice teachers. The first months at school are usually considered very important in the professional socialization of teachers. A series of studies have tried to identify the essential influences on novice teachers, i.e., the personal and situational factors in their professional socialization.

Jordell (1987) has developed a well structured model. He defines a first level of influence as personal. It includes the former experiences of novice teachers as students (biographical data), and also their experiences during their university education. On a second level, Jordell differentiates the influences of the teaching situation. Empirical data show that especially the students and some characteristics of classroom interaction (for instance, multi-dimensionality, simultaneity, immediacy, unpredictability) socialize the novice teachers. The third level of influence is institutional. Here, colleagues, parents and the administrative officials play important roles. In addition, the curriculum and the formal, as well as the informal rules influence the novice teachers' professional development. According to further sources in the literature, answers to the following questions might shed some light on the discrepancy between teacher training and the demands of school practice:

(1) What are the main troubles and problems of teachers during their first year in school?
(2) On which levels of professional socialization do novice teachers experience influences during their first year in school?
(3) How do novice teachers evaluate the different forms and methods of teacher training?

Data collection and analysis

In this study, 105 novice teachers participated, 46 men and 59 women. They were teaching at primary and secondary schools in the south of Spain. All subjects were asked to answer the following questions during an interview several months after the beginning of their teaching activities:

- Please describe your teaching experiences, after the first few months in school.
- What kind of difficulties have been important for you during this time?
- How do you like your school?
- How do you get along with your colleagues?
- Please describe an ordinary day in your classroom.
- What kind of problems do you have to deal with in your classroom?
- How do you feel as a teacher?
- How do you get along with your students?
- What do you think about teacher education?

Additionally, all subjects had to fill in a "teachers' conviction inventory" and a "teachers' problem inventory." The results of these quantitative instruments are not considered here. The interviews were audio-taped, transcribed with the help of a word processor, and saved on disks. According to recommendations by Miles and Huberman (1994) we defined codes for the interpretation of the transcripts. In this process we combined a deductive and an inductive strategy.

First, we chose a theoretical model appropriate for our questions. We took the model of the different levels of influence by Jordell (1987). In order to generate a provisional code repertory, we simply used the concepts of his model. With increasing understanding of our subjects' world view, we elaborated this coding system, included new features for our interpretation, and eliminated others. As the result of this first process of interpretation, we created the codes listed in the box below:

1. Personal dimension
   - ECA Experiences as student
   - EFP Experiences in teacher education
   - EDP Former teaching experiences
   - SIM Self
   - PRE Sorrows
   - APR Learning
   - CRE Convictions
   - NEF Needs for training
   - CDO Burden of teaching

2. Teaching dimension
   - 2.1 Classroom
     - RPA Ratio teachers/students
     - EFC Size (of the room)
     - EQU Equipment
     - AMB Social climate
   - 2.2 Students
     - CON Behavior
     - REN Achievement
     - COM Understanding
     - CNP Previous knowledge
     - REL Teacher-student relations
     - PAR Contribution to lessons
     - EXP Expectations
     - PRO Family
   - 2.3 Planning
     - PLA
   - 2.4 Subject matter
     - COA
   - 2.5 Interactive teaching
     - MET Methods
     - ACT Activities
     - DIS Discipline
     - MOT Motivation
     - GES Classroom management
   - 2.6 Evaluation
     - EVA

3. Institutional dimension
   - 3.1 School
     - COL Colleagues
     - MAT Materials and media
     - AMC School atmosphere
     - CUR Curriculum
     - IDE Rules
     - ORG Organization/administration
   - 3.2 Context
     - PAD Parents
     - ENT Relations to context
   - 3.3 School system
     - ADM Administration
     - LIM Limitations/rules

(Originally, the codes were defined in Spanish. In this language, the abbreviations are meaningful, of course; for instance, the first code *ECA* signifies *experiencias como alumno*.)

First, we tried then to get an overview of the most and the least frequent topics in the interviews. With a function in the module *"Retrieval"* we counted code-frequencies. As you can see in the table below, the novice teachers talked most of the time about teaching methods (MET), followed by statements about themselves (SIM), and their troubles (PRE).

EDP:	14	EFP:	10	ECE:	12	**SIM:**	**25**	**PRE:**	**23**
APR:	0	CRE:	4	NEF:	8	CDO:	4	RPA:	5
EFC:	2	EQU:	3	AMB:	2	CND:	2	REN:	4
COM:	2	CNP:	9	REL:	17	PAR:	4	EXP:	3
PRO:	1	PLA:	8	CON:	7	**MET:**	**29**	ACT:	7
DIS:	6	MOT:	12	EVA:	9	COL:	18	MAT:	15
AMC:	6	CUR:	2	IDE:	0	ORG:	2	PAD:	14
ENT:	0	ADM:	1	LIM:	4				

In the next step, we used the module *"Retrieval"* to analyze the "profile" of the codes under particular perspectives. We reasoned that we would get important information for further steps of the analysis if we would find out which other units of meaning appear within a certain maximal distance (text lines) of the segments we were interested in. In AQUAD Seven we would apply the option "*Sequence of codes*" in the sub-menu "*Coding structures*" of "*Retrieval*". In this way, we would get access to hypothetical patterns of meaning in the statements of our interview participants. We decided to define five lines as the maximum distance for the search of sequences focused around the most frequent code MET, then around PRE.

During our third step, we took a closer look at the interview transcriptions on the basis of these results. We tried to make sure that there were meaningful relationships between those text segments coded with MET and PRE, and the segments in close proximity to them. If this was not the case, we could change the criterion for the sequence (in this case: decreasing it), or eliminate unrelated codes from more refined further analyses (i.e., exclude these codes from the formulation and testing of specific linkage hypotheses).

In our case, one hypothesis could state that some of the novice teachers think a lot about teaching methods, however, they do so within a narrow perspective: segments that concern methods of teaching in their interviews are mostly related to passages in which they talk about other aspects of interactional teaching or about general dimensions of teaching; methodological topics are only rarely related to statements about the personal or institutional dimension. Another hypothesis could emphasize the relation between PRE (troubles) and EVA (evaluation), and state that some of the novice teachers think primarily about how to evaluate their students.

If the search for novice teachers with a narrow perspective concerning methodological questions should be successful, we could distinguish two types of teachers. Their narrow or wide perspective could be related to their convictions, to their problems in the classroom (for this purpose, we have data from quantitative instruments), and --perhaps after a specific new interpretation (coding) of relevant parts of the interviews -- to their relations to students and colleagues. If the hypothesis cannot be affirmed, not much was lost, since we get this result after only few minutes of work. Due to AQUAD we can save a lot of time, as well as the boring and error-prone work of looking through pages of interview transcriptions (105 interviews!) in order to search for the joint appearance of particular segments. Consider the fact that any one of the related segments could be represented by any of 19 codes (i.e., by the codes that represent the "teaching dimension").

Let us use this example to demonstrate how to translate a statement formulated in everyday language:

There are teachers in our sample whose reflections about teaching methods are connected to thoughts about other aspects of the teaching situation, but not to considerations about their own person or about school in general.

Of course, it would make sense to formulate immediately the alternative statement, i.e., that there are teachers who connect thoughts about teaching methods to personal and institutional considerations. For spatial reasons, and in order to avoid a too complex, confusing structure of the hypothesis, we will limit our efforts to the more "narrow" hypothesis. In order to avoid superfluous discriminations between numerous codes, and also to save processing time, we used the function *"Meta-codes"* within the module "*Coding*" first. We transformed all

- codes belonging to the personal dimension into PDI;
- codes belonging to the institutional dimension into IDI;
- codes belonging to the dimension of the teaching situation (MET excluded) into TDI.

In this way we get new (meta-) code files containing only four different codes. The next step is to reformulate our hypothesis according to the syntactical demands of constructing a linkage hypothesis in AQUAD, in agreement with its deductive procedures:

> There is an entry MET in the data,
> AND another segment which refers to TDI,
> but we do NOT find entries that refer to PDI and
> NOT to IDI;
> only those segments are relevant that follow a segment coded MET within a distance of five lines.

8.3.5 How to apply and create sequence codes

AQUAD offers the opportunity to mark linked data segments, which were found as result of a linkage test, by a sequence code. In the construction window (right side, lower part) you find an entry slot labeled "Sequence code:". In this section, Leo Gürtler will share experiences from the work on his doctoral thesis with you and describe some possibilities how to analyze more complex problems and research questions by creating sequence codes.

Applying the operator NOT in comparisons of speakers

Above we learned how to apply preconstructed linkages to compare speakers. You apply this option in "*Linkages*" -> "*Apply preconstructed linkages*" -> "*Comparison of 2 speakers*." These constructions link the speaker codes per default by the logical "AND." Normally this will serve well for the purpose of speaker comparisons. However there may be hypotheses that one speaker will address a particular topic, which the other speaker does exactly avoid or neglect. In this case, we have to construct linkage hypotheses on our own and apply the logical operator "NOT." We want to demonstrate a construction of this type and the usefulness of corresponding sequence codes in the following.

In a study on implicit theories of humor a relation between answers to two particular questions was assumed (Gürtler, 2004, coded open answers of students in questionnaires):

Question 7 – "*Do you think there is enough humor in your classes*?" and
Question 8 – "*If you were allowed to modify the classes, what would you do to promote humor*?"

It was interesting to find out whether students, who expressed dissatisfaction in question7, that is, who were not content with quality and/or quantity of humor in their classes, nevertheless did not suggest possible improvements as answers to question 8. This finding would indicate a difference between students, who combine dissatisfaction and tendentious passivity, and students, who see possibilities of improvement (for instance, changes of classroom climate, teaching style, teaching methods, etc.) although they are not contentas regards humor in their classes. The following sequence codes were defined and added to the code files with the goal to differentiate between these two groups of students:

```
SeqCode 1: "Q7_dissatisfaction"
      AND   speaker code:   /$question7
      AND   code:          StatusQuo: lack of humor
      NOT   code:          StatusQuo: enough humor

SeqCode 2: "Q7_satisfaction"
      AND   speaker code:   /$question7
      AND   code:          StatusQuo: enough humor
      NOT   code:          StatusQuo: lack of humor

SeqCode 2: Q8_no answer"
      AND   speaker code:   /$question8
      AND   code:          Missing Data
      OR    code:          Don't know/irrelevant

SeqCode 4: "Q8_suggestions"
      AND   speaker code:   /$question8
      AND   code:          Modifications (classes/methods)
      OR    code:          Promote climate/relations
      OR    code:          Institutional changes
```

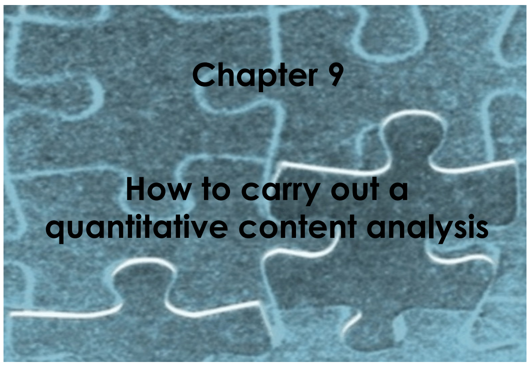
We see, each of these sequence codes is attached unambiguously to one of the questions, and positive results of a linkage analysis therefore are linked exclusively to this question (= "speaker"). In a second step additional sequence codes were created:

```
SeqCode 5: "Comp_Q7<->Q8_inconsistent1"
      AND   code:      Q7_dissatisfaction
      AND   code:      Q8_no answer

SeqCode 6: "Comp_Q7<->Q8_inconsistent2"
      AND   code:      Q7_dissatisfaction
      NOT   code:      Q8_suggestions

SeqCode 7:"Comp_Q7<->Q8_inconsistent3"
      AND   code:      Q7_satisfaction
      AND   code:      Q8_suggestions

SeqCode 8:  "Comp_Q7<->Q8_consistent1"
      AND   code:      Q7_dissatisfaction
      AND   code:      Q8_suggestions
```

Based on this system of codes, we are able to distinguish four groups of students according to their configurations of answers. Instead of comparing two speakers within a data file we compare two particular questions within a data file. There are students, who are

- dissatisfied with their situation, but do not answer (= "missing data") the question asking for possible desirable changes. This behavior is inconsistent and may indicate problems of individual competence or readiness to consider or to express any school-centered modifications.
- dissatisfied, but do not suggest any changes. This is also inconsistent, however, as distinguished from the first mentioned group, these students give answers to question 8 although not suggesting positive modifications.
- satisfied with their situation, but nevertheless present suggestions how to improve the situation. (This leads to an additional question: Are these students really satisfied? Or are these students socially highly motivated and creative?)
- dissatisfied, but able to present suggestions how to improve their school situation.

Combining sequence codes is a means to answer even complicated research questions without too much effort. Above all the possibility to have data segments retrieved together with linked codes gives perfect access to the meaning of all findings. Additionally, this example can demonstrate how to apply "speaker codes" creatively: Data segments cannot only be analyzed separately for real speakers, but also for different questions in a questionnaire or for various content domains like emotions, cognitions, actions, etc. found in a data file. This in turn may serve as perfect base of a logical minimization (see chapter 11) or discrimination of types of cases. It is a big advantage in an analysis of implicants, if you are not limited to relate isolated conceptual codes to each other in your code configurations, but are able to apply propositions (for instance, subject-predicate-object), which were operationalized by sequence coding. Usually sequence based conclusions are much more meaningful than only frequency based conclusions. We would like to refer here to Fühlau (1978; 1982), who explained meticulously the advantages and disadvantages of content analysis in de-contextualized domains.



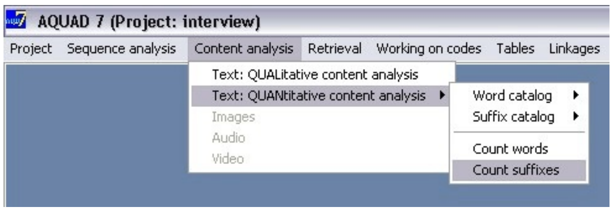
Chapter 9

How to carry out a quantitative content analysis

Leaving aside the controversy about using any quantitative means in a study where the data consist of narrative text, for some research purposes it may be useful to find out how frequently certain critical words appear. The count could tell the researcher, for instance, about the emphasis various people have placed on certain matters. Of course, any issue can be expressed with different words, and the researcher may want to conduct consecutive searches for several words all referring to the same issue. Further discussion about word frequencies can be found in Vorderer and Groeben (1987).

9.1 Counting words

Before you can count the words, applying "*Count words*" from the option "*QUANtitative analysis*" in "*Content analysis*", you have to create a catalog of words to be counted or to enter the name of an appropriate word catalog. The rules and conventions described above (see chap. 7.5) for keywords within the text are **invalid** here. When we count words, they are isolated from their context and counted just as they appear in the text. Therefore, it is a good idea to create "*word catalogs*" for counting by applying "*write a word catalog*", assembling a catalog via "*select words from text*" (or several catalogs from several texts and "*merge word catalogs*" afterwards).



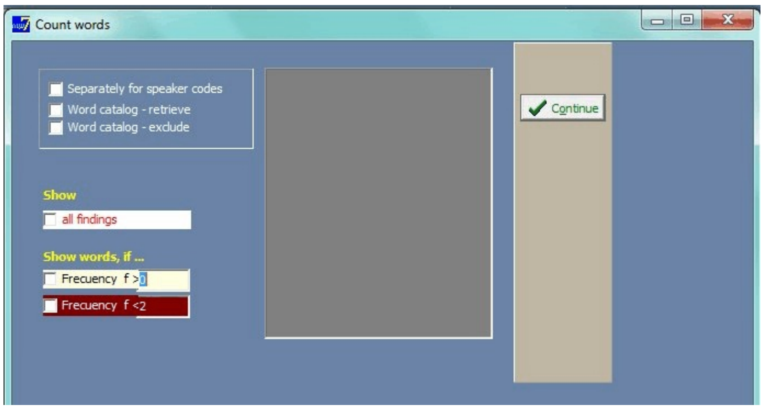
Assuming we want to count the words "no," "not," "none," and "nothing," we have to put all of them separately into our word catalog. On the other hand, the resulting frequencies will not include – as would be the case in the option "*Keywords*" – words like "connotation" or "Nottingham." Of course, singular and plural forms, case constructions, etc. must be entered separately, for instance: "student," "students," "student's," "students'," etc.

In case you had AQUAD sort alphabetically the words of some texts and you selected from these lists several word catalogs (see screen shot above), the third option permits to "*merge word catalogs*" and continue your analyses with one joint catalog.

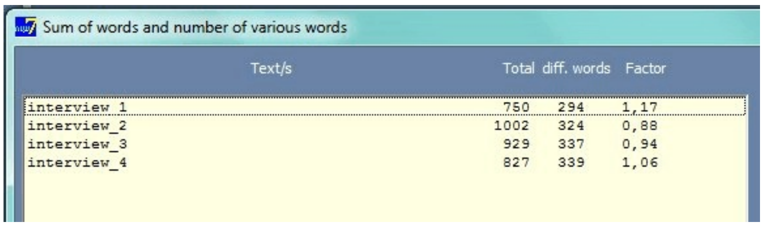
Finally, there is the possibility to "*edit a word catalog.*" This function loads an available word catalog to the screen, and you erase those elements, which are not relevant for your actual frequency count, and/or you add missing words to this catalog.

All catalogs are saved for later use. You decide about appropriate file names, and AQUAD adds automatically a particular extension ".cwo" (catalog of words). If you create word catalogs containing only related words with similar meanings or "dictionaries of meaning," you can achieve most rapidly an overview on relevant content areas in your texts.

Clicking on "*Count words*" opens a window where you may select among the text files of your project (label: "Selection") and count only words in one text or a few texts. Of course, you may accept the default setting and find word frequencies in all of your texts. Then you have several decisions to make: Do you want the words counted separately for each speaker, do you want to apply a word catalog either to count just a critical sample of words or to exclude words from counting. Additionally you may set a filter, which determines whether all findings are shown or only those between a particular maximal and minimal frequency:



Before these findings are shown, you get information (in case of deciding to see all findings) about the total number of words and the number of different words in your text. If the texts differ remarkably in their size, these data may serve as basis for calculating a correction factor for the word frequencies in each file to prevent distortions in your statistical analyses. In the following screenshot you see a difference of about 25% of the total amount of words in interview_1 vs. interview_2:


Sum of words and number of various words

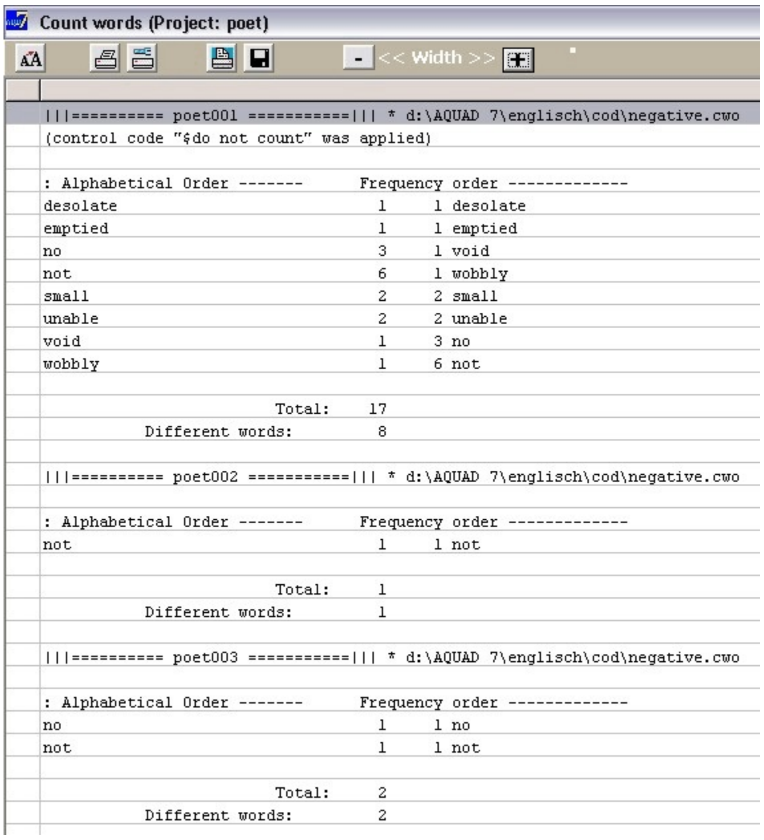
Text/s	Total	diff. words	Factor
interview_1	750	294	1,17
interview_2	1002	324	0,88
interview_3	929	337	0,94
interview_4	827	339	1,06

Again: In case the lengths of our texts vary relevantly, we have to correct these differences before we run any statistical comparisons as regards specific words. Assuming the personal pronoun "I" occurs 12 times in each of two texts, then any conclusions based on this frequency information have to take into account the length of these two texts, because probably it makes a difference if we find this frequency in a text of 700 words or in a text of 1400 words. The frequencies of relevant words should be "standardized" in the context of available data by multiplying them with this factor. Now you may also understand, why AQUAD calculates this factor *only* if we have all words in all text files counted – the resulting factors depend on this context.

The result of counting words (here in the project "Poet") is shown in a list with two columns: The left column shows the words in alphabetical order, followed by a number representing each word's frequency in the text; the right column shows the words ordered by frequency, with the number of appearances in front of each word. The following graphic (see next page) shows the result of counting some words (listed in the sample catalog "negative.cwo" on your hard drive); the comparison of part 1, 2, and 2 shows a big difference – to evaluate its relevance, we have to return to the text.

The speaker codes work also during word counting, if you mark this setting; that is, you can count words separately for different speakers, for instance in the transcription of a group discussion.

For further statistical analyses, the resulting list can be transformed into a table (CSV-format for the import to statistical packages) with the function "*Convert word frequencies into table*" from the "*Tools*"-group in the main menu.



```
|||========== poet001 ===========||| * d:\AQUAD 7\englisch\cod\negative.cwo
(control code "$do not count" was applied)

: Alphabetical Order -------      Frequency order -------------
desolate                            1     1 desolate
emptied                             1     1 emptied
no                                  3     1 void
not                                 6     1 wobbly
small                               2     2 small
unable                              2     2 unable
void                                1     3 no
wobbly                              1     6 not

                         Total:     17
           Different words:         8

|||========== poet002 ===========||| * d:\AQUAD 7\englisch\cod\negative.cwo

: Alphabetical Order -------      Frequency order -------------
not                                 1     1 not

                         Total:     1
           Different words:         1

|||========== poet003 ===========||| * d:\AQUAD 7\englisch\cod\negative.cwo

: Alphabetical Order -------      Frequency order -------------
no                                  1     1 no
not                                 1     1 not

                         Total:     2
           Different words:         2
```

The decision to count words in all or only in some texts of your project determines, whether AQUAD will produce a comparative summary of sums of words and coefficients of correction; here is an overview:

Count words

Selection of files: some files all files

▼ ▼

determination of conditions

▼ ▼

sums and coefficients
of correction

▼ ▼

ATTENTION: frequency list
Save lists for quant. analyses! print save

9.2 Counting suffixes

For specific questions, as for example when identifying cognitive styles from verbal productions of a person (see Günther, 1987), it may be useful to have a search and count algorithm which only considers suffixes (elements that are commonly added to the ends of words, such as "-ly", "-ed", "-ion", or "-ism"). You must make another word list in which you enter all the suffixes you wish to count. After that, you choose "count suffixes", which provides you with frequency results as well. If you search suffixes, do not forget to take into consideration the singular and plural forms of nouns.

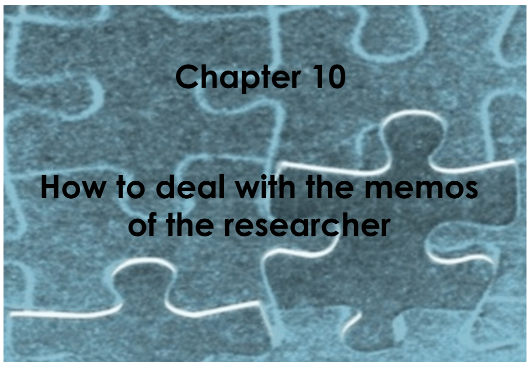
9.3 How to exclude parts of a text from word analyses

We know already that the control code "$do not count" will exclude parts of a text from quantitative analysis. This possibility is really necessary when you analyze your texts on the level of single words. Otherwise, for example, all questions or remarks of the interviewer would also be counted or retrieved – but you want to find out something about the interviewees only.

Here we show a tiny text example from a study on teacher-student interactions, which demonstrates how everything mentioned by a teacher during a particular interaction sequence is blocked for analysis. To all text segments containing the teacher's utterances the control code "$do not count" is attached. Thus they become excluded for AQUAD. Utterances of students are marked not marked by this code, thus they are open for analyses:

```
54  ...
55  TCH How many balls can we put              $do not count  55 - 56
56      on the scale additionally?
57  STD Hm?
58  TCH How many more balls could we compare?   $do not count  58 - 59
59      How many balls on each side? ...
60  STD Two -- three -- and four, or one ball ...
61  TCH And what is optimal?                    $do not count  61 - 61
62  STD Four
63  TCH Why?                                    $do not count  63 - 63
64  ...
```

Due to AQUAD's master code file, these codes are just a mouse click away from your code files; you do not have to type them again and again. If you want to remove them, this can also be done very easily by deleting them. For more details see chapter 7 on coding.



Chapter 10

How to deal with the memos of the researcher

10.1 What are memos good for?

All notes, references, cross references, ideas, tentative hypotheses, contradictions, etc., that you do not want to forget during the processes of data collection, data reduction or drawing of conclusions can be typed directly into the program. We want to recommend that you use the memo function in AQUAD extensively! Because memos are so important in processes of permanent comparison within and across texts in theory-constructing analyses, the program offers a variety of retrieval options. Thus you can construct very easily queries that support you in bringing your memos later again in contact with those texts, segments, codes, etc., which caused you to take a note.

We agree with Miles and Huberman (1994, p. 72) that collecting and analyzing texts particularly from field observations is so exciting, that coding is so strenuous that the probability is high of getting lost in the midst of highly interesting details – "... the poignant remark, the appealing personality of a key informant, the telling picture on the hallway bulletin board, the gossip after a key meeting. You find it nearly impossible to step back, to make deeper and more conceptually coherent sense of what is happening." Therefore it is indispensable to note everything that comes to your mind during an interview, during your observations, talks, etc., as well as later when you are occupied with coding and reflecting about central categories and their possible links.

Some years ago Glaser (see Glaser, 1992, pp. 108 ff.) and Strauss (see Strauss & Corbin, 1990, pp. 197 ff.) started a debate on the importance of memos in the process of qualitative analysis. The authors agree that memos should accompany the process of research from the very beginning to the final report. Memoing should not be abandoned in any of the phases of a study, because memos always are very important for reflections about the data. Neglecting memos will often becomes obvious from characteristics of the final product of a project: "A theory whose concepts lack density and/or are only loosely related." (Strauss & Corbin, 1990, p. 199)

However, Straus and Corbin try to "canonize" the process of memoing and its spontaneous, creative, maybe sometimes overwhelmingly speculative products; they distinguish between memos, code notes, theoretical notes, operational notes, diagrams, and logic diagrams – and they introduce specialized procedures for handling these different types of researcher memos. As regards these rules, we recommend pondering Glaser's objection (1992, p. 109) that it is impossible to develop a general system of characteristics in advance; such characteristics have to be observed for particular reasons during the process of "grounding" an emerging theory before they are found to emerge from the data. On the other hand, systems of critical characteristics may have important heuristic functions (cf. Huber, 1992, pp. 145 ff), but they never should serve as a complete description or as a blueprint that would enforce or guarantee a particular theoretical construction. Or formulated in other words: From the point of view of generating a "grounded theory" it is an ideal approach, if "... the analyst starts with no idea of an outline and thereby lets the concepts outline themselves through emergence." (Glaser, 1992, p. 110) However, how to cope with a situation when nothing seems to emerge? Some hints where to focus the attention or how to look at a text, a video, etc. from a different angle could be helpful in this situation.

Let us summarize and draw a conclusion: Take memos, write down all your ideas during all phases of qualitative analysis. Do not try to obey the rules of a rigid system of notation that was developed independently of your texts when memoing.

10.2 How to write memos within AQUAD

AQUAD offers two ways of access to its memo functions, depending on the module you are working with:

- There is a "Memos" option in the main menu, and
- you find a "Memo" column on the left margin of the code table in all windows in which you will do your coding.

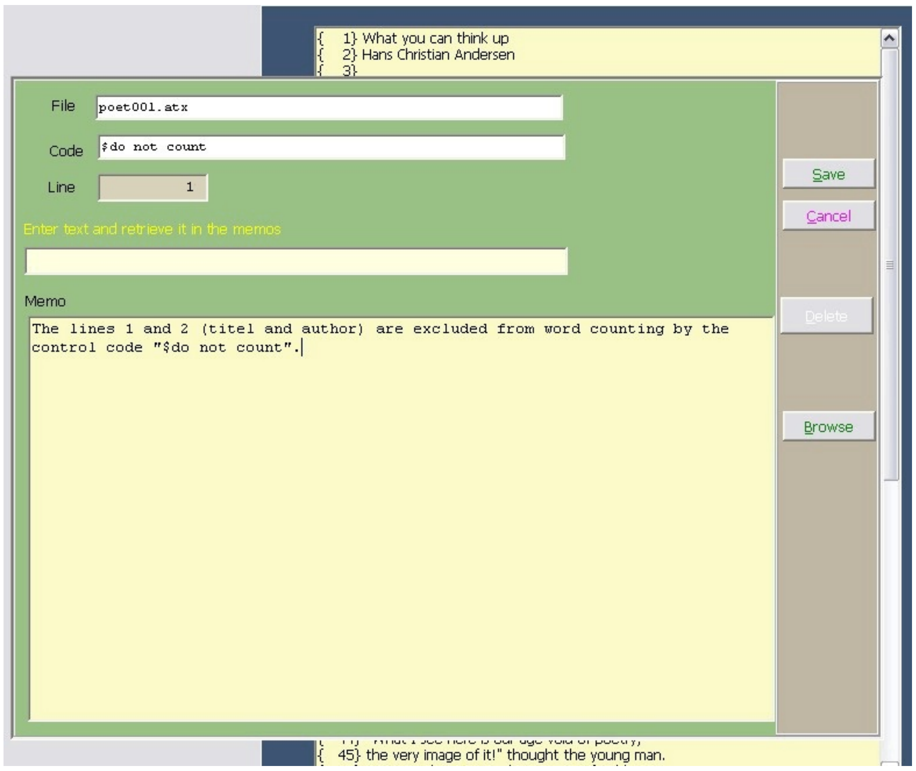
Mostly you will want to note an idea during the process of coding. Let us study memoing during coding first.

10.2.1 Memoing during coding

During coding of text files or during analyzing pictures, audio- or video-recordings you may add codes to your data files. While doing so, you may activate the memo-function whenever you need it by clicking either in the memo column ("M") on the left side of the window or on the "*Memo*" option in the main menu.

In those rows of the code table to which no memo was attached until now, you will find in the "M"-column a "0," the numbers in other rows may show that here are already attached one or more memos. Clicking on any cell in this column opens the memo window with the first of potentially many memos attached to the data segment specified in this row of the code table. Clicking on a "0" cell in the "M" column opens an empty window for memo input.

- Do not forget to "*Save*" it after you are finished with your entries.



{ 1} What you can think up
{ 2} Hans Christian Andersen
{ 3}

File: poet001.atx
Code: $do not count
Line: 1

Enter text and retrieve it in the memos

Memo
```
The lines 1 and 2 (titel and author) are excluded from word counting by the
control code "$do not count".
```

{ 45} the very image of it!" thought the young man.

Several entries are already visible in this box: the name of the data file to which this memo will be connected ("poet001.atx"), the code ("$do not count"), and the number of the line (row "1"), where you clicked into a cell of the "Memo" column. According to the logic of memoing during coding, the line or counter numbers (as margins of a data segment) can be changed during coding of texts or other types of data (audios, videos, pictures) only by attaching your memo to another of the already entered codings. More degrees of freedom are available within the "*Memo*" option in the main menu.

An additional key-word may be entered by typing it in the text field. For instance, we start developing our code systems by writing (ory pasting) "code definition" as first line of preliminary definitions of our codes, which we save as memos.

You get easy access to the list of text files and (master) codes in your project by double-clicking on the empty entry field. Two additional boxes will pop up, from which you select the appropriate items by clicking on them. However, normally you would not wish to change the number of the text file, to which your memo is related, but in case ... To close these boxes again, you just double-click on them.

All of these entries are optional, that is, you do not have to fill in these fields – however, you may really miss some markers if you want to retrieve a particular memo later. But there is additionally the possibility to search for memos using words from the memo text, characteristic words you remember to have used when you formulated a memo. That's the function of the small field labeled "Enter text and retrieve it in the memos."

What about the text of a memo? You see a relatively small window for text entry below, however, the real length of your memos is practically unlimited. You may, for instance, copy a complete text file into the memo box – if this should make any sense. In other words, there are no restrictions for writing memos. Copying text passages and copying parts of text from or to other memos is achieved by clicking into the text with the *right* mouse button.

The functions "Copy", "Cut", and "Paste," which are available from a small pup-up window, should be familiar from other WINDOWS based software. If you need further explanations, please open the general help option from the main menu and look for key words such as, "copying", "cutting", "pasting" or "clipboard".

As you are looking on the small pop-up window: For sure, you notice the option to print your memos, maybe after determining a particular lay-out for the print-out.

10.2.2 Access to memos from the main menu

Probably you have already drawn the conclusion from the descriptions in section 10.2.1 that the option "*Memos*" in the main menu gives you access to memoing functions whenever you need them – not just during coding. All explications in section 10.2.1 are valid also during all other phases of analysis, there is no principal difference. The only little difference is that no entry window is already filled in, because in this case AQUAD cannot guess to what data file and data segment you are going to relate your memo.

10.3 How to retrieve memos

This question kept us quite busy during programming AQUAD. Especially some spontaneous ideas, which should be kept in a memo, will come to your mind later in some other connection. You will remember, that this idea seemed to be extremely valuable in this or that text, for applying this or that code – however, what was it exactly?

If you remembered these details, you would have immediate access to the memo. Of course, you can read your memos one after the other – this can be done in AQUAD, too. However, some support for queries will be most helpful in this situation. When you start a query for your memos, all entries visible in the memo window will serve as criteria for retrieval.

10.3.1 Retrieving memos during coding

While you are occupied with coding a particular data file, a remarkable part of your work will consist of "permanent comparisons" (Glaser & Strauss, 1967). Whenever you are going to determine a data segment as a significant unit of meaning for which a particular code seems to be appropriate, another instance in your data files may come to your mind, which seems to be very similar or quite different. How did you code this segment – and why? So you will go back to this segment and scrutinize the code(s) attached to it. Or you remember that you used a particular code already in a similar situation. Memos preserving your considerations at the time when you coded this segment or when you applied the specific code would be very valuable now! Look for the data segment or the location where the critical code was applied, click into the corresponding cell in the "Memo" column and read the memos related to this data segment and/or code.

Finding "*Next*" and "*Previous*" memos

The buttons "*forward*" and "*backward*" are shown, if more than one single memo is attached to the same data segment; the buttons will cause what their labels are announcing: Clicking on one of these buttons opens the memo next or previous to the one you are just reading on the screen. However, this function is limited to memos attached to the line you activated by clicking into its cell in the "Memo" column, that is, these buttons are visible only if more than one memo is attached to a data segment.

The contents displayed in the profile fields of each of the memos shown do *not* serve as criteria of retrieval here, but they inform you where each of the memos belongs to.

"*Browse*" through your memos

The option "*Browse*" needs some more explanation. If you want to retrieve memos during the process of coding which were attached to other data files than the one you are occupied with at the moment, use this button! However, here a criterion is necessary to determine the retrieval process. Again, the contents displayed in the profile fields of the memos (see last paragraph above) do not serve as criteria, but give information only. Instead, you have five options to specify, which memos should be found in the retrieval run:

- All memos in the project
- Memos with keyword (enter text in the yellow entry slot labeled "enter text and retrieve it in your memos")
- Memos for specific code
- Memos for specific file
- All memos of the actual file

Then click on the button "*Browse*" and you will get all memos, which contain the criterion. If you want to copy, save or print them, just click on the right mouse button (with the cursor inside the text display) and continue according to AQUAD's suggestions.

10.3.2 Retrieving memos from the main menu

If you activate the "*Memos*" option in the main menu, you have immediate access to all memos connected with your actual project. As regards criteria of retrieval you have again five options to specify, which memos should be found in the retrieval run:

- All memos in the project
- Memos with keyword (enter text in the yellow entry slot labeled "enter text and retrieve it in your memos")
- Memos for specific code
- Memos for specific file

Click on the button "*Browse*" after entering the criterion and you will get all memos, which correspond with the criterion. If you want to copy, save or print the findings, just click on the right mouse button (with the cursor inside the text display) and continue according to AQUAD's suggestions.

Chapter 11

How to carry out an implicant analysis (Qualitative Comparative Analysis)

In addition to the option to test hypotheses about linkages between coding categories, the procedures for the logical minimization of configurations of meaning are a further special feature of AQUAD. The basic concepts of this sort of systematized qualitative meta-analysis, single-case comparison or qualitative comparative analysis are based on the work of Charles Ragin (Ragin, 1987).

Here we summarize how the module *"Implicants"* transforms case-specific configurations of meaning into truth values of a binary logic (i.e., values representing only whether a particular characteristic is given or is not given), and then combines these truth values according to the rules of Boolean algebra. In this process, one of the characteristics (or conditions) is used as the criterion for the comparison. Comparing all those cases in which this criterion is "true" (or "false") results in a reduction of these configurations to the main implicants of this criterion. In the following, we describe the steps and options when using the component *"Implicants."*

11.1 How to write data tables?

AQUAD allows the researcher to perform a logical minimization of configurations of conditions in several studies or cases directly, using quantitative and/or qualitative data. The researcher can construct a table of data (*"Create a data table"*), edit data tables, or print them. Qualitative data, for instance "meaning A marked distinctly here" (i.e.,"true"), can be directly entered into a table of truth values. However, it may be more convenient to represent even such qualitative data by means of natural numbers, for instance, by "9" representing "true". Analogically you would enter "meaning A marked weakly here" as "1" or as "false." AQUAD provides both alternatives.

Here we describe a mixed form. In order to illustrate the steps of the whole procedure, we will refer to the fictitious example of a meta-analysis: In this example ("example-imp.adt"), the relation between school achievement and class size was investigated in 35 studies. Our data shall include

qualitative values	(A/a: large/small class) and
quantitative scores	(B/b: ability level in the class above/below average;
	C/c: duration of the experiment relatively long/short;
	D/d: achievement high/low).

As distinguished from the display of truth values, we change configuration no. 5 a little in order to demonstrate a variant of the possible results. In addition, we do not enter all the results of all the 35 studies for space reasons; we only enter a selection from the data. Each of the seven observed configurations of conditions appears at least once in this selection. With real data we would not bother with differentiating these patterns of configurations. Instead, we would just enter the scores or markings of the conditions from all 35 studies without sorting them ourselves, and let the computer do the reduction to patterns of different configurations. Now, here are the various data possible in this study:

A:	Size of class according to the information "small" or "large" in the study reports 9 = small class ("true") / 1 = large class ("false")
B:	Results (standard scores) of an ability test (arith. mean of individual scores in a classroom)
C:	Time (duration in weeks)
D:	Results (standard scores) of an achievement test (arith. mean)

The first steps need no explanation: You choose the menu option "*Implicants*" and select from the sub-menu "*Write a data table*" (the complete table is shown below using the "*Edit a data table*").

The two arrows in the upper left corner direct your attention to the two parameters of your data table, which you have to determine first:

- *Number of conditions*: In its current version, the program is prepared to process between two and twelve conditions (including the criterion condition). The parameter "number of conditions" defines the number of columns of the data table. The program could take more conditions into account, but an extension of its capacity does not seem to make much sense. Think of the complexity of the results that has to be expected in this case (and the difficulties to interpret them)! In our case, achievement (condition D) serves as the criterion for comparisons, the configurations of the remaining characteristics A (size of class), B (average ability), and C (duration of observation) are examined as "conditions" of criterion D.

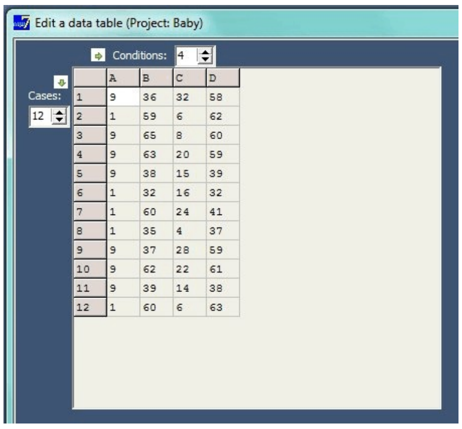
  In our case, we type the number “4" into the first entry field or click on the upper arrow in the spin button until "4" is reached.

- *Number of cases*: At this point you have to define how many findings (in our case: number of studies; usually we enter here the number of data texts) you are going to enter into the table of data. In other words, you specify the number of rows in the table. The only limit is that you need at least three cases for a meaningful comparison.

  In the example we enter the number 12, because we want to use only a selection of 12 of the 35 available studies, due to reasons of space.

The entry grid is enlarged automatically according to the number of columns and rows you determine. So you can start immediately to fill in the cells with your data. Remember, in our fictitious example we use both qualitative data (cond. A) and quantitative data (cond. B, C, and D). By clicking on the "*OK*" button you terminate the entry of data determining the structure of your table.

You find these data in the ..\cod sub-directory on your hard disk. The data table is named "example-imp.adt." You will then find this file as one of two data tables listed in the file box appearing on your screen.


Edit a data table (Project: Baby)

Conditions: 4

Cases: 12

	A	B	C	D
1	9	36	32	58
2	1	59	6	62
3	9	65	8	60
4	9	63	20	59
5	9	38	15	39
6	1	32	16	32
7	1	60	24	41
8	1	35	4	37
9	9	37	28	59
10	9	62	22	61
11	9	39	14	38
12	1	60	6	63

Just a comment as regards the meaning of the numbers in this table: We assume that we do not have more exact information about the participating classrooms than whether it is a small or a large class (condition A). For later transformation into truth values we enter great numbers for characteristics which are to be treated as "true", and we enter small numbers for characteristics to be treated as "false." Since we expect a correlation of small classes and high achievement, we attribute the number 9 to small classes (deliberate decision!), and we attach "1" to "large classes."

For the other conditions metric scores are available (B: aptitude scores; C: duration of the experiment; D: achievement scores). Those scores are entered into the table as they are. Later, the program will transform them into truth values.

It is recommended to transform all conditions into numbers, even if you have to deal exclusively with qualitative conditions that were represented originally by means of expressions of natural language (like "much," "rarely," "often," etc.). After that, you enter these numbers into a table of data (as described above).

In the example above, condition A was treated like this. In other cases we could, for instance, put the number 9 into the defined column of the data table, if we have to qualify a statement like "condition ... is given," or "person ... is assertive." In case of the reverse interpretation "condition .. is not given" or "person ... is not assertive" we could enter the number 1. You may choose almost any scores you prefer. It is only important that during the transformation of the distribution, standard values which are above or below the cut-off (see below) are set up, and that they then are substituted accordingly with capital ("true") respectively small ("false") letters.

By the way, there is a short-cut from frequency tables to data tables for logical minimization. You may apply the option "*Count codes*" from the "*Retrieval*" sub-menu to create a frequency list of those codes, which represent relevant conditions for some criterion code in your study. The criterion code has to be counted together with the other relevant codes. The resulting frequency list will be saved under any name which you enter. Just select this file again, when you activate the option "*Data table from list of frequencies*" in the sub-menu "*Implicants*".

11.2 How to convert your data in truth values

On the way from the data table to a table of truth values which we need for the logical minimization we have now to transform different types of data into the simple information "true" or "false." This is the decisive step. In order to facilitate the interpretation of the different results of the minimization process, AQUAD does not use binary digits (and the corresponding algorithms). In a term like *1001* or in the result *01**, it would be necessary each time to infer the meaning of every particular truth value from its position within the term. Instead, AQUAD uses letters as symbols for the conditions. A capital letter signifies *true*, a small one *false*.

In our example, the following configuration of conditions
A: small class ("true")
B: with high ability ("true")
C: in which only after a few weeks of the study ("false")
D: low average achievement has been found ("false")
would not be indicated by the term 1100, but by the sequence of letters ABcd.

How do we get from a table of data to terms of truth values like the ones in our example? We select either the option *"Convert data table into truth table"* in the sub-menu "*Implicants*" or we transform the original data ourselves into truth values and select "*Create a truth table*". In the latter case, the procedure is the same as for writing data tables, however, instead of numbers you have to fill in capital letters ("true") and small letters ("false") into the table's cells.

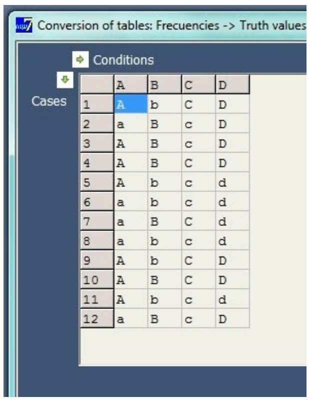
In the usual case, when you have a data table to begin with, AQUAD employs the following transformation strategy: For each of the conditions the scores in the table are first standardized (over all the cases), that is, they are transformed into standard scores with M=100, SD=10 . Then every z-score in this intermediate table is transformed into a capital letter or a small letter according to a particular criterion for *cut-off*. The default value for this cut-off criterion is

Cut-off = 50

which means that the lower 50 % of all scores in a condition are transformed into the truth value "false" and symbolized by small letters. Correspondingly, the upper 50 % of scores are reduced to the truth value "true" and symbolized by capital letters.

Of course, AQUAD allows that you change the cut-off as your research question demands. At the bottom of the window for transformation of data you see a small box displaying (in red figures) the default cut-off value. Click on the spin button and change this value (in steps of 5 %). Remember: the higher the cut-off, the more of your data are cut off as "false" (for example, in case of "70" showing in this box, the 70 % of lower data in your sample are taken as false, only the highest 30 % are accepted as "true"). Changes to the cut-off are valid only within the table for which you changed the cut-off criteria. That is, if you load a table at some later point of your analysis again, your particular cut-off settings will be employed. However, if you start to enter data for a new data table, the default cut-off value will be used – unless you adapt it to the demands of your study.

Now we are ready to start the transformation process. You select the data table to be transformed into a table of truth values from a selection window, you set or accept the cut-off value and after click on the "*OK*" button. Here is the resulting table of truth values:



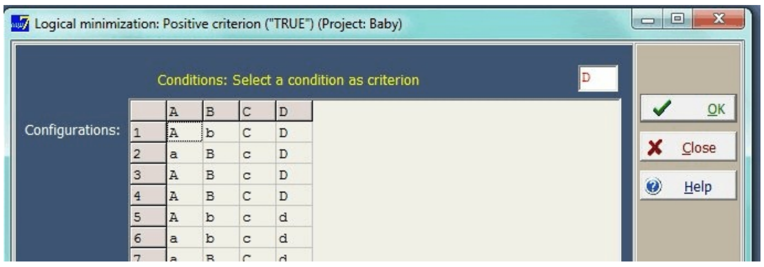
	A	B	C	D
1	A	b	C	D
2	a	B	c	D
3	A	B	c	D
4	A	B	C	D
5	A	b	c	d
6	a	b	c	d
7	a	B	C	d
8	a	b	c	d
9	A	b	C	D
10	A	B	C	D
11	A	b	c	d
12	a	B	c	D

11.3 How to examine implicants of "positive" and "negative" criteria

Having created truth values we finally can examine the "implicants." The sub-menu demands a decision between implicants for cases in which a critical condition is positive, i.e., "true" and cases in which a critical condition is negative, i.e., "false." As we will see, both examinations complement each other very meaningfully. For the moment, we select "*Analysis: positive criterion.*"

After selecting the name of the table of truth values, which we want to have minimized, the table is loaded and displayed on the screen. Now we have to select one of its conditions (columns!) as the criterion; then AQUAD will select from all truth values in our table only those combinations for which the critical condition (our criterion) is "true" – given we selected "*Analysis: positive criterion.*" We select the criterion condition by clicking into its column header.

By the way: You may wonder why there are no longer 12 cases in they table shown below, but only 7 combinations. AQUAD reduces all redundancies from the table. That is, from cases with identical truth values only one representant is left in the final table for logical minimization:


Logical minimization: Positive criterion ("TRUE") (Project: Baby)
Conditions: Select a condition as criterion
D
Configurations:
OK
Close
Help

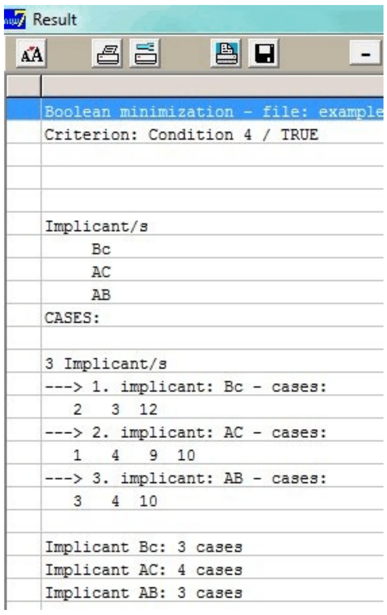
	A	B	C	D
1	A	b	C	D
2	a	B	c	D
3	A	B	c	D
4	A	B	C	D
5	A	b	c	d
6	a	b	c	d
7	a	B	C	d

In our case we select the condition labeled "D" as criterion. That is, we are looking for those configurations of conditions which may play a role in effecting above average academic achievement (criterion "D") together with the size of a class. Or formulated differently: In our example, we choose the condition "D" as the criterion, because we want to search among the various conditions for all those that play a prominent role (together with the class "size") in cases in which we observe high academic achievement.

The fact that the criterion for the minimization must always be defined shows a further special feature of AQUAD. You need not decide from the very beginning which characteristic shall be the criterion and shall therefore be entered (into the last column), when you construct a data table (or a table of truth values). This would only make sense in connection with search strategies that define independent and dependent variables. For many questions within the domain of qualitative research, however, such presumptions about causal relations of conditions are an exception. AQUAD leaves it to you to select any given category (code, characteristic, meaning) from the pool of categories, and then to search for the typical configurations of the other characteristics which appear together with the value "true" of this category (defined as criterion). Thus, Boolean minimization has primarily heuristic functions in AQUAD. Now, here is the result of our examination of configurations (see screen shot on the right): In our example the minimization provides the main implicants

Bc, AC and AB

for the criterion "D". That is, high achievement was observed in classrooms


```
Boolean minimization - file: example
Criterion: Condition 4 / TRUE


Implicant/s
     Bc
     AC
     AB
CASES:

3 Implicant/s
---> 1. implicant: Bc - cases:
   2   3  12
---> 2. implicant: AC - cases:
   1   4   9  10
---> 3. implicant: AB - cases:
   3   4  10

Implicant Bc: 3 cases
Implicant AC: 4 cases
Implicant AB: 3 cases
```

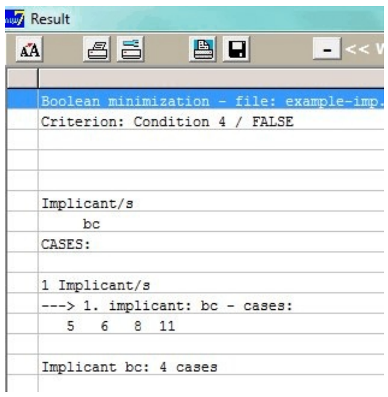
- with high aptitude level (B) *and* a relatively short period of observation (c) (Bc represents the logical relation B *and* c); or

- with low numbers of students (A) *and* long periods of observation (C); or

- with low numbers of students (A) *and* high aptitude level (B).

These implicants could be further reduced to the *essential* implicants AC and Bc. This means that in this study high achievement is observed in all those cases in which the classes are small *and* the observation lasted over a long time (AC), *or* in which the ability level of the students was high *and* the observational time was short. In the first configuration it is the ability, in the second one it is the class size that does not play an important role.

We do not get information only about configurations of conditions, here for criterion "D", but also information about groups or clusters of comparable cases. As this example also demonstrates, the main implicants often are redundant, that is, they determine overlapping groups of cases. There are cases which belong to two different configurations, namely cases 3, 4, and 10.


```
Boolean minimization - file: example-imp.
Criterion: Condition 4 / FALSE


Implicant/s
     bc
CASES:

1 Implicant/s
---> 1. implicant: bc - cases:
   5   6   8  11

Implicant bc: 4 cases
```

Now, what about the option *"Negative criterion"*? As you will assume correctly, the only difference consists in the fact that AQUAD chooses in this case those combinations of conditions for minimization in which the criterion is "false". As a result, we get the "negative conditions" of the criterion, i.e., those implicants that are related to the logical value "false" of the criterion condition.

In our example, low school achievement (d) can be observed in classes with an average level of ability and a short duration of the study(bc).

11.4 What else can you do with implicants

From the preceding examples and the general descriptions in chapters 1 and 6 you know that logical minimization is used in AQUAD mostly to *compare* the results of qualitative analyses. In particular, you can compare linkages of meaning or configurations of categories

- when you study a larger number of single cases or
- when you want to meta-analyze qualitative studies.

Commonalities and differences of cases/studies became clearly visible. During the final steps, when we want to *summarize or group findings*, to differentiate among types of texts or speakers, the process of logical minimization seems to be indispensable.

In efforts to find causal relations beyond the boundaries of case-specific conditions valid for local causality, we have to identify one category across our cases as the effect we are interested in, i.e., as the effect for which we would like to learn more about its possible causes. For instance in the already mentioned study by Marcelo (1991), the researcher became more and more interested in finding reasons for some beginning teachers' problems with classroom discipline. In the logical formulation "if ... then ..." of empirical causality, the effect category defines the "then"-part. Stated concretely, the analysis was focused on the problem "If some yet unkown things happen, then beginning teachers are confronted with discipline problems." What we want to know is the content of the "if"-part, i.e., those groups or configurations of categories that cause the critical effect. Since this if-then linkage is known as the logical relationship of "implication," we also say that the propositions within the if- part imply or implicate the proposition determining the then-part, and we call these causal propositions the effect's implicants.

To illustrate his approach, we will again take examples from Marcelo's (1991) study of beginning teachers' experiences. As was just mentioned, the author found that these teachers talked most often about discipline problems in their classrooms, although not all of them mentioned this problem. Looking for critical differences between teachers, which might also explain their classroom problems, the analysis concentrated on six categories:

A	self,
B	teacher-student-relations,
C	teaching methods,
D	discipline problems,
E	student motivation, and
F	classroom climate.

An analysis of configurations for condition D (discipline problems) as criterion resulted in three groups of implicants:

D = ABC + ACEF + abcef

From this reduction we learn that we can distinguish between three groups among those beginning teachers, who talk much about discipline problems (D). An interpretation of these groupings seems to be highly relevant for the organization of in-service training of teachers:

- *Configuration ABC*: A first group is characterized by the configuration ABC, i.e., these teachers reflect about themselves, teacher-student-relations, and teaching methods - but not about student motivation and classroom climate.
- *Configuration ACEF*: A second group, characterized by the configuration ACEF talks about self, teaching methods, student motivation and social climate - but does not seem to reflect on teacher-student-relations.
- *Configuration abcef*: The third group, typified by the configuration abcef mentions discipline problems often in their interviews, but none of the other central categories!

This application of the analysis of implicants renders clusters of cases. For theoretical and methodological reasons as well as for practical reasons we may wish to switch from the wide angle view of general findings of different configurations of conditions for a critical category to a close-up view of single cases. In other words, we may be interested in reading once more, but now concentrating on particular codes, the interview transcriptions of all those teachers that belong to one of the sub-groups experiencing discipline problems. If you study the listing of cases in the result output, you will get much stimulation for permanent comparison also on this level of analysis, opening the road from the heights of abstraction back to the lowland of case-specific formulations.

Heuristic functions

Besides support for summarizing the findings of a study, logical minimization already offers important *heuristic functions during early stages of analysis.* By analyzing the implicants or configurations of conditions for particular criterion categories we may get valuable heuristic hints how to elaborate our interpretive approach. Let us assume a study with 50-60 interviews. And let us assume additionally we had developed five important categories of interpretation during the interpretation of the first transcriptions. We name these categories here simply A, B, C, D, and E. These categories stimulated interesting, but unfortunately controversial assumptions about central messages in our data texts. Probably you would not like to continue your efforts of interpreting and coding text after text when you are in doubt about your analytic approach – until you detect after coding all of your interviews that you missed a decisive feature from the beginning!

Instead, you could take your codings of the first ten or twelve interviews, determine a particularly important category first as the "positive" criterion, then as the "negative" criterion of logical minimization and have AQUAD find the implicants of this criterion. Assuming, condition A is critical, we run the option "*Implicants*" first for all those cases, where an issue A was mentioned as very important for the speakers; i.e., we take A as "true"in these interviews. Thus we will know those configurations of conditions B, C, D, and E that go together (maybe: cause) the state "true" of condition A. The resulting configurations could be:

BD + BC + bcd.

Afterwards we activate the option "*Negative criterion*" to detect configurations of conditions B, C, D, and E in interviews where the criterion A was *never* mentioned or was characterized as *unimportant.* That is, in these interviews criterion A appears as "false". Here we find the configurations

BD + cde.

Obviously there is a contradiction. The configuration BD is found as a configuration of conditions for statements under the critical condition A = true as well as for statements under the condition a = false. After a relatively short time of interpreting only about a dozen of our 60 interview texts we would thus get valuable heuristic advice how to differentiate our coding system. Probably we missed including evaluative aspects when we applied the categories B and D. Let us assume as an example that we are dealing with interview texts about student teachers experiences during a practicum in classrooms. We coded, for instance, statements about their observations of classroom interactions between teachers and students, but we missed including in our codes whether a student teacher experienced a particular interaction as successful/positive or as unsuccessful/negative. Thus, codings referring to category B or D would be employed in most interview texts, regardless of the truth value of A. If we now take into account whether an observation of interactions was evaluated positively and therefore probably consistent with condition A or whether an interaction was evaluated negatively and therefore maybe totally inconsistent with condition A (but consistent with a), we will be able to dissolve the contradiction in a short time. We see, as a heuristic tool, configuration analysis may facilitate the task of generating adequate categories even if only a few texts are analyzed.

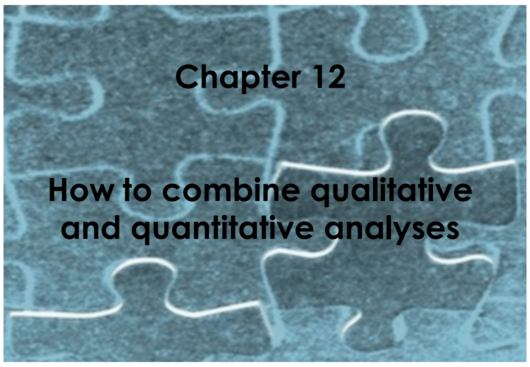
Finally, we should think of *meta-analytic approaches* opened by logical minimization. A meta-analysis does not have to be quantitive in nature, as Glass, McGaw & Smith (1981) state as "not deniable." Depending on the nature of data, qualitative or quantitative meta-analyses are possible. Both approaches must not differ in the strictness and the systematicness of the comparison. With an instrument like AQUAD this demand can be fulfilled.

However, the sources of errors for comparisons of studies as summarized by Jackson (1980) in a disillusioning way, cannot be eliminated, even with the support of a computer. As emphasized above, it is the researcher, who controls the analysis and not the computer: the computer is only a useful tool. If only a very small percentage of those authors, who establish their work on summaries of other studies discuss these findings critically, then the tool is useless. With the support of a computer, however, already the original authors should become aware of the fact that they only picked and chose specific configurations of conditions from other researchers' studies or case analyses, or that they overlooked contradictory configurations. Jackson's (1980) charge that most of the comparisons are done a lot less strictly at the moment becomes particularly severe, if we consider the availability of software like AQUAD.

11.5 Functions of implicants in the process of theory building

How should researchers conceive of implicants as results of configuration analysis? Do they serve as evidence to proof or to reject underlying theories, the blueprints for reconstructing other people's views of the world, or to establish theories emerging in the process of qualitative analysis? The answer is a maybe puzzling "neither ... nor." As here is no space for elaborate methodological considerations, Ragin's (1987) explanations of the dialogue of evidence and ideas in Boolean configuration analysis may be interesting for further reading:

> The Boolean approach to qualitative comparison ... is a middle road between two extremes, variable-oriented and case-oriented approaches – it is a middle road between generality and complexity. It allows investigators both to digest many cases and to assess causal complexity (Ragin, 1987, p. 168).



Chapter 12

How to combine qualitative and quantitative analyses

12.1 Qualitative versus quantitative methods of research?

During the 80s of the last century debates about "quality" and "quantity" were a running topic in the issues of the "Educational Researcher," one of the American Educational Research Association's journals dedicated to general questions of educational research. Mostly the contributions referred to each other and kept the readers excited like a serialized novel. Four types of texts could be distinguished:

(1) Texts trying to corroborate an unbridgeable contradiction of quality and quantity in educational research (Smith, 1983; Smith & Heshusius, 1986);

(2) texts expressing consternation about the process of polarization and trying to shed some light on fundamental misunderstandings (Shulman, 1981; Phillips, 1983; Tuthill & Ashton, 1983; Howe, 1985, 1988; Firestone, 1987) or to develop alternative interpretations (Eisner, 1981, 1983);

(3) texts reacting with complementary strategies of research to the thesis of incompatibility (Miles & Huberman, 1984, 2. Aufl. 1994; Huberman, 1987), and

(4) texts concentrating on qualitative approaches and trying to contribute to a systematic of qualitative research (Fetterman, 1982, 1988; Jacob, 1988).

However, a discussion of goals and conditions of qualitative research particularly as compared to quantitative approaches did not get going very well, at least not in the domain of psychological questions and research. Some of the articles gave the impression that the authors did not prefer to reflect advantages and disadvantages of qualitative and quantitative approaches and to promote complementary applications, but to keep going a process of dichotomization. Even within thematically specialized scientific communities the formation of opposing "methodological camps" could be observed. Unfortunately, this walling as regards types of approved data often seemed to affect also the goals of research, because methods are – so the etymological roots – particular roads to particular destinations, and a decision against a specific road also excludes specific goals. Sometimes this development brought it about that questions and findings of the "others" were not even taken note of, nothing to say about serious discussions. The exchange of labels seemed to be sufficient for some opponents, for instance labels like "soft" vs. "hard" research – to quote just one of the more friendly examples.

In our opinion these differing methodological orientations do not define incompatible contradictions. Numerous convincing examples from empirical social research meanwhile demonstrate that it is possible to study the subjective perspectives of people and at the same time reduce this information objectively. Each scientific approach constructs a specific perspective, however, some methods give a stronger impression than others that they are able just to find the world as it is. Nevertheless, even today the strategy of delimitation is still alive. Quite pronouncedly have Smith and Heshusius (1986, S. 11) formulated this position in the old days when they stated that it is impossible from an epistemological point of view to talk to each other, if a quantitative and a qualitative researcher disagree on some issues.

However, what all scientific methods have in common is the demand to plan and realize studies systematically, and to generate reliable and valid results. No matter whether frequencies of behavior are counted or behaviors are interpreted, a researcher has to bring out the structures of meaning he/she has constructed – either when selecting test items, experimental conditions, etc. or developing a category system of interpretation – to guarantee criteria of reliability and validity. No doubt, there are exact rules in the realm of quantitative methodology, whereas some proponents of qualitative orientations seem to question the necessity to disclose the system of their personal processes of construction so that they themselves or third persons would be able to check these processes and to repeat them if necessary. However, this is an indispensable claim.

Unfortunately most textbooks of qualitative methodology describe in great detail various approaches to collect data, but they pay little attention to problems and procedures of data analysis. Sieber (after Miles, 1983) found already more than 20 years ago that leading textbooks devoted only 5%-10% of their content to data analysis; today this proportion seems to be better, but it is still far from being balanced. Therefore, development of software for the analysis of qualitative data means progress also in this respect. Computers as tools of data analysis support reduction and analysis of qualitative data in a way that we can easily check, reconstruct, and above all communicate process and results of a study. This could get going a dialog between members of both camps, maybe in a sense as Miles and Huberman (1984, p. 23) described their own position. What mattered for them is that researchers proceed systematically and aim at consensus in terms of content and methods. The authors themselves described in the first edition of their book their own position as a "realistic" one somewhere between "right wing" qualitative research and "weakly" positivism, because they recommend to pay attention to reality in interpretative approaches and, on the other hand, to ask what one thinks that quantitative "data" are and which role one's own perspectives play in the process of attributing meaning to these data.

Here again a reference to the role of computer-assisted data analysis, now formulated differently: Computers may be helpful in qualitative research, because they support communication, reconstruction, and control of research processes. Thus computers can assist in structuring naturalistic approaches and systematize even markedly interpretative approaches. At the same time computers support a pragmatic, but nonetheless very important goal: simplifying complex, time consuming, and sometimes boring routines.

Eisner (1981, p. 6), however, warned against reducing possibilities and purposes of qualitative approaches by structuring their methods in this way. Instead, he demanded to put more weight on "artistic-intuitive" processes:

> Manifest behavior is treated primarily as a cue, a springboard to get someplace else. That is to "indwell," to empathise; that is, to imaginatively participate in the experience of another. ... The difference between the two is subtle but important. In the former, observables are used in a kind of statistical fashion; one intuitively (or statistically) estimates the probability that this behavior means one particular thing or another. There is no real need for empathy. The latter banks on the observer's ability to imaginatively project himself into the life of another in order to know what that person is experiencing. ... Thus, a major focus in artistic approaches to research is the meanings and experiences of the people who function in the cultural web. (Eisner, 1981, p.6)

Explanations like these are easily (and readily) misunderstood, if readers assume that facts should be replaced by fiction. On the contrary, Eisner (1981) describes very convincingly the potential of mutually complementary perspectives in qualitative and quantitative research. To present an example, which may exceed what aficionados of quantitative methods would take for scientifically acceptable, we refer to literature: What do we learn about the situation of mentally handicapped children from quantitative studies – as compared to what we learn when reading Peter Härtling's novel about little Hirbel? But what meaning has empathy into this child's fate without a reliable base for conclusions in other cases? Quantity per se is meaningless, quality per se has no consequences. Particularly research in "applied disciplines" like education or educational psychology relies on intelligent complementation of qualitative approaches by quantitative findings, and vice versa.

Therefore, one may only wonder when reading that the conclusion by Smith and Heshusius (1986) quoted above is meant to close down the conversation, that is, to represent "the end of the quantitative-qualitative debate among educational researchers." However, we should not leave it at wondering, because the consequences of this dichotomization are far too serious in social sciences. Instead, we should try to bridge the gap. One of the possible solutions is on the one hand to elaborate qualitatively on the meaning of quantitative findings, on the other hand to systematize the procedures, provide documentary evidence for interpretative processes, and develop (quantitatively based) order of findings in qualitative approaches. Insofar there is no "good" or "bad," no "hard" or "soft" research, but besides the quality of the theoretical base of a study there is only methodological rigor or sloppiness – in both approaches.

Happily, we find in discussions about the options of qualitative content analysis and in developments of concrete procedures many suggestions how to take systematically into account the complementary relation of quality and quantity. For instance, Mayring suggested already in 1988 a general model of analytical phases to optimize the relation of qualitative and quantitative approaches. According to this model, content analysis starts from qualitatively determining the research questions, critical concepts or categories, and instruments of research. Depending on goal and/or topic of an analysis, the instruments may then be applied with support of quantitative procedures (for instance, counting frequencies of words, running cluster analyses of critical concepts, etc.). Final conclusions based on the findings again demand qualitative-interpretative activities. Villar and Marcelo (1992) their own studies as data base and demonstrate und discuss various possibilities how to combine qualitative and quantitative methods during the phases of designing a study, collecting data, and analyzing data.

This schema promises to re-activate a long-standing demand by Lisch and Kriz (1978, p. 46) not to eliminate the subjective aspects of a content analysis, but to explicate them:

> If a content analyst becomes aware ... of his/her decisions in reconstructing the social reality, and if he/she informs his/her scientific community about these decisions ... we become able to comprehend inter-subjectively and to check the framework of interpretation, and thus to answer the question how to reconcile the content-analytical results with an action-relevant theory. ... In this way the specific experiences of a content analyst with a text can be communicated, reconstructed, and thus learned again by others as far as possible. (Lisch & Kriz, 1978, p. 46)

Methodological developments during the last years as well as concrete studies in the social sciences show – at least outside of Germany – that research is more and more oriented at the challenges of the specific research question and less at a general frame of reference of a particular methodological approach.

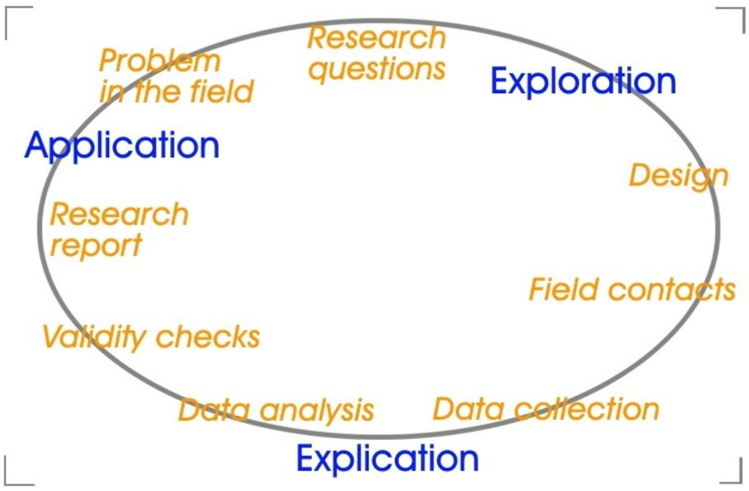
By alternatively using qualitative and quantitative procedures according to a strategy of "mixed methodology" (Mathison, 1988; Tashakkori & Teddlie, 1998) we can expect to succeed in elaborating also on those aspects of a complex research problem, which we would have been unable to analyze with a limited repertory of methods only – if we had been able at all to perceive these aspects, blinkered as we would have been without a wider methodological orientation. Recently a handbook on "mixed methods" (Tashakkori & Teddlie, 2003) was published. This event nurses hope that this approach will unfold fruitfully in the practice of social and behavioral research.

12.2 "Mixed methods" – slogan or strategy of research?

As we have seen, qualitative and quantitative methods do not exclude each other in a research project, but are often used in complementary relation. Of course, it is true that decisions about methods depend on the research question we want to answer in an investigation. Supposing we want to know how many people will vote for a specific party in elections next year or how many people think about buying a specific product during the next six months, we are interested in precise and reliable figures as results. If, however, we want to find out why people wish to buy a particular product or why they prefer a competing article, we are interested in differentiated evaluations from the subjective points of view of potential customers.

Scrutinizing the design of both studies outlined above we see that the first case results in some percentage figures complemented by data about the range of confidence, that is, quantitative findings, while the second case reports a summary of attributed qualities, but we will become aware of qualitative and quantitative techniques in both types of investigation. However, usually neither the role of qualitative procedures is discussed in the first case, nor the role of quantitative elements in the second case.

Investigations are stimulated by some problem, in the domain of social research mostly a problem of everyday life or a critical situation, event, etc. in specific social areas like, for instance, effects of TV on young people or violence in schools. The initially more ore less fuzzy perception of a problem gains clarity and sharpness once we start to formulate research questions. In this phase of an investigation we do not decide about methods – neither quantitative nor qualitative ones, but try at first to develop a most precise version of our research question(s). The activities necessary in ths phase are by no means of quantitative nature! During further explorations of the problem domain under the conditions of a quantitative study we will elaborate hypotheses, develop a research design, and select methods appropriate to test our hypotheses quantitatively. In case of qualitative studies we concentrate on possible methods how to gain more and more differentiated information on the critical domain. With this information we should then be able to formulate scientific hypotheses or a preliminary theory of the problem under study. Whether we apply tests, rating scales, questionnaires, interviews, copies of diaries, etc. as data base – in any case we need access to the social field, were we have to collect our data. Establishing field contacts, building confidence and readiness for participation are elementary research activities beyond any polarizing conceptions of qualitative versus quantitative methods.



Without further discussion we want to mention that the construction of quantitative instruments, which are used in quantitative studies during the following phase to collect data, of course, was not based exclusively on quantitative methods. During the development even of highly standardized tests there was a point when somebody must have had an idea and made a decision, that certain test items would assess exactly what the instrument was meant to test. In general we have to take into account that the researchers themselves and the roles, they attribute themselves due to their particular epistemological orientation determine, which research activities will be understood as important and scrutinized in a given research context. Maxwell (2002) underlines the influences of researchers' personal characteristics on the one hand, for example prior experiences of researchers, their convictions, personal goals, and on the other hand personal relations of researchers to the field, above all their contacts to potential subjects. I depends on identity and world views of researchers as well as on their relations to subjects how they conceptualize the details of a study and how much they become engaged in it. The same is true, vice versa, for the subjects.

Independent of the concrete methodological approach follows now the phase, during which the selected instruments are applied to collect data and analyze them. The validity of findings has to be checked. This is a process, in which qualitative and quantitative methods complement each other very well – of course, depending on the criterion of validity. Finally the researcher(s) has(have) to write a report. Subsequently the readers and/or the researcher will reflect how the results may contribute to solve the problem studied. In this phase again – if it happens at all – controversies about qualitative versus quantitative methods are not a topic of debates.

The question remains, whether we can combine systematically qualitative and quantitative methods on the concrete level of specific phases, above all during data collection and data analysis, even if we agree with the idea of complementary relations of qualitative and quantitative methodologies on the abstract level of a cycle of research phases. Subsequently we will see that combinations of this kind happen always happen implicitly in the process of qualitative investigations, and are recommended explicitly as essential characteristic of designing, realizing and controlling scientific studies.

12.2.1 Implicit combinations of methods

In chapter 6 we described the interaction of strategies of differentiation and generalization according to Shelly and Sibert (1992) as a cyclic process of inductive and deductive reasoning. Mayring (2001) states as common goal of both processes to generate systematically categories and ascribe them to data segments. He concludes: "If we work systematically in this way with categories, we may assume to conceive of these attributions as 'data' and continue the work quantitatively in a second analytic step." (Section [16])

That is, the findings of a first qualitative-interpretative reduction of initial data are, for instance, counted and ordered by frequency, expressed in percentages, ordered on ordinal scales (for instance, many – intermediate – few), compared across data segments and/or cases by means of statistical calculations, etc. Whether these new findings contribute anything to answer the research question must be evaluated qualitatively in a third step (Mayring, 2001, [17]).

12.2.2 Explicit combinations of methods

According to Villar and Marcelo (1992) the progress in social research is based on applying various methods, of course, depending on the problem under study – and not adapting questions and empirical access to the possibilities of a specific methodological approach. The authors refer to a suggestion of Greene, Caracelli & Graham (1989) to collect data within the framework of a "mixed-method" design. Thus, as Villar and Marcelo (1992) continue to explain, the research process is opened for the demand to "triangulate, " that is at least to promote the validity of a study by integrating different methods, sources of data, and researchers (Mathison, 1988).

Villar & Marcelo (1992) unfold this idea in suggestions for core phases of empirical research, namely access to the field, methods of data collection, and methods of data analysis. They describe concrete procedures, which they used themselves in their studies to profit from the advantages of combining qualitative and quantitative methods:

- Combinations of methods in the phase of access to the field
  If it is important to study a representative sample drawn from a particular population (for instance, "the" elementary school teachers; "the" math teachers in senior high schools, etc.), a combination of selecting subjects based on quantitative procedures (for instance, selection of a specific percentage of subjects by chance) and checking the representativity based on qualitative characteristics of the sample (for instance, proportion of male/female, old/young subjects) is used frequently. As example Villar and Marcelo (1992) describe a study on professional socialization of novice teachers. The participants in this study were selected from a list of teachers by random numbers. Then the composition of the resulting sample was checked regarding gender of subjects, teaching specialization, type of school, geographical location of the school, that is, as regards important qualitative characteristics.

  How about "mixed methods" if we want to select a number of single cases, but not a more or less numerous sample? Combining qualitative and quantitative methods will raise the precision of selection and thus contribute both to save resources and to advance the validity of results. What we do in this instance is "theoretical sampling:" single cases are not chosen at random, but to the purpose to study persons, events, situations, etc. that are for instance, critical as regards the research question, unique or extreme or particularly informing, because usually they are hard to access (Yin, 1989). Quantitative diagnostic instruments may be advantageous: Supposing we intend to study (qualitatively) subjective effects and individual learning processes under the condition of a specific teaching method, and assuming differences between low and high achieving students, it would be most promising to administer appropriate achievement tests and/or intelligence tests and then to pick out precisely those few subjects with low/high prerequisites which we are able only – for practical reasons – to interview or observe.

- Combination of methods in the phase of data collection

  The examples in the preceding section show additionaly, as expressed in the graphic above, that decisions in specific sections of a research process are entwined. For example, if we decide during the design of a study to use specific empirical methods and instruments, there are not many degrees of freedom left during the phase of data collection – unless we enter a backward loop and revise our former decision.

Of course, it depends on the research question, which methods and which combinations are most suitable. As regards qualitative research in the area of teaching/learning processes in teacher education Villar and Marcelo (1992) list as frequently used methods of data collection participant observation, in-depth interviews, and classroom or teaching diaries. In studies oriented at approaches of product or effect research in this area mostly non-participatory observation (for instance, via video) and preconstructed category systems are used to find the frequencies of critical phenomena.

As an example for one of many possible combinations the authors describe a quantitative study of student teachers' practical training at the University of Seville (Villar, 1981), which mixed non-participatory observation (sound recordings of student teachers' classes), interviews about subjective impressions, and rating scales of opinions, problems, and social climate in the university seminar. This combination permits mutual completion,that is, adding more *depth* to the information of various findings as well as methodological *triangulation* by comparing data from interviews and rating scales (see below and Mayring, 2001).

- Combination of methods in the phase of data analysis
  Qualitative and quantitative techniques of analysis may be combined both during the analysis of data produced by a specific method, for instance, interview data, and during the analysis of data from different methods, for instance, a combination of interview and questionnaire.

  An example of the first case was already described in detail in chapter 12, section 12.2. A simple analysis of frequencies supported the identification of critical statements in a large number of interviews. Afterwards, possible relations between these statements were checked by analyses of correlation (see Villar & Marcelo, 1992).

  The second possibility of combination is typical for triangulation approaches. On the one hand, under a qualitative perspective we can profit from specific contributions of various analytical methods (see Medina, Feliz, Domínguez & Pérez, 2002; see also below), on the other hand we can apply quantitative techniques like dimensional analyses or analyses of regression and identify or confirm much more easily cardinal patterns of argumentation. Accentuating the quantitative approach, Schweizer (2004) uses the first chapter of Cervantes' "Don Quixote" as an example and describes in great detail how a quantitative analysis of vocabulary/word forms and evaluation, which role vocabulary and particular word forms play in fostering text understanding, may assist in elaborating important hermeneutic findings (for instance, insights into the relevance of reading for individual development as well as opening access to motives and interests of the real author by "looking behind" his fictitiously created world).

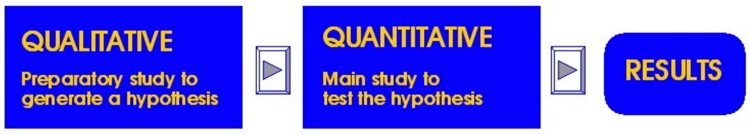
  During the last phase of data analysis, which we label in the context of AQUAD according to Ragin (1987) as "analysis of implicants" or "logical minimization" (see chap. 13), we try to generalize our interpretations beyond a single set of data or cases. If we ask, what is general or typical in a particular case, the answer will consist of a list of relevant characteristics this case has in common with a class of similar cases. If we ask the other way round what makes a case unique, the answer will stress those features that distinguish this case from others. We are only satisfied with this categorization of cases, if each of its classes or categories contains nothing but cases most resembling each other and therefore most distinguished from cases in other categories. This process of categorization leads to an order of given cases. Additionally we expect it to represent a typology, which allows implicitly or explicitly to clarify relations, to link various experiences, establish expectations and formulate prognoses. You eill find examples in chapter 13 and in Huber (2001). Mayring (2001) conceives of "identifying a single case as typical of a particular content area ..." as "... a first quantifying step of generalization." [18]

12.2.3 Combinations of methods in research designs

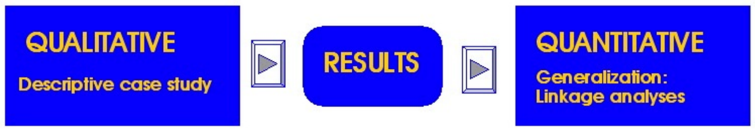
On the level of design of empirical studies Mayring (2001) suggested four models how to combine qualitative and quantitative methods. He calls them the model of pre-studies, the model of generalization, the model of profoundization, and the model of triangulation. We sort these models into two categories and complement them by a third type of combinations:

(1) Macro-sequences of combinations:

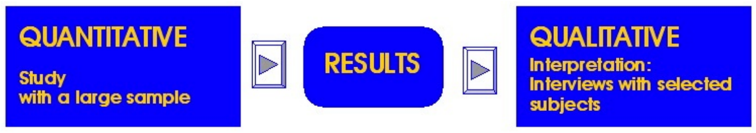
- The model of preparatory studies:
  This model represents an approach, which was recommended in former years as sort of division of labor according to a traditional conception of empirical science: "... qualitative methods are used for explorative studies above all in the beginning of a research project. They have to clarify the field of research, differentiate problems and questions, establish preliminary hypotheses and prove the practicability of research methods" (Krapp, Hofer & Prell, 1982, p. 130). This conception does not justice to the complementary function of qualitative approaches in empirical research, but concedes only a function of assistance for "real", that is, quantitative research. Nevertheless this model demonstrates that quantitative studies can rarely neglect qualitative foundations. Schematically presented the relation of qualitative and quantitative methods in the model of preparatory studies is the following (cf. Mayring, 2001 [21]):


QUALITATIVE
Preparatory study to generate a hypothesis
QUANTITATIVE
Main study to test the hypothesis
RESULTS

- The model of generalization:
  The project starts with a genuine, independent qualitative study that leads to results, which are important already within their original frame of reference. Taking into account the relatively small data base of qualitative studies, however, often researchers do not feel justified – particularly in the phase of application – do draw conclusions implying maybe far-reaching modifications of the social situation studied. They prefer to add quantitative studies to probe the generalizability of t their findings and to ascertain quantitatively probabilities and effect sizes. As an example Mayring (2001) describes a case study followed by a quantitative study, which was designed to check linkages in a representative sample of subjects:



- The model of profoundization:
  This model reverses the sequence of design components, that is, a representative quantitative study is followed by a small-scale, case-centered qualitative study. The qualitative findings are meant to assist in a better understanding of the meaning of some quantitative findings.

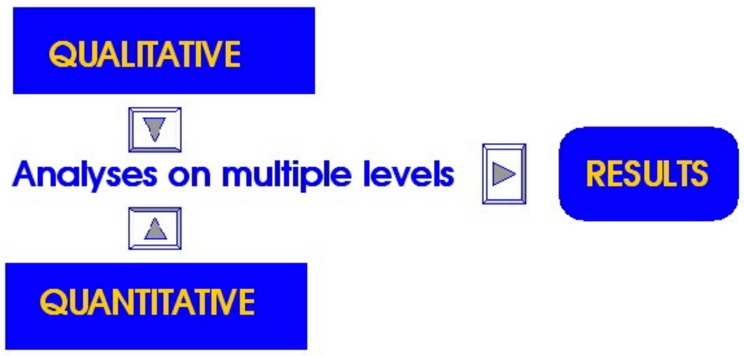

QUANTITATIVE
Study with a large sample
RESULTS
QUALITATIVE
Interpretation: Interviews with selected subjects

Many studies on problems of youth run by J. Held and his team at the Department of Educational Psychology at the University of Tübingen are designed according to this model (see Held, 1994; Kiegelmann, Huber, Held & Ertel, 2000). In a project on the topic of "Political orientations of young workers" Held, Horn and Marvakis (1996) tried to find the reasons, why young people had adopted their specific orientations. Therefore they asked a large number of young people to answer a questionnaire, and in a second step some youth were selected for interviews, which helped to understand the quantitative findings more profoundly.

In this study as well as in a former investigation by Held and Marvakis (1992), an additional effect of the model of profoundization becomes obvious: This model is apt to link explication and application. The two-step design implicated to stay in contact with the subjects even after they had filled in their questionnaires. What followed were feedback sessions in schools and companies. The participants were informed about preliminary results, which stimulated group discussions and motivated to participate in an additional interview. Thus, the quantitative instrument became a means of potential change. The process of research was during this phase at the same time a process of education.

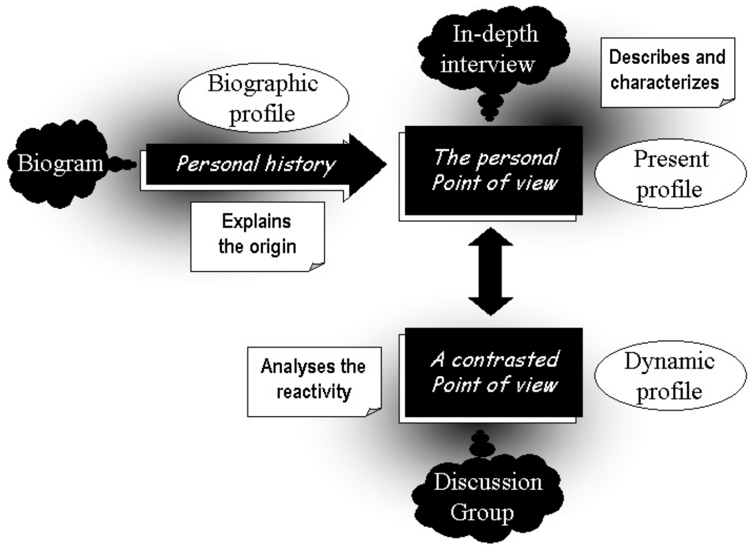
(2) Simultaneous use of qualitative and quantitative methods

– The model of triangulation:
This approach makes use of different methods simultaneously and with equal rights. What matters is that we are able to compare varying data, which were collected as answers to the research question under the conditions of different methods, and to determine the overlap of all these findings as final result of our study.



If we triangulate methods, any combination is possible. That is, in qualitative research we may combine qualitative and quantitative or various qualitative methods. These methods are not applied sequentially – as prescribed by a research design – but simultaneously. A well substantiated example for the latter type of triangulation was published by Medina, Feliz, Domínguez & Pérez, (2002, S. 178).

In their study the method of "biograms" gives access to a subject's biography, the method of in-depth interviews assists in understanding specific relevant aspects, and finally group discussions allow to relate the subjective perspectives of several participants in a study to each other and to compare them:



In the central area, where the chronological-biographical perspective, the personal-individual perspective, and the dynamic social-interactive perspective overlap we find at last balanced answers to our research question. Information about other variants of the model of triangulation, for instance combination of perspectives of various researchers or of different theoretical orientations can be found in the articles in Tashakkori and Teddlie (2003), particularly interesting is a paper on rules of integration by Erzberger & Kelle (2003).

(3) Micro-sequences of combinations

Depending on strategic decisions we often have to call into question both when mixing methods implicitly and explicitly how to analyze qualitative data quantitatively, too. Here we summarize answers to this question in a third type of combinations, that is, micro-sequences of combining methods. In any case, we start by reducing our initial data and projecting selected aspects on a quantitative scale, usually a nominal scale. Considering concrete steps, we have AQUAD count selected elements in our data, put the resulting frequency figures into lists and convert these lists into tables ready for statistical analyses. Often the results of a quantitative analysis will be a starting point for further qualitative activities, for instance interpretations in form of determining meta-codes and then analyses of implicants.

In chapter 9 we already mentioned one of the frequently used quantitative techniques in text analysis: counting particular words. More information how to do this will follow in the next section. In section 12.4 we will describe additionally how to count code frequencies. Finally, in section 12.5 we will refer to two examples and go into concrete details of micro-sequences of combinations.

Let us not forget: We can profit from the advantages of combining qualitative and quantitative methods already on the primary level of data given, for instance, particular words in texts, as well as on the secondary level of interpretation, that is our codes and linkages established between codes..

12.3 Frequencies of words

Berelson (1952) developed the principles of content analysis, which he conceived of as a quantitative method, based on assessing the frequency of elements of the *manifest* content. We already mentioned that determining and selecting those elements, which are relevant for content analysis, for sure is impossible without using qualitative procedures. Let us concentrate her on the "mechanics" of counting words.

You get access to frequency options in AQUAD in the main menu under "*Content analysis*" -> "*QUANtitative content analysis*" -> "*Count words*." The following screen shot gives an overview; you find the function "*Count words*" within the last subgroup of functions. The details were described already in chapter 9.

12.4 Frequencies of codes

A combination of qualitative and quantitative analyses is of particular interest on the second level of data analysis, when interpretations, that is categories can serve as data. Counting codes gives general access to these combinations. You may count all types of codes. Particularly counting sequence codes, which AQUAD has inserted additionally into your code files during *linkage* analyses, often leads to most revealing results.

Before AQUAD can count any codes, it needs a list of those codes, which you want to have retrieved. There are two alternatives: (1) Whenever you want to learn about code frequencies, you select the critical codes anew from your project's the master code list or (2) you create a code catalog, save it, and load it whenever you want to have counted its content. We will describe the second alternative in the following; these are the necessary steps:

① Main menu: "*Retrieval*" -> "*Code catalog*" -> "*Create code catalog*"

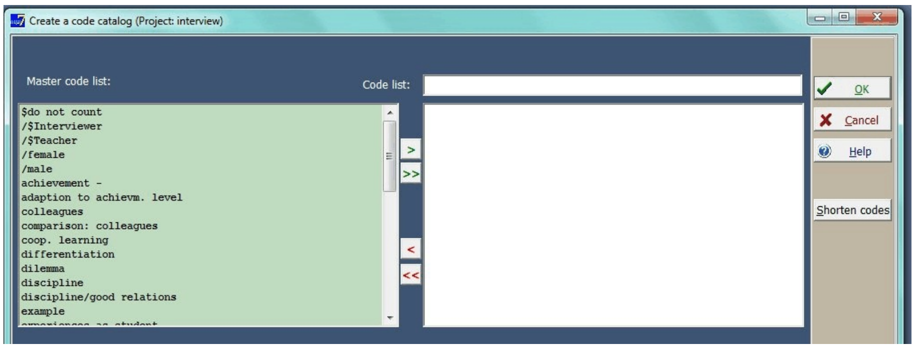
② Main menu: "*Retrieval*" -> "*Count codes*"

③ Main menu: "*Tools*" -> "*Convert frequency list into frequency table*"

In step 2 we saved the frequency results in form of simple listings code file by code file. If we intend to export these frequencies and analyze them in any statistical program, we need to convert the listing of frequencies into a table. Most convenient for this purpose is the format "CSV", which stands for "comma separated values." The conversion tool is integrated in AQUAD.

Here are now the details:

①: "*Create code catalog*"

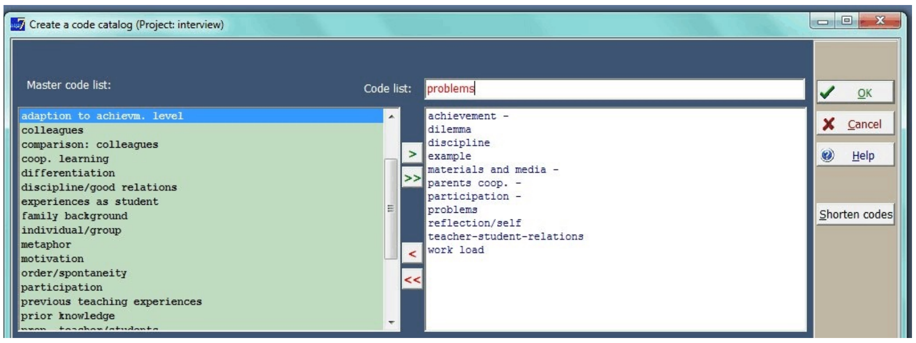


```
$do not count
/$Interviewer
/$Teacher
/female
/male
achievement -
adaption to achievm. level
colleagues
comparison: colleagues
coop. learning
differentiation
dilemma
discipline
discipline/good relations
example
```

We create the list of codes simply by marking interesting entries in the master code list on the left, click on the arrow "to the right" and move them into the selection box at the right – which is still empty in the screen shot above . The double arrow moves *all* codes into the selection box!

Do not forget to write a name for saving your selection into the slot labeled "Code catalog." However, if you click on "*OK*" without having written a name for your code catalog, you will be warned.

Sometimes the button "Shorten codes" at the right margin may be interesting: It presents all selected codes in an additional box, where you can cut (end) parts of critical codes. That is, the algorithm for retrieving/counting will react positively to all codes, which have the remaining beginning. Below you find a screen shot taken before saving the code catalog "problems." If there are similar codes, like in our example "participation -" and "participation" you could just enter the beginning or shorten them,, for instance, "partic" (of course, only if there are no other codes like "particles", "particular issue," etc. which would lead to very mixed results...). To shorten codes, you click on the code name, erase the final, redundant part, and confirm your modifications by clicking "OK" (occurring at the right margin).


Create a code catalog (Project: interview)

Master code list:

```
adaption to achievm. level
colleagues
comparison: colleagues
coop. learning
differentiation
discipline/good relations
experiences as student
family background
individual/group
metaphor
motivation
order/spontaneity
participation
previous teaching experiences
prior knowledge
```

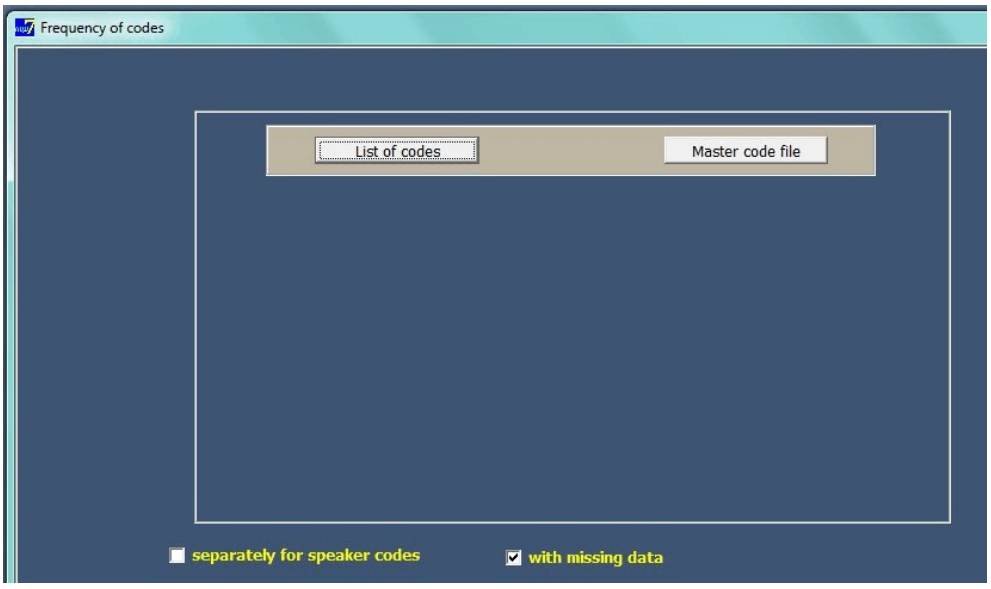
Code list: problems

```
achievement -
dilemma
discipline
example
materials and media -
parents coop. -
participation -
problems
reflection/self
teacher-student-relations
work load
```

Clicking on the green "*OK*" button on top of the right margin starts the function. In our example here, the code catalog is saved as "problems.cco" (**.cco* for "catalog of codes"). Now we may proceed and have AQUAD count these codes in all of our interview files.

②: "*Count codes*"

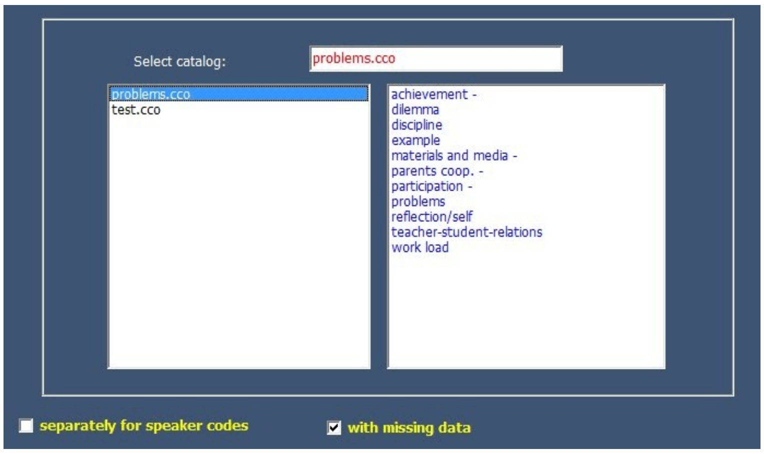
The next screen shot shows two buttons to select codes for counting: Either we load an available code catalog or we open the master code list and create a temporary selection of codes.



At the bottom of this window we find two important settings (in yellow): Codes may be counted in each file "*separately for speaker codes*" (provided that we used speaker codes!). In our interview examples this would not make much sense, because htough we marked the text segments of interviewer and teacher by speaker codes, we have coded the teachers' segments only.

The second option is extremely important! Therefore, the checkbox in form of "*with missing data*" is ticked off by default. Except for quick overviews you should not change this setting! The frequency results can only be used for statistical analysis under the condition that the frequencies of *all* selected codes in all files appear in the final table, that is, negative findings or frequencies = 0 have to occur in the preliminary listing, too.

Now we are ready to click on "*Code catalog*" and select the just created catalog "problems.cco" (see above):

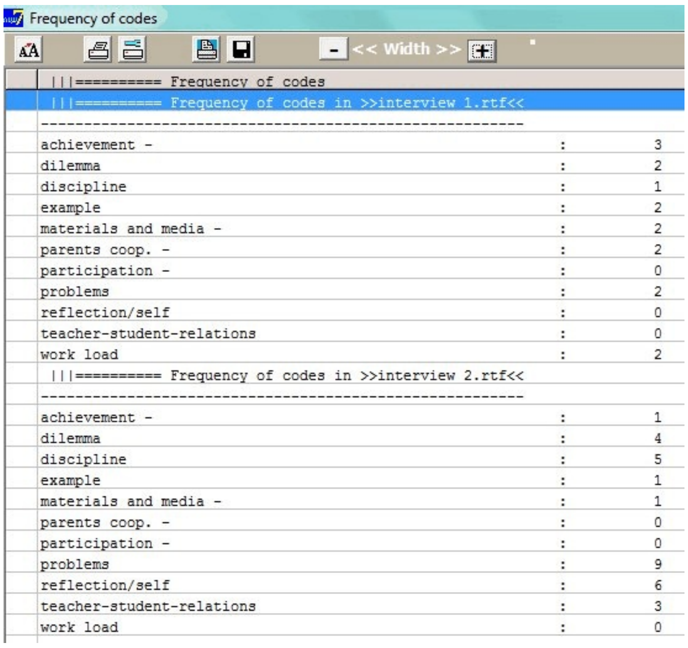

Select catalog: problems.cco

problems.cco
test.cco

achievement -
dilemma
discipline
example
materials and media -
parents coop. -
participation -
problems
reflection/self
teacher-student-relations
work load

☐ separately for speaker codes ☑ with missing data

To check whether you really get what you want the name of the selected catalog and its content appear on the right. Clicking "*OK*" starts the frequency count. Here is a part of the resulting frequency list.

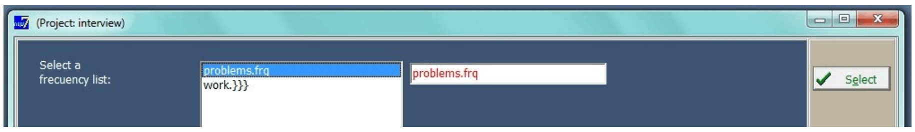


```
 |||========== Frequency of codes
 |||========== Frequency of codes in >>interview 1.rtf<<
-------------------------------------------------------
achievement -                              :      3
dilemma                                    :      2
discipline                                 :      1
example                                    :      2
materials and media -                      :      2
parents coop. -                            :      2
participation -                            :      0
problems                                   :      2
reflection/self                            :      0
teacher-student-relations                  :      0
work load                                  :      2
 |||========== Frequency of codes in >>interview 2.rtf<<
-------------------------------------------------------
achievement -                              :      1
dilemma                                    :      4
discipline                                 :      5
example                                    :      1
materials and media -                      :      1
parents coop. -                            :      0
participation -                            :      0
problems                                   :      9
reflection/self                            :      6
teacher-student-relations                  :      3
work load                                  :      0
```

Remember to save this list, if you want to use these results in further quantitative analyses! We suggest, you save this frequency list under the name of its code catalog and the extension "*.frq" – like word frequencies (see above). That is, here, we enter as name and extension "*problems.frq*"

③: "*Convert frequency list into frequency table*"

We find this option among the "*Tools*" in the main menu. First it shows all frequency files saved up to now in the subdirectory ..\RES of AQUAD (see screen shot below). We select our frequency list "problems.frq" and click on it:

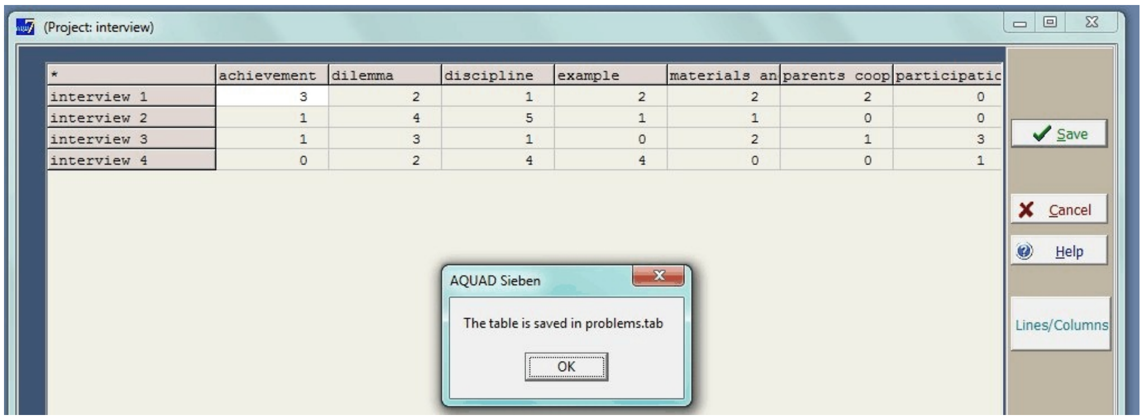


Select a frecuency list:

problems.frq
work.}}}

problems.frq

Clicking on "Selection" produces immediately a table version of our list:

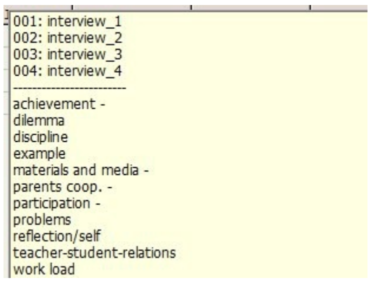

(Project: interview)

*	achievement	dilemma	discipline	example	materials an	parents coop	participatic
interview 1	3	2	1	2	2	2	0
interview 2	1	4	5	1	1	0	0
interview 3	1	3	1	0	2	1	3
interview 4	0	2	4	4	0	0	1

AQUAD Sieben

The table is saved in problems.tab

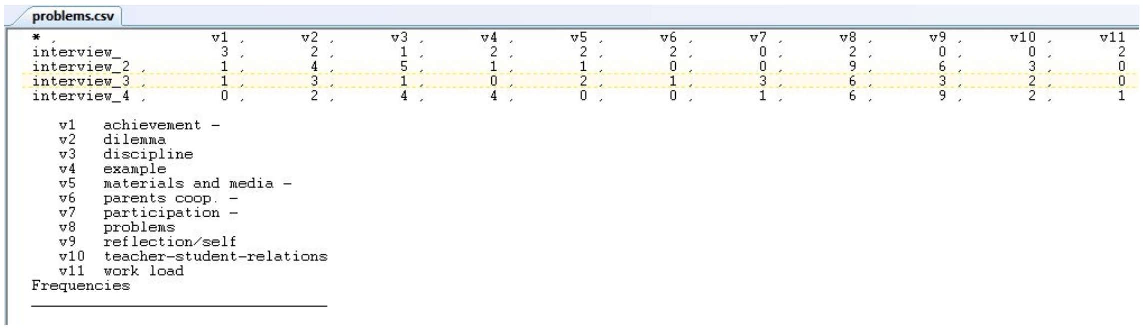
Instead of codes you find as column headers only consecutive numbers; the first column contains numbers representing the files – here the numbers 1 to 4 standing for the four interview examples. If you want to know, to which code the frequencies in a specific column are belonging, just click the button "Codes" on the right margin. That is, what will appear on the screen:


```
001: interview_1
002: interview_2
003: interview_3
004: interview_4
------------------------
achievement -
dilemma
discipline
example
materials and media -
parents coop. -
participation -
problems
reflection/self
teacher-student-relations
work load
```

Clicking with the right mouse key into this list lets pop-up an additional menu, where you may select a print option.

Clicking on "*OK*" at the right margin opens a small window, where AQUAD offers to save the frequency table under the same name as the basic code catalog, but with "*.adt*" (AQUAD data table) as distinguishing extension. If you do not want to save this table, just click on "*Cancel*" instead of "*Save.*"

The "OK" button on top of right margin has meanwhile changed its label - it says now "*CSV.*" As you assume probably and correctly, clicking this button converts the table into the format CSV or "comma separated values" for export purposes:


problems.csv

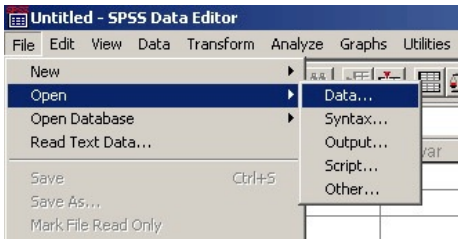
```
*  ,               v1 ,    v2 ,    v3 ,    v4 ,    v5 ,    v6 ,    v7 ,    v8 ,    v9 ,   v10 ,    v11
interview_          3 ,     2 ,     1 ,     2 ,     2 ,     2 ,     0 ,     2 ,     0 ,     0 ,      2
interview_2 ,       1 ,     4 ,     5 ,     1 ,     1 ,     0 ,     0 ,     9 ,     6 ,     3 ,      0
interview_3 ,       1 ,     3 ,     1 ,     0 ,     2 ,     1 ,     3 ,     6 ,     3 ,     2 ,      0
interview_4 ,       0 ,     2 ,     4 ,     4 ,     0 ,     0 ,     1 ,     6 ,     9 ,     2 ,      1

    v1    achievement -
    v2    dilemma
    v3    discipline
    v4    example
    v5    materials and media -
    v6    parents coop. -
    v7    participation -
    v8    problems
    v9    reflection/self
    v10   teacher-student-relations
    v11   work load
Frequencies
________________________________
```

Here again, the rows represent our cases (interview texts), while the codes or variables are separated by commas instead of put into columns.

The cell in the upper left contains an asterisk remembering us after exporting the table to a statistic program, that both the column and the row marked by this asterisk contain only numerical labels, no frequency data!

Of course, you must erase the top row containing nothing else but numerical labels and we may erase also the first column with nothing but case numbers in its cells – but we do so later in your statistic program.

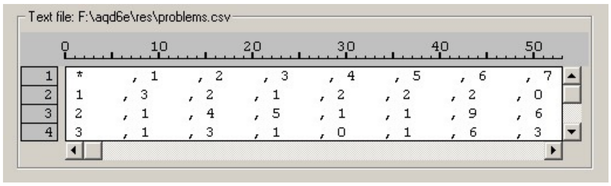
Do not forget to save also this version of your table. We use again the same name, but AQUAD adds now the distinguishing extension "*.csv" Let us see, how this table is imported, for example into the statistical package SPSS:



After starting SPSS we select

-> "*File*" -> "*Open*" -> "*Data*"

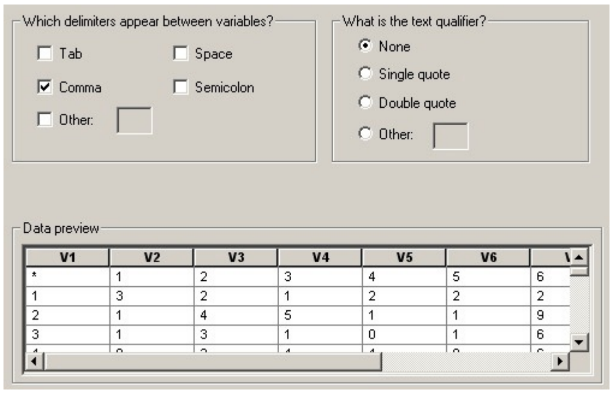
and click on this option.


Text file: F:\aqd6e\res\problems.csv

```
1  *   , 1  , 2  , 3  , 4  , 5  , 6  , 7
2  1   , 3  , 2  , 1  , 2  , 2  , 2  , 0
3  2   , 1  , 4  , 5  , 1  , 1  , 9  , 6
4  3   , 1  , 3  , 1  , 0  , 1  , 6  , 3
```

In the usual "*Load*"-dialogue we enter directory and file name of our csv-table, here "*problems.csv*." SPSS loads this file then and confirms the operation by showing the first columns and rows of this table:

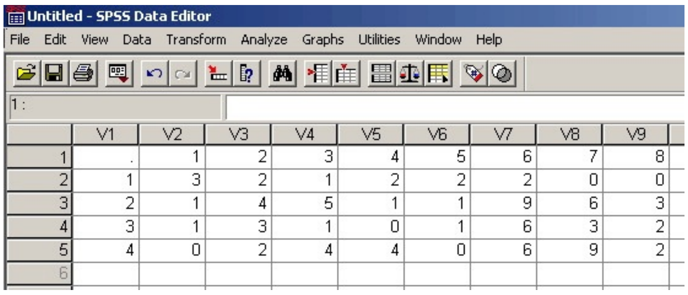
What follows then is a series of windows, where we inform SPSS about essential characteristics of our import table, for instance the fact, that the values are comma separated. Again, this information is put into action and we are presented with a preview of the forthcoming SPSS table as confirmation:


Which delimiters appear between variables?
☐ Tab ☐ Space
☑ Comma ☐ Semicolon
☐ Other:

What is the text qualifier?
● None
○ Single quote
○ Double quote
○ Other:

Data preview

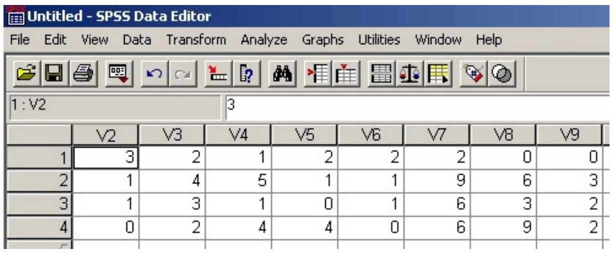
V1	V2	V3	V4	V5	V6	
*	1	2	3	4	5	6
1	3	2	1	2	2	2
2	1	4	5	1	1	9
3	1	3	1	0	1	6


Untitled - SPSS Data Editor
File Edit View Data Transform Analyze Graphs Utilities Window Help
1 :

	V1	V2	V3	V4	V5	V6	V7	V8	V9
1	.	1	2	3	4	5	6	7	8
2	1	3	2	1	2	2	2	0	0
3	2	1	4	5	1	1	9	6	3
4	3	1	3	1	0	1	6	3	2
5	4	0	2	4	4	0	6	9	2
6									

After the final input, the SPSS data editor produces the data table as shown on the left. However, it contains still code numbers in the first row. We must erase these numbers, otherwise they would be taken as frequencies and added (or whatever mathematical operation will be performed) to the frequency scores in the following rows.

Since SPSS shows tables with row numbers, we should also erase the first column of our table, where our cases are already numbered consecutively. On the right, you see the final version of the table:


Untitled - SPSS Data Editor
File Edit View Data Transform Analyze Graphs Utilities Window Help
1 : V2
3

	V2	V3	V4	V5	V6	V7	V8	V9
1	3	2	1	2	2	2	0	0
2	1	4	5	1	1	9	6	3
3	1	3	1	0	1	6	3	2
4	0	2	4	4	0	6	9	2

Now we are ready to run more sophisticated quantitative analyses than mere counting of frequencies, but these analyses cannot be object of considerations in this manual.

12.5 Example: Combination of methods in a study on humor

This section summarizes a study of Leo Gürtler to demonstrate the combination of qualitative and quantitative methods. Mixed methods are especially useful, if we have to analyze large data sets (for instance, questionnaires with open questions/answers), because in this case some of the research questions are necessarily of quantitative nature, and both approaches will complement each other. The goal is to give an example how the processes of theory building and theory testing are linked logically.

In study on students of secondary schools (N = 363, see example of creating sequence codes in chapter 11, section 11.6) Gürtler tried to get access to subjective theories on humor of high school students. They were asked to fill in a questionnaire with nine, theoretically deduced open questions, which should help to find answers to two central research questions:

- *How do students experience humor in classrooms?*
- *Are there differences between students as regards their experiences and reactions, and how are they linked?*

During the analysis with AQUAD, the answers to each of these nine question were coded by a speaker code /$Q1 ... /$Q9. Thus it was possible to select specific questions for closer analysis. Here are two questions as examples:

- *Question 1: What do you understand by "humor?"*
- *Question 6: Maybe you had also negative experiences with humor (in the classroom)? If this is true, what happend in this situation?*

Each student's answers were saved in a separate text file, that is, the AQUAD project included 363 data files. All answers were coded both by content and by structure. Content codes were, for instance, "e*xternally directed humor*" or "*social atmosphere/climate*." Structure was coded as aspect of validity, that is, these codes express whether questions were answered according to their meaning or not. An example is the structural category "*definition*" that can be expected to qualify answers to question 1 – subjective understanding of humor – but not answers to question 6, which asks to describe "humorous" situations in the classroom. Other categories related to activities were "*positive effect(s)*" or "*negative cause(s).*"

In a second round of coding the content-related codes were summarized in meta-codes, based on a qualitative content analysis. Thus, the number of different codes was reduced to 105 content-related and 22 structural meta-codes.

12.5.1 Linkage hypotheses based on analyses of variance

Already when writing memos during coding the answers to the nine questions, and even more when codes were subsumed under meta-codes, several interesting hypotheses were outlined. Additionally the researcher noted that girls seemed to produce more text, more verbose answers than boys; the same difference seemed to be true between students from two different types of German high schools. These observations led to a first qualitative analysis. A tow-factor analysis of variance (gender by type of school) was administered with the following a priori contrasts (Helmert contrasts according to Fox 2002, p. 142 ff.):

			[1]	[2]	[3]
Male	/	Gymnasium (mG)	- 0.5	+0.5	- 0.5
Male	/	Realschule (mR)	- 0.5	- 0.5	+0.5
Female	/	Gymnasium (fG)	+0.5	+0.5	+0.5
Female	/	Realschule (fR)	+0.5	- 0.5	- 0.5

The following comparisons were realized with these data:

[1]: mG & mR versus wG & wR	-> test for gender effect
[2]: mG & wG versus mR & wR	-> test for effect of school-type
[3]: mG & wR versus mR & wG	-> test of interaction gender x school-type

The resulting linear model was highly significant as well as the factors gender ans school-type. No significance could be found for the interaction of gender x school-type. Based on these results, further qualitative analyses were performed separately for the groups mG, mR, wR, and wG.

Theoretical considerations led to a great number of linkage hypotheses regarding subjective understanding and experiences of humor. Some of these linkages involved answers to more than one question, that is, they postulated linkages across the content domains of questions. Since questions were coded as "speakers" within each student's answers, linkage constructions caused no problems; here are some examples:

```
Sequence code
FR_HYPO_2-0-26: AND        /$M2 limits of humor
                AND        MX_negative cause (-)
                AND        MX_reason/relation
```

This linkage states that a student, who writes about negative causes in his/her answer to question 2 ("*limits of humor*") will add some reason for his/her evaluation. That is, the linkage hypotheses probes the assumption that students do not express just some opinions, but have good reasons and can justify their statements. Please, note that "/$M2 limits of humor", that is a speaker code attached to question 2 is used as an element of the hypotheses. Thus we make sure that positive findings in the sense of justified negative causes really are linked with question 2. If we intend to retrieve all those text segments containing the same sequence "*negative cause*" and "*reason/relation*," but do not occur linked to question 2, we just modify the sequence of linked elements: We move the speaker code to the last position and relate it to the other codes by a logical NOT (Remember: Linkages most not begin with a NOT operator):

```
Sequence code
FR_HYPO_X-X-XX:       AND        MX_negative cause (-)
                      AND        MX_reason/relation
                      NOT        /$M2 limits of humor
```

According to this basic pattern quite a number of linkages could be formulated and tested. The quantitative step that followed tried to encounter typical sequences in the answers across students in all groups (mR, mG, wR, wG) or across groups. That is, based on principles of cluster analysis we looked for prototypes in data structures. Contrary to an ideal type in Max Weber's (1988) sense, which never occurs in reality, a prototype is an empirical fact – we only have to find it. A prototype is defined as *the datum or data, which has/have minimal distance to all other data.*

The procedure follows the principles of cluster analysis in explorative data analysis. Normally it has two phases: (1) The distances between each datum and all other data are computed; (2) according to various criteria these distance scores are used to determine clusters. The first step leads directly to the identification of prototypes: If we compute the sum of distances separately for each data set, then – according to the definition – the smallest sum indicates the prototype. Or in other words: The prototype is the center of the data given.

The next step of analyses brings us back into the qualitative domain, that is, we are mixing methods again. Based on our knowledge of prototypes and contrasting data sets (that is, those with maximal distance to all other data) we continue with focused interpretations. In the following we talk about contrasting types as "runaways." Now, above all the following questions need to be answered:

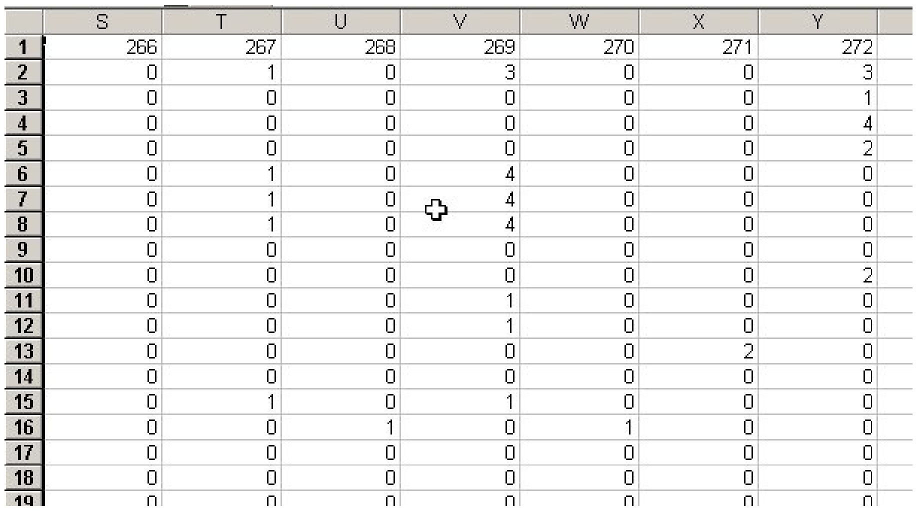
- What is characteristic in a/the prototype(s) as compared to runaways?
- Which codes are involved ?
- Do we learn about anything special or systematic as a result of comparing prototypes and runaways?

As regards our example we expect answers telling us, which of our theoretically grounded linkage hypotheses are represented best or worst by our data. The best fitting data are prototypes, the worst fitting ones are runaways. Remember: *Do not concentrate your interpretations exclusively on prototypes, but pay attention also to runaways as components of equal importance. The most relevant answers to your research questions will be elaborated within the field of tension between the poles of typical and a-typical answers.*

In the following we want to give concrete advice how to put the results of data analyses with AQUAD into distance matrices and continue with quantitative analyses.

Data analysis: How to create distance matrices based on linkage analyses

As already explained above, we have to transform our codings into distance matrices which show how close or how distant linked data segments represented by a sequence code occur in our data. Basis of calculations are the frequencies of all sequence codes in the files of all persons. We have AQUAD count the code frequencies (Which sequence occurs how many times in the code files of which person?) and convert the resulting lists into tables. These tables then are converted int CSV format, because this format can be read by spreadsheet programs or – in our example – by the components of the language "R" (2004) for statistical programming. The final table should look like this example:



	S	T	U	V	W	X	Y
1	266	267	268	269	270	271	272
2	0	1	0	3	0	0	3
3	0	0	0	0	0	0	1
4	0	0	0	0	0	0	4
5	0	0	0	0	0	0	2
6	0	1	0	4	0	0	0
7	0	1	0	4	0	0	0
8	0	1	0	4	0	0	0
9	0	0	0	0	0	0	0
10	0	0	0	0	0	0	2
11	0	0	0	1	0	0	0
12	0	0	0	1	0	0	0
13	0	0	0	0	0	2	0
14	0	0	0	0	0	0	0
15	0	1	0	1	0	0	0
16	0	0	1	0	1	0	0
17	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0

Headline (Row 1): Name or consecutive number of linkage hypotheses = variables (here: 266, 267, 268 etc.)
Columns: Persons
Rows: Linkage hypotheses
Cells: Frequencies of linkage hypotheses

In case rows and columns should be reversed in your table, "R" offers an efficient option to transpose matrices.

Statistical analysis with "R"

"R" (2004) is a programming language dedicated to statistical analysis. You get it free on the Internet (under http://www.r-project.org). On this website you will find also information, introductions, and references to literature on how to work with "R." As regards distance matrices, a particularly helpful manual was written by Handl (2002). Nevertheless, just to give you an impression, we will describe a few commands:

```
table <- read.csv("file_name_of_our_frequency_table.csv", header=TRUE)
```

That's all you type to load a frequency table in format CSV and put it into *one* variable "*table*" in "R." Our table had column headers (consecutive numbers of linkage hypotheses), therefore the characteristic "header" is set to "TRUE". If your table has no labels in the first row, you have to set header=FALSE, otherwise the first row is neglected in later calculations. If you want to check what you got, the command

```
table
```

puts your matrix on the screen.

All you should do now is follow the instructions by Handl (2002, p. 83 ff). Please observe to standardize the matrix (your table) **before** you calculate distances. Thus you balance differences of standard deviations due to interpersonal differences (Handl, 2002, p. 97). Since we want to identify prototypes, we have to calculate the complete matrix of distances(Handl, 2002, p. 96 ff.; on p. 437 you find the program that performs this task). Principally should be found in the rows those variables, which are the objects of distance calculations. In our example we defined the rows by linkage hypotheses; the above screen shot shows hypothesis 1 - 19 as row headers. The columns are reserved for the "data carriers", that is, the persons in our example. Consequently, the cells are filled with frequency scores of linkage hypotheses for each student. Reversed matrices like our initial CSV-table are transposed most simply:

```
table <- t(table)      # reversal of rows and columns in a matrix named "table"
```

You should use "euclidic" distances as unit of distance calculations, except you have theoretically justified reasons to apply an alternative distance unit. The prototype is identified by comparing the sum of distances per row (that is, per linkage hypotheses). The diagonal cells contain the score ZERO, that is the maximal score of similarity or identity, because in these cells we find the distance of a variable to itself. Correspondingly, a score of 1 would express the maximum of dissimilarity.

Well, that is how we construct the vector of prototypes:

```
prototype <- apply(distance_matrix, 1, sum)
# adding the scores of a distance table or matrix labeled "distance_matrix" by rows and creating a  prototype vector
labeled "prototype"
```

This vector is saved together with the distance matrix in format CSV:

```
write.table(cbind(prototype,distance_matrix), file="prototype_table.csv",sep=",")
# saving the prototype vector together the distance matrix under the name"prototype_table.csv"
```

The new table can be loaded and opened in any spreadsheet program. The first (left) column contains the prototype vector. We sort the table according to this vector in ascending order. The resulting order of linkage hypotheses will indicate on top, which linkage hypotheses are most prototypical (highest scores) and at the bottom those linkages, which are least prototypical, that is "runaways." Now we should find out, which meanings are carried by these linkage hypotheses. In our study we found the following order of linkages (across students, independent of school types) as regards question 1 (*What do you understand by "humor?"*); the figure behind "position" indicates the amount of prototypical characteristics: "1" stands for a maximum of prototype qualities, "2" is somewhat less prototypical, etc.

Prototype position 1:	AND	M_externally_dir_humor	AND	M_laughing, joyful, fun
Prototype position 1:	AND	M_laughing, joyful, fun	AND	M_externally_dir_humor
Prototype position 2:	AND	M_classical humor	AND	M_laughing, joyful, fun
Prototype position 2:	AND	M_classical humor	AND	M_borderline / distance to (-)

What stands out is that the linkages placed on position 1 differ only in the sequence of their components. Since both received the same prototype score, we can treat them as equivalent. Thus we can state that our sample is characterized that our students conceive of humor as something that has to do with laughing, ajoy, and fun and has an external object (or is directed externally). This may indicate that humor is a social, interactive matter – not something happening alone at home in front of TV. Additionally, this finding backs the validity of our students' answers: They really had the classroom context in mind, when they filled in the questionnaires.

The linkages on position 2 reveal an understanding of humor as something that has to do with "jokes" (classical humor), which cause laughing or fun (second place within the linkage sequence) – but also negative experiences (see second linkage on prototype position 2).

Next we should gather information about the distribution of these linkages among our sample and then return to AQUAD and retrieve and compare the corresponding original answers of students.

Additionally, we would like to draw your attention to a publication by Oldenbürger (1981), who described how to dichotomize empirically a data base (p. 153ff.) by establishing a proximity matrix (our distance matrix is one of this kind) and distinguishing between concepts related to each other and concepts, which do not occur in linkages. Thus, it is possible not only to identify prototypes, but also clusters that – unlike the findings of "classical" hierarchical cluster analysis – must not be strictly separated from each other. This sort of splitting up a data base is directly built on empirical data and of course compatible to the procedures of "grounded theory"

(Glasser & Strauss, 1965). The output is a 0-1-matrix, in which we can read directly whether a given code is related to any other codes (matrix score = 1) or not (matrix score = 0). Again, adding the scores across rows amounts to the already described prototype vector. However, here the maximum indicates the prototype, because "1" represents an existing link or minimal distance, while a matrix score "0" represents a lack of relations. There is a program called "OptSchnPr" available in in the "R"-module "Proxim" (Oldenbürger, 2004; see web link in the references), which was constructed to split up distance matrices. A further "R"-program to manage input and output of CSV-formated tables is available from the author of AQUAD.

12.5.2 How to identify types by quantitatively based analysis of implicants

Usually hypotheses and decisions about promising components of implicants are formulated qualitatively, based on impressions noted in memos, comparisons within and across original data, linkage hypotheses, etc. Sometimes there are only minimal differences or we are confronted with highly complex data, for instance, if our topic is really new and we cannot rely on any available empirical findings. In such a situation alternative possibilities to generate hypotheses would be helpful. One of these alternatives will be presented subsequently.

The necessary statistical tools are really simple. We begin by counting codes, that is, we count those codes or meta-codes, which signal some relation as regards content area or topic. Then we standardize (z-transformation) the frequencies of each code across subjects or cases and correlate afterwards the resulting z-scores. Beginning with extremely scoring codes (for instance, z-scores beyond a range of ± 2 units of standard deviation, which is above or below scores of ± 2 in a z-distribution) we select codes correlating significantly ($p <= 0.05$) with these "extreme" codes as components of code configurations (implicants) and decide about a criterion code (a possible "cause" or "predictor" of "effects" represented in implicants). However, as already described, we are not limited by a research design that postulates independent and dependent variables, thus enforcing determined configurations. Instead we state plausible, reasonable, but anyway for the time being undetermined components as well as criteria (= "predictors") of implicants.

In a following step we have to evaluate qualitatively, whether these configurations of codes are relevant and contribute to understand our research problem better. Finally we have the remaining selection of configurations tested in AQUAD – both applying the criterion in a positive sense (criterion = TRUE) and in a negative sense (criterion = FALSE). This strategy resembles a sort of double-entry accounting and may sensitize researchers not to focus their attention on positive findings only, but to derive knowledge also from "negative" configurations.

We repeat these steps until a well-founded classification or typology does not change relevantly, that is until we reach a state of analysis that is called "saturation" by proponents of Glaser and Strauss' (1965) approach of grounded theory. In other words: At this point we do not expect any longer to come upon substantially new discoveries that would necessarily modify the answers to our research questions.

Of course, we must neither overdo in testing numbers of various configurations and smuggle the notorious "shotgun strategy" from quantitative testing into the field of qualitative analysis nor should we build complex arguments referring to several people upon a single criterion of implicants. The first strategy would rise the probability to detect reasonable combinations "at random" and – even worse – accept them blindly. The second strategy is problematic, because interpretations based on very few implicants demand a degree of generality, which is not permitted due to the preceding data reduction.

What we recommend is to identify at least one configuration per case (person, interview, video, etc.) to raise the probability that relations between configurations across units of analysis can be found. Especially tensions or even contradictions between configurations ("Why do we find this here, but not in other cases?") stimulate further, empirically based conclusions by reverting to the original data. Interpretations then do not depend on a single analysis of implicants, but are derived from explicit efforts to establish a typology. And, not to forget, because Weber's ideal types are constructs not to be found in reality, we are not so much interested in types, but in relations of various types and differences between them.

An example will demonstrate subsequently how to put these considerations into practice. Gürtler (2004) got a hunch during the analysis of teachers' implicit theories on humor in private and professional contexts that one of his subjects had a highly idealized, spiritually slanted concept of humor. As described above, all (meta-)code frequencies were z-transformated and correlated to identify a metacode representing an ideal attitude towards humor. This metacode was used as criterion in logical minimizations. All metacodes that correlated significantly with the criterion were put into the model of implicants and tested, one after the other. These are the results:

```
A = M_awareness
B = M_ideal_attitude_tow._humor
C = M_communication_soc._IA
D = M_self-knowledge_insight
E = M_self-concept_int_ext
F = M_jokes_active_evaluation

Criterion:            Condition B / FALSE
Implicant/s:    ACeF [6 9], acde [4 8 10]
Criterion:      Condition B / TRUE
Implicant/s:    ACDEf [1]
```

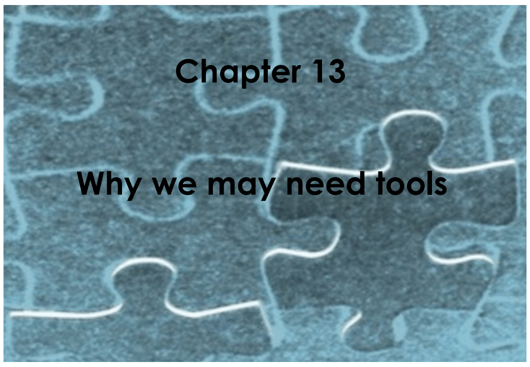
Further analyses showed it was necessary to test similar configurations:

```
A = M_ideal_attitude_tow._humor
B = M_cog.activities_without_soc._IA
C = M_communication_soc._IA
D = M_people_unspecific
E = M_self-knowledge_insight
F = M_reflection
G = M_negation_of_concept

Criterion:      Condition A / TRUE
Implicant/s:    BCDEFG [1]
(There were no cases matching the criterion condition 1 = FALSE)
```

Both results led to interesting assumptions. One of them, for instance, corresponds to the researcher's impression during the interview that in this case humorous activities involving jokes are meaningless (ACDEf), while self-knowledge, social interaction/communication, and this person's self concept played an important role.

Taking into account also the negative criterion (Condition B = FALSE) showed that in one of the other configurations jokes were an important component, which made the criterion "ideal attitude towards humor" meaningless, that is FALSE. In this configuration (ACeF) the role of self-concept (from an intra-personal and an external point of view) is no longer important.

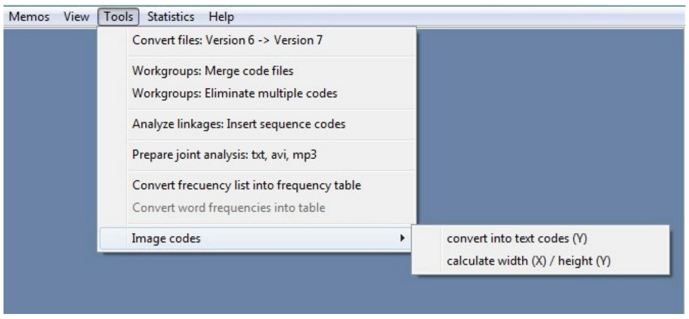
A second analysis with a slightly modified model did no longer include the code "awareness", but integrated "cognitive activities without social interaction" and "reflection" as predictors. More detailed analyses should clarify the meaning of this configuration.



Chapter 13

Why we may need tools

The last group of functions in the main menu is labeled "*Tools*." The screen shot below shows, which options are available. The option labels as well as the windows occurring after clicking are more or less self explaining. Therefore we describe only briefly the purpose of each function and list stepwise how to handle its possibilities.


Memos
View
Tools
Statistics
Help
Convert files: Version 6 -> Version 7
Workgroups: Merge code files
Workgroups: Eliminate multiple codes
Analyze linkages: Insert sequence codes
Prepare joint analysis: txt, avi, mp3
Convert frecuency list into frequency table
Convert word frequencies into table
Image codes
convert into text codes (Y)
calculate width (X) / height (Y)

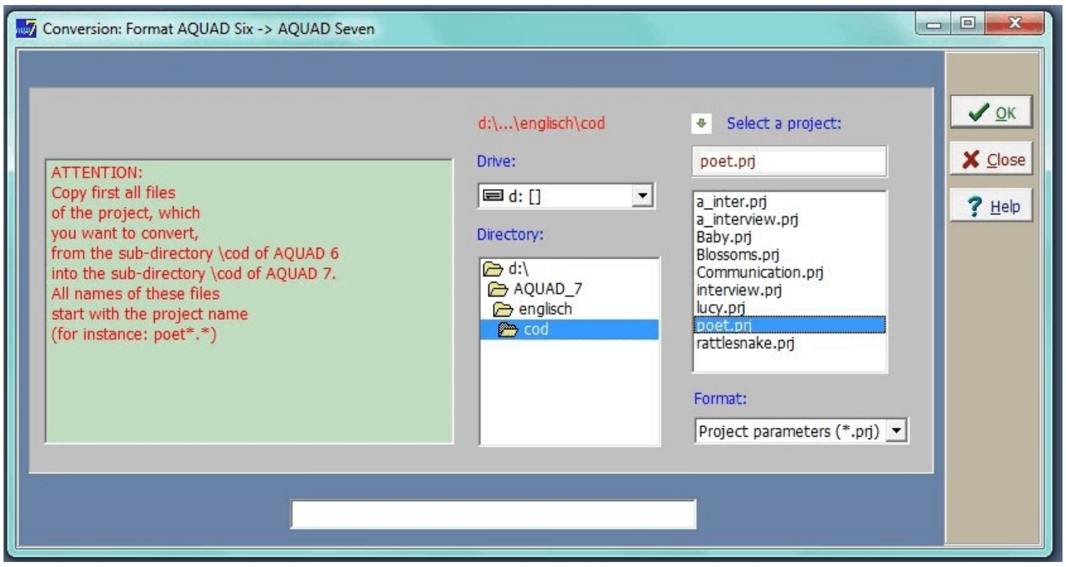
13.1 Converting codings of AQUAD 6 into the format of AQUAD 7

Aquad 7 cannot handle directly code files produced with the preceding version, because earlier codes were only up to 30 characters long. Now the limit are 60 characters. This conversion routine adapts your old code files to the conditions of version 7.

You will experience no problems, if you observed the conventions valid for coding in AQUAD 6. The conversion routine expects that

– all names of text files have an extension ".txt" or ".rtf ;"
– all code files were saved (automatically) a sub-directory ..\COD;
– in this sub-directory the master code list ("*.amc") of the project can be found.

Clicking on the conversion routine in the menu opens the window presented below. There (in the right part) you care first about the settings that tell AQUAD where to find your old code files. That is, you select appropriate drive and path settings (in the example below: drive "d:" and path "d:\aqde\COD"). In the box on the left you will be presented with the names of available project settings. If these names do not occur automatically, just double click on the marking "COD". If still no project names are visible in the box on the left, you may have moved your old project settings to some other location ...


Conversion: Format AQUAD Six -> AQUAD Seven

ATTENTION:
Copy first all files
of the project, which
you want to convert,
from the sub-directory \cod of AQUAD 6
into the sub-directory \cod of AQUAD 7.
All names of these files
start with the project name
(for instance: poet*.*)

d:\...\englisch\cod

Drive: d: []

Directory:
d:\
AQUAD_7
englisch
cod

Select a project: poet.prj

a_inter.prj
a_interview.prj
Baby.prj
Blossoms.prj
Communication.prj
interview.prj
lucy.prj
poet.prj
rattlesnake.prj

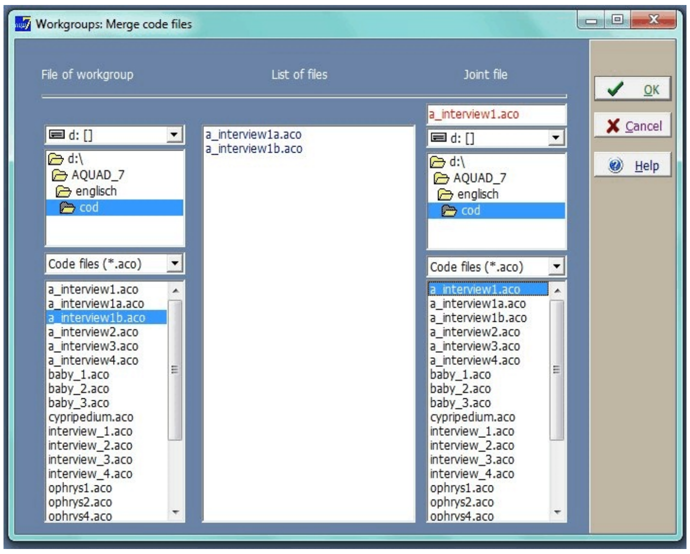
Format: Project parameters (*.prj)

In our example we find the familiar project "*poet.prj*" among other project files. We click on its name and the name will be copied into the selection slot above (the green arrow is pointing to it). Now AQUAD has all information it needs, and we can click on "*OK*" and wait for the results.

The routine effects that we find afterwards our old project listed among the projects of AQUAD 7; correspondingly all (internal "*.atx") text files and code files belonging to this project were converted and copied into the matching sub-directory of AQUAD 7.

13.2 Tools for work groups: Merging code files

A group of researchers working together at a common project often decides to divide the work among the members. Particularly during coding this strategy both helps to save time and to intensify the analysis:


Workgroups: Merge code files

File of workgroup

d: []
d:\
AQUAD_7
englisch
cod

Code files (*.aco)

a_interview1.aco
a_interview1a.aco
a_interview1b.aco
a_interview2.aco
a_interview3.aco
a_interview4.aco
baby_1.aco
baby_2.aco
baby_3.aco
cypripedium.aco
interview_1.aco
interview_2.aco
interview_3.aco
interview_4.aco
ophrys1.aco
ophrys2.aco
ophrys4.aco

List of files

a_interview1a.aco
a_interview1b.aco

Joint file

a_interview1.aco
d: []
d:\
AQUAD_7
englisch
cod

Code files (*.aco)

a_interview1.aco
a_interview1a.aco
a_interview1b.aco
a_interview2.aco
a_interview3.aco
a_interview4.aco
baby_1.aco
baby_2.aco
baby_3.aco
cypripedium.aco
interview_1.aco
interview_2.aco
interview_3.aco
interview_4.aco
ophrys1.aco
ophrys2.aco
ophrys4.aco

For instance, part of the team could look in the interview recordings (or texts) on your installation file only for what the teacher says about social relations (to students, colleagues, parents, superiors), another part for statements about teaching and learning, a third sub-group for reflections on the relation of theory and practice in everyday tasks in classrooms, etc. As a variation of cooperation in research teams, the files of a project could be distribu-ted among subgroups or individual members once they have elaborated a common system of categories for coding.

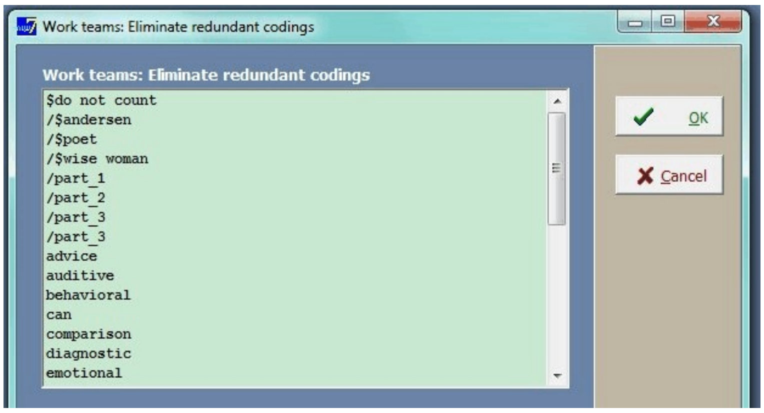
Both strategies are not only well suited to speed up the common work, but also to introduce beginners into concrete applications of qualitative methods. Comparing ways of access, problems, and results between groups sets off highly intensive and fruitful debates about approaches of qualitative methodology. At least , we experienced these effects in university seminars for beginners and for doctoral students.

However, there is always the same problem: various code files referring to the same data file have to be merged into a single code file for all team members. The following analyses will then be based on this file, which contains all interpretative perspectives elaborated by the whole group. Already when we distribute the work tasks, we should take care that the groups use different names for saving code files attached to the same data file, otherwise one code file will overwrite the other when we copy them from the teams' PCs to the project manager's computer. Let us see, how the "merge"-tool is operated:

After clicking on the function in the main menu, the window shown on the left page occurs – with mostly empty boxes, of course. If necessary we change the settings in the upper left part, where drive and directory of the actual code files (after copying them to the project manager's PC) are determined. On top of the file box (lower left side) we find two code files "*a_interview1a.aco*" and "*a_interview1b.aco.*" These names tell us, that a group A and a group B coded separately the audio recording of interview 1 (on your installation file: "a_inter1.mp3"). Now we intend to merge these two code files, therefore we select them by clicking, which copies their names into the box labeled "List of files" in the middle. What we need now is a name for the merged files. The box "Merged files" on the right contains at the moment the same list as the box on the left. We can click on one of the already selected files, in the example the file "*a_interview1a.aco*;" this copies the highlighted name in read color into the name slot right on top. One of the selected code files after the other (here only on file more) will be added to the file with this name after clicking "*OK.*" Because we need a file name corresponding to the initial data file "*a_interview.mp3*" we erased the "*a*" from the selected file name an will thus get a merged code file "*a_interview1.aco.*"

13.3 Tools for work groups: Eliminate multiple codes

If we merge code files related to the same data file (see section 13.2), we usually find multiple codes, in this case several identical codes attached to the same data segments. In our interview example each work group will use the speaker codes "/*$Interviewer*" and "/*$Teacher*" to characterize corresponding data segments. Of course, we have to eliminate multiple occurrences of identical codes, otherwise we would produce erroneous results when we have our codes counted, when we have linkage hypotheses tested, etc.


Work teams: Eliminate redundant codings
Work teams: Eliminate redundant codings
OK
Cancel

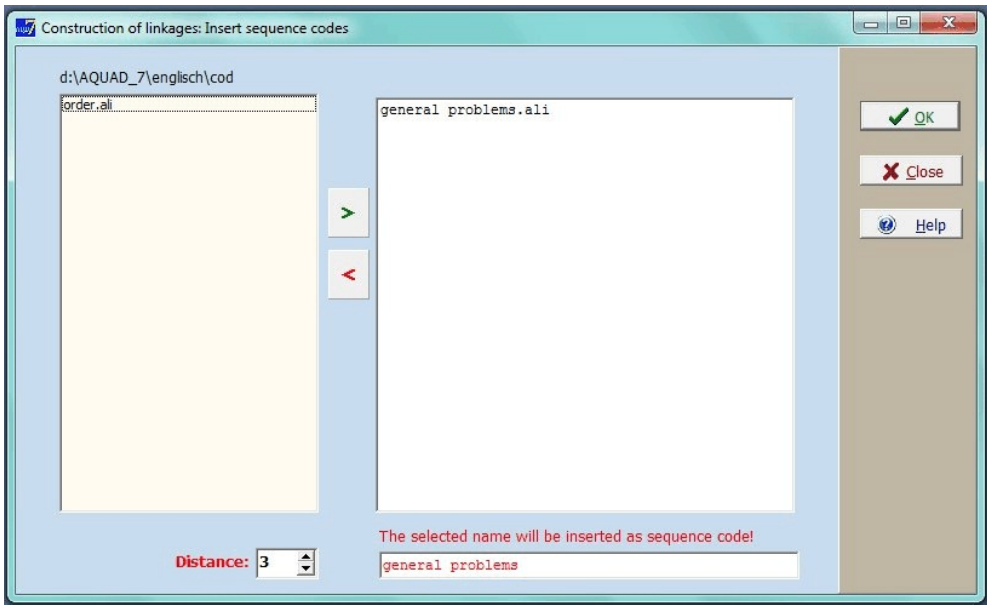
```
$do not count
/$andersen
/$poet
/$wise woman
/part_1
/part_2
/part_3
/part_3
advice
auditive
behavioral
can
comparison
diagnostic
emotional
```

This job is done by the function "*Work groups: Eliminate multiple codes.*" Based on the master code file of your actual project, AQUAD looks for redundant code entries and adapts the merged files. Clicking on "*OK*" starts the job.

13.4 Applying linkage designs and inserting sequential codes automatically

In large projects with a great number of data sets and codes it has turned out to be effective, if researchers first concentrate on constructing linkage hypotheses under theoretical points of view. These constructions are saved and we can apply them later and have AQUAD additionally insert sequence codes in our code files wherever the hypothetical linkage is true.

However, if we construct a number of hypotheses, it will be boring as well as a waste of time to go through all the steps of linkage testing – selecting the function from "*Tools*," selecting a linkage construction, writing a sequence code, saving the results – again and again. Therefore, all these steps are automatized in a sort of batch function:


Construction of linkages: Insert sequence codes
d:\AQUAD_7\englisch\cod
order.ali
general problems.ali
>
<
OK
Close
Help
The selected name will be inserted as sequence code!
Distance: 3
general problems

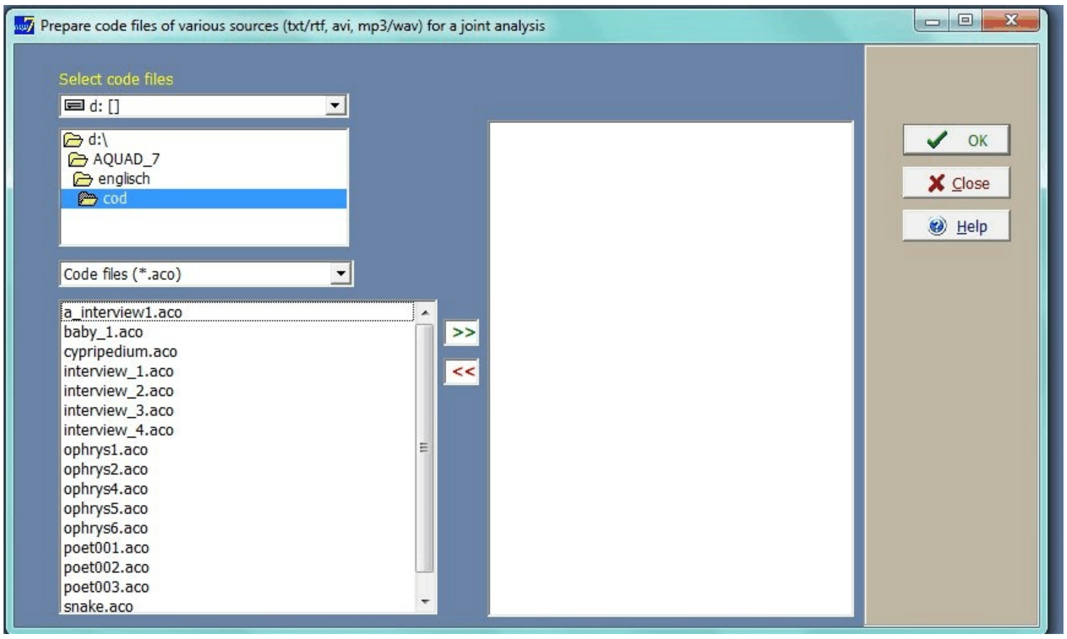
In the left box all linkage constructions ("*.ali" for AQUAD Linkage) available in the code subdirectory (here: "D:\AQUAD_7\englisch\cod") are listed. There were only two linkage constructions, but we already selected one by marking it and we moved it into the selection box on the right side by clicking on the green arrow. If you assume that the red arrow will reverse the operation (for marked items, of course), you are absolutely right! Additionally we may modify the default value of 3 units of distance (lines or seconds).

What we cannot enter in this batch mode is determining a specific sequence code for each of the selected linkage hypotheses. Sequence codes will be determined automatically, that is, AQUAD takes the name of the actual linkage hypothesis and uses it as code name. In our example all positive findings according to the linkage "general problems.*ali*" will be coded as "*general problems*."

13.5 Applying codes from text-, audio- and video-files in the same project

In a large project the problem may arise, that some of the conversations, discussions, interviews etc. that you want to analyse are transcribed, while others are available only as sound- or video-recordings. This tool here will create a new project file, in which the different types of data may be mixed. What has to be implemented is a function that allows to determine critical distances between coded data segments (important for testing hypothetical linkages, see chap. 8.3) no matter whether they are expressed in text lines, 1/10 of seconds or frames. The function takes as a basis that one text line (formatted as recommended here) corresponds with ca. three seconds of talking or 75 frames in a video scene. In accordance with these assumptions the time marks in sound files and the frame numbers in video files are converted.

Nothing will happen to your original code files – the converted files as well as the project file will be saved with an underscore in front of the original names (_xxx.cod). To work with these new files you simply open the new project file and continue as usual.



13.6 How to convert lists of code frequencies in tables

This function was already describe in detail in the section on combinations of qualitative and quantitative methods. Please, read chapter 12, section 12.4 to find out, what this function is good for and how to use it.

13.7 How to convert lists of word frequencies in tables

The result of a quantitative text analysis is stored in form of a list of frequencies of those words that were counted (see chapters 9.1 and 12.3). To convert this list in a table for statistical analyses, AQUAD takes the part containing the alphabetically sorted words and processes it like a list of code frequencies. The final result is a table in the format CSV, which can be imported in statistical programs. Please, read more about the details in chapter 12.4.

13.8 How to modify codes of graphic files

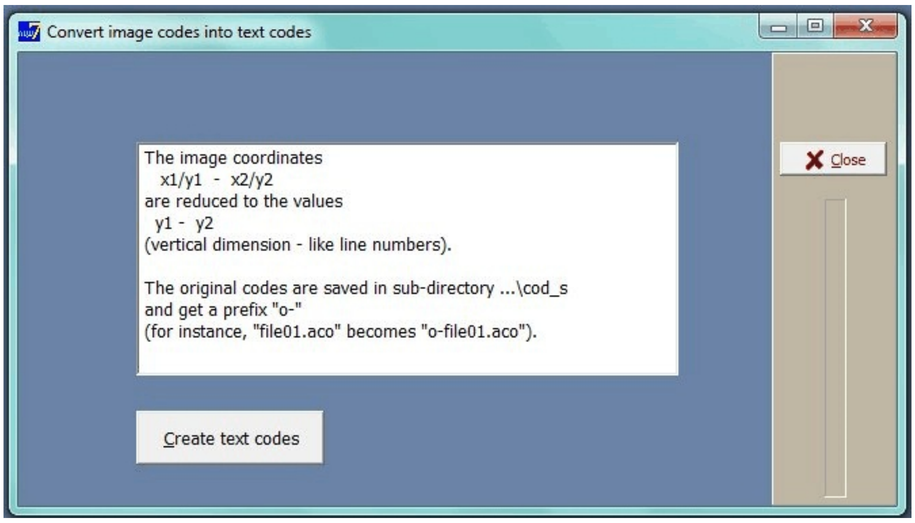
13.8.1 Converting picture codes (x/y) into text codes (y)

In chapter 7, section 7.1.12 we described how to analyze scanned text files, that is texts available in form of picture files. The only problem is that we have to find a way how to localize coded text segments without given line numbers.

When we code texts, we mark relevant segments by drawing a rectangular frame around them. In our code files the program saves the coordinates of the upper left corner (x1/y1) and the lower right corner (x2/y2) of this frame.

With these data it is no problem to retrieve any coded segment irrespective of what its content is – some scanned lines of text, part of a photo, etc. However, if we intended to determine the sequential order of graphic data segments like we do with segments of text, audio and video recordings (for instance: What occurs before a specific segment? What is included in a specific segment? Which segments overlap each other?), we would run into trouble. A sequence is defined as a one-dimensional, linear order of elements, but now we have two-dimensional objects and corresponding coordinates. Well, there is a simple solution, which works well for "texts as pictures," in our case texts, which were scanned or photographed and saved in JPG format: we eliminate the x-components from each segment's coordinates of position on the graphic surface. What we keep are the y-coordinates, that is information about beginning and end of each segment on the (vertical) y-axis – that is, a functional equivalent of line numbers in text files.

The initial code files are saved and automatically activated again, if a function of analysis (for instance, coding again later) demands graphic coordinates. In this case, that is, if you interrupt a sequential analysis of graphically represented data be editing your initial graphic codes, you have to repeat the conversion into text codes afterwards.



The image coordinates
x1/y1 - x2/y2
are reduced to the values
y1 - y2
(vertical dimension - like line numbers).

The original codes are saved in sub-directory ...\cod_s
and get a prefix "o-"
(for instance, "file01.aco" becomes "o-file01.aco").

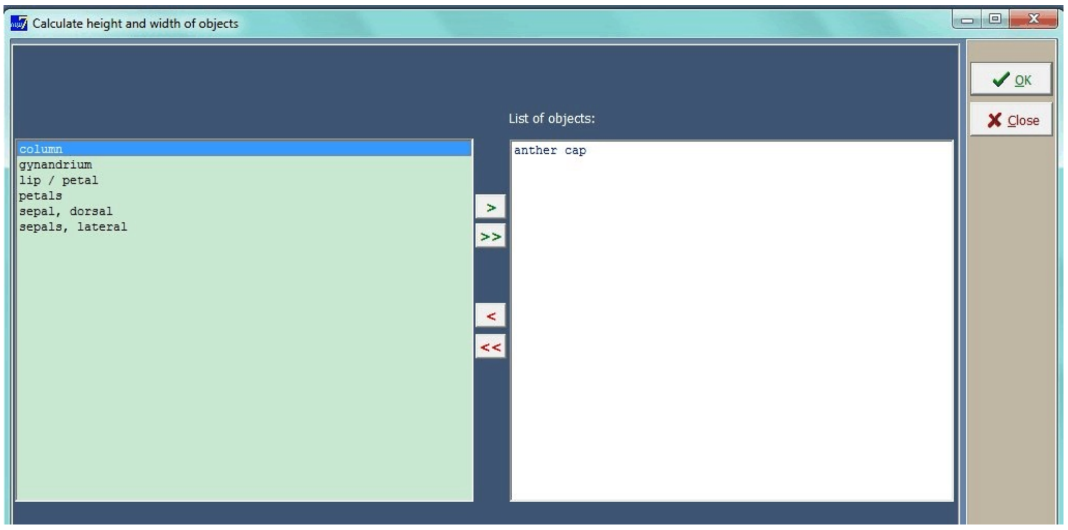
This option is available only if you actually work with the data of a picture project, that is, if an appropriate project is "opened." Clicking on the button "*Create text codes*" starts the conversion like described above in all files of the open project. Don't worry: Your initial picture codes are saved automatically in a sub-directory "..\cod_s" of AQUAD and written back into "..\cod" if necessary. An additional button "*Original codes*" appears, if you start this function again. Clicking on it permits to undo the conversion on demand. Text codes are *not* saved in this case, that is they are not copied into "..\cod_s", but overwritten in "..\cod" by your original code files. However, you can repeat this conversion in almost no time whenever you need text codes again.

13.8.2 Computing/coding width and height of picture segments

To answer some specific research questions we need a function to compute the dimension of selected picture segments. It may be interesting, for instance, to find out the sizes of particular picture elements and to compare them when analyzing children's drawings. How big was the father drawn as compared to mother and siblings in this child's family drawing – and did all children produce similar differences in size? Or: Which proportion of area does a specific object consume in the drawings of different children?

The function "*Add height/width codes*" computes height and width of selected picture segments (selected by their corresponding codes) as differences of vertical and of horizontal pixel-coordinates and inserts the results in form of new codes in the available code files. Consequently, you have to code critical picture segments *before* you can use this function.

Either by loading an already created code catalog or by selecting actually from the master code list you determine those elements in your pictures, whose height and width are interesting in your analysis. The computation is quite simple: The difference of coordinates y2-y1 gives the height, the difference x2-x1 the width of a segment (in pixels). The procedure begins – like counting codes – with selecting critical codes:

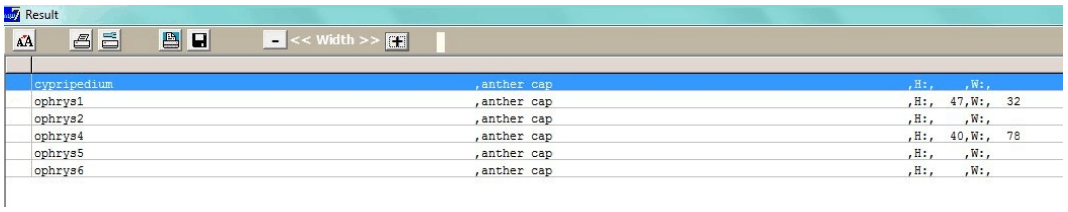


List of objects:

```
column
gynandrium
lip / petal
petals
sepal, dorsal
sepals, lateral
```

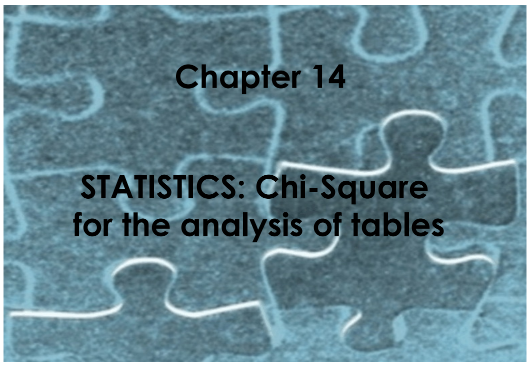
```
anther cap
```

Clicking on "*OK*" after selecting codes produces two code entries added to each of the selected codes. For instance, if we select in the example files of our project "blossoms" the code "anther cap" (see screen shot above), the routine computes height and width as difference (in pixel units) the coordinates of this part of the blossoms in our photos. Below you see that two codes were added, one for ophrys2.aco, the other for "ophrys4.aco," showing height and width of the coded object in pixels. In the other files, no "anther cap" was coded, therefore position 4 and 6 are blank.



```
cypripedium                                     ,anther cap                              ,H:,    ,W:,
ophrys1                                         ,anther cap                              ,H:,  47,W:,  32
ophrys2                                         ,anther cap                              ,H:,    ,W:,
ophrys4                                         ,anther cap                              ,H:,  40,W:,  78
ophrys5                                         ,anther cap                              ,H:,    ,W:,
ophrys6                                         ,anther cap                              ,H:,    ,W:,
```

Do not forget to save these results for later use in statistical analyses – just click on the diskette icon on top of the results. Remember that this file will also be saved in CSV format, that is, you can easily pick the relevant data (height and width) on the fourth and sixth position in each line.



Chapter 14

STATISTICS: Chi-Square for the analysis of tables

In a table analysis with AQUAD categorial data, that is here the frequencies of particular codes, are sorted according to profile characteristics of cases. Let us assume a fictitious interview study with 52 young teachers, 22 male and 30 female, who talk about their experiences in the classroom. Most of them report discipline problems and many of them talk in connection with discipline about themselves and their role in the classroom. Because the study tries to find out about differing reactions of male and female teachers to discipline problems, the researchers run a table analysis entering the profile codes "/*male teacher*" and "/*female teacher*" to define the columns and the conceptual codes "*discipline problems*" and "*reflection/self*" to define the rows of the table:

	/male teacher	/female teacher
Discipline problems	24	28
Reflection / self	29	33

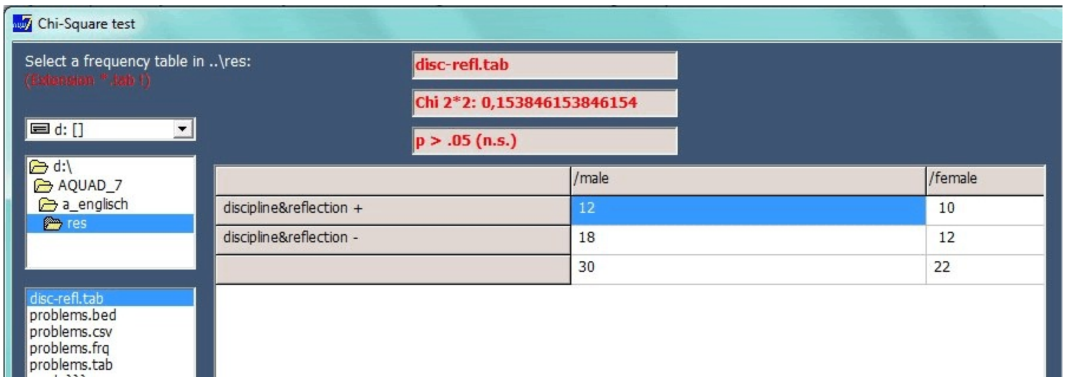
The researchers decide to have the linkage "*discipline problems*" and "*reflection/self*" marked by a sequence code "*discipline + reflection*" and thus find quickly (by applying "*Retrieval*" -> "*particular code*") that in some cases this linkage appears several times, i.e. almost whenever discipline problems are mentioned, whereas in other cases this sequence is not observed at all. Altogether the sequence code appears 15 times in interviews with male teachers and 22 times in interviews with female teachers – however, in some cases several times and in others not at all. In a statistical comparison of several samples (here: male vs. female teachers) based on qualitative (nominal) variables in each cell of the frequency table has to appear the exact number of cases/persons identified by the same characteristic. Therefore, the critical variable "*discipline + reflection*" has to be converted in a variable of alternatives. In small samples we recommend to follow the strategy of truth values (characteristic observed = True; not observed = False; cf. chapter 11) and enter these two possibilities as additional "*discipline + reflection* +" and "*discipline + reflection* -". Thus we find in the code files of 22 male teachers 12 positive and 10 negative alternatives, in the code files of female teachers 18 positive and 22 negative alternatives.

In case of larger samples we could create three (or more) shaded variables "++", "+" and "– ". The number of cases in the sample is important for this decision, because the expectancy values in the cells of the new table (see below) must not be too small. Otherwise, the Chi-Square distribution would be distorted. As a rule of thumb Cochran (in Lienert, 1973) recommends: In maximally 20% of all cells the expectancy value (number of expected cases) may be less than 5.

A table analysis with the modified code files – counting the new codes "*discipline + reflection* +" and "*discipline + reflection* -" – we get a new table of four cells for the comparison of two samples:

	/male teacher	/female teacher
Discipline&reflection +	12	18
Discipline&reflection –	10	12

We save this result (click on the diskette icon in the tool bar on top of the window) using the file name "*disc-refl*" and the extension "*.tab*". Then we can activate the option "*Chi-Square*" in the menu-group "*Statistics*".



disc-refl.tab
Chi 2*2: 0,153846153846154
p > .05 (n.s.)

	/male	/female
discipline&reflection +	12	10
discipline&reflection -	18	12
	30	22

disc-refl.tab
problems.bed
problems.csv
problems.frq
problems.tab

The codes have a length of 60 characters each, therefore only part of the table is visible, but the rest can be moved with the scrollbar at the bottom of the window.

A Chi-Square of 0.154 (at 1 degree of freedom in this table) is not statistically significant, male and female teachers in this sample are not different as regards the particular comparison.

Of course, you may also compare more than two samples of alternative characteristics, for instance, data of teachers in primary schools, secondary I and secondary II. The we get a 3*2-table instead of a simple 2*2-table, generally a k*2-table as the basis of calculations. The other way round, working with two samples and a variable with more than two levels, we get a 2*m-table (m for the number of levels). Finally, in the most general case, we could calculate the Chi-Square for k samples and m characteristics. AQUAD takes these possibilities and their varying degrees of freedom automatically into account.

References

Berelson, B. (1952). *Content analysis in communication research.* Glencoe, Ill.: The Free Press.

Eisner, E. W. (1981). On the differences between scientific and artistic approaches to qualitative research. *Educational Researcher, 10(4)*, 5-9.

Eisner, E. W. (1983). Anastasia might still be alive, but monarchy is dead. *Educational Researcher, 12(5)*, 13-14/23-24.

Erzberger, C., & Kelle, U. (2003). Making inferences in mixed methods: The rules of integration. In A. Tashakkori & C. Teddlie (Eds.), *Handbook of mixed methods in social and behavioral research* (pp. 457-488). Thousand Oaks: Sage.

Fetterman, D. M. (1982). Ethnography in educational research: The dynamics of diffusion. *Educational Researcher, 11(3)*, 17-22.

Fetterman, D. M. (1988). Qualitative approaches to evaluating education. *Educational Researcher, 17(8)*, 17-24.

Firestone, W. A. (1987). Meaning in method: The rethoric of quantitative and qualitative research. *Educational Researcher, 16(7)*, 16-21.

Fischer, P. M. (1994). Inhaltsanalytische Auswertung von Verbaldaten. In G. L. Huber & H. Mandl (Hrsg.), *Verbale Daten* (2. Aufl.) (S. 179-196). Weinheim: Beltz.

Fox, J. (2002). *An R and S-PLUS companion to applied regression.* Thousand Oaks: Sage .

Fühlau, I. (1978). Untersucht die Inhaltsanalyse eigentlich Inhalte? Inhaltsanalyse und Bedeutung. *Publizistik*, 7-18.

Fühlau, I. (1982). *Die Sprachlosigkeit der Inhaltsanalyse. Linguistische Bemerkungen zu einer sozialwissenschaftlichen Methode.* Tübingen: Gunter Narr.

Galtung, Johan (1990). Theory formation in social research: A plea for pluralism. In E. Øyen (Ed.), *Comparative methodology: Theory and practice in international social research* (S.96--112). London: Sage.

Glaser, B. G. (1978). *Theoretical sensitivity.* Mill Valley, CA: Sociology Press.

Glaser, B. G. (1992). *Emergence vs. forcing. Basics of grounded theory analysis.* Mill Valley, CA: Sociology Press.

Glaser, B. G. & Strauss, A. L. (1965). T*he discovery of grounded theory. Strategies for qualitative research.* Chicago: Aldine Publishing Company.

Glaser, B. G., & Strauss, A. L. (1979). Die Entdeckung gegenstandsbezogener Theorie: Eine Grundstrategie qualitativer Sozialforschung. In C. Hopf & E. Weingarten (Hrsg.), *Qualitative Sozialforschung* (S. 91-111). Stuttgart: Klett-Cotta.

Glass, G. V., McGaw, B., & Smith, M. L. (1981). *Meta-analysis in social research.* Beverly Hills: Sage.

Greene, J. C., Caracelli, V. J., & Graham, W. F. (1989). Toward a conceptual framework for mixed-methdos evaluation designs. *Educational Evaluation and Policy Analysis, 11 (3)*, 255-274.

Günther, U. L. (1987). Sprachstil, Denkstil und Problemlöseverhalten. Inhaltsanalytische Untersuchungen über Dogmatismus und Abstraktheit. In P. Vorderer & N. Groeben (Hrsg.), *Textanalyse als Kognitionskritik?* (S. 22-45). Tübingen: Narr.

Härtling, P. (o.J.): *Das war der Hirbel.* Stuttgart: Klett-Cotta.

Handl, A. (2002). *Multivariate Analysemethoden. Theorie und Praxis multivariater Verfahren unter besonderer Berücksichtigung von S-PLUS.* Berlin: Springer.

Held, J. (1994). *Praxisorientierte Jugendforschung. Theoretische Grundlagen, methodische Ansätze, exemplarische Projekte.* Hamburg: Argument.

Held, J., Horn, H.-W. & Marvakis, A. (1996). *Gespaltene Jugend. Politische Orientierungen jugendlicher ArbeitnehmerInnen.* Opladen: Leske+Budrich.

Held, J., & Marvakis. A. (1992). Empirische Jugendforschung und ihr Verhältnis zur politischen Bildung. In K.-H. Braun & K. Wetzel (Hrsg.). *Lernwidersprüche und pädagogisches Handeln.* Bericht von der 6. internationalen Ferienuniversität Kritische Psychologie, 24. bis 29. Februar 1992 in Wien (S. 243-256). Marburg: Verlag Arbeit & Gesellschaft.

Howe, K. R. (1985). Two dogmas of educational research. *Educational Researcher, 14 (8)*, 10-18.

Howe, K. R. (1988). Against the quantitative-qualitative incompability thesis or dogmas die hard. *Educational Researcher, 17 (8)*, 10-16.

Huber, G. L. (Ed.) (1992). *Qualitative Analyse. Computereinsatz in der Sozialforschung.* München: Oldenbourg.

Huber, G. L. (1992). Qualitative Analyse mit Computerunterstützung. In G. L. Huber (Hrsg.), *Qualitative Analyse. Computereinsatz in der Sozialforschung* (S. 115-176). München: Oldenbourg.

Huber, G. L. (2001). *Typenbildung in der qualitativen Analyse am Beispiel der Lehrerforschung.* Beitrag zur Tagung der Fachgruppe Pädagogische Psychologie der Deutschen Gesellschaft für Psychologie, Landau 2001.

Huber, G. L. & Kenntner, S. (1988). *Struktur biographischer Selbst-Schemata.* Bericht an die DFG. Tübingen: Arbeitsbereich Pädagogische Psychologie am Institut für Erziehungswissenschaft I der Universität Tübingen.

Huber, G. L., & Marcelo García, C. (1992). Voices of beginning teachers: Computer-assisted listening to their common experiences. In M. Schratz (Ed.), *Qualitative voices in educational research.* (S. 139-156). London: Falmer.

Huber, G. L., & Roth, J. W. H. (2004). *Research and intervention by interviews and learning diaries in inservice teacher training* . Paper presented at the workshop on "Mixed Methods in Psychological Research" (organized by SIG #17, EARLI and the Center for Qualitative Psycholgy, University of Tübingen) at Freudenstadt, Germany, Oct. 21-24, 2004.

Huberman, M. (1987). How well does educational research really travel? *Educational Researcher, 16 (1)*, 5-13.

Jackson, G. B. (1980). Methods for integrative reviews. *Review of Educational Research, 50*, 438-460.

Jacob, E. (1988). Clarifying qualitative research: A focus on traditions. *Educational Researcher, 17 (1)*, 16-24.

Jordell, K. (1987). Structural and personal influence in the socialization of beginning teachers. *Teaching and Teacher Education, 3*, 165-177.

Kelle, U. (Ed.) (1995). *Computer-aided qualitative data analysis. Theory, methods, and practice.* London: Sage.

Kiegelmann, M., Held, J., Huber, G. L. & Ertel, I. (2000). Das Zentrum für Qualitative Psychologie an der Universität Tübingen [24 Absätze]. *Forum Qualitative Sozialforschung; Forum: Qualitative Social Research [On-line Journal], 1/(2).* Verfügbar über: http://www.qualitative-research.net/fqs-texte/2-00/2-00kiegelmannetal-d.htm

Kracauer, S. (1952). The challenge of qualitative content analysis. *Public Opinion Quarterly, 16*, 631- 642.

Lisch, R. & Kriz, J. (1978). *Grundlagen und Modelle der Inhaltsanalyse.* Reinbek bei Hamburg: Rowohlt.

Marcelo García, C. (1991). *El primer año de enseñanza.* Sevilla: Grupo de Investigación Didáctica de la Universidad de Sevilla.

Marcelo García, C. (1992). *Desarrollo profesional e iniciación a la enseñanza. Estudio de caso de un programa de formación para profesores principantes.* Resolución de 30 de sept. de 1992, de la Universidad de Sevilla.

Mathison, S. (1988). Why triangulate? *Educational Researcher, 17 (2)*, 13-17.

Mayring, Ph. (1988). *Qualitative Inhaltsanalyse.* Weinheim: Deutscher Studien Verlag.

Mayring, Philipp (2001, Februar). Kombination und Integration qualitativer und quantitativer Analyse [31 Absätze]. *Forum Qualitative Sozialforschung / Forum: Qualitative Social Research* [On-line Journal], *2*(1). Verfügbar über: http://qualitativeresearch.net/fqs/fqs.htm

Maxwell, J. (2002). Realism and the roles of the researcher in qualitative psychology. In M. Kiegelmann (Ed.), *The role of the researcher in qualitative psychology* (pp. 11-30). Tübingen: Ingeborg Huber Verlag.

Medina, A., Feliz, T., Domínguez, M.-C., & Pérez, R. (2002). The methodological complementariness of biograms, in-depth interviews, and discussion groups. In M. Kiegelmann (Ed.), *The role of the researcher in qualitative psychology* (pp. 169-184). Tübingen: Ingeborg Huber Verlag.

Miles, M. B. (1985). Qualitative data as an attractive nuisance: The problem of analysis. In J. Van Maanen (Ed.), *Qualitative methodology.* (5. Aufl.). (S. 117-134). Beverly Hills: Sage.

Miles, M.B. & Huberman, A.M. (1984). *Qualitative data analysis: A sourcebook of new methods.* Beverly Hills, CA: Sage.

Miles, M. B., & Huberman, A. M. (1994). *Qualitative data analysis. An expanded sourcebook.* (2nd edition). Thousand Oaks, CA: Sage.

Oldenbürger, H. (1981). *Methodenheuristische Überlegungen und Untersuchungen zur "Erhebung" und Repräsentation kognitiver Strukturen.* Göttingen: Dissertation.

Oldenbürger, H. (2004). *Proxim: Repräsentation von Proximitymatrizen - Beschreibung und Analyse.* http://www.liteline.de/~holdenb/fst/nwz/R-PHP/Proxim.R

Phillips, D. (1983). After the wake: Post-positivistic educational thought. *Educational Researcher, 12(5)*, 4-12.

Popko, E. S. (1980). *Key-word-in-context bibliographic indexing. Release 4.0 users' manual.* Cambridge, Mass.: Harvard University, Laboratory for Computer Graphics and Spatial Analysis.

R - Development Core Team (2004). *R: A language and environment for statistical computing.* Vienna: Austria. http://www.r-project.org

Ragin, C. C. (1987). *The comparative method. Moving beyond qualitative and quantitative strategies.* Berkeley: University of California Press.

Schratz, M. (Ed.) (1993). *Qualitative voices in educational research.* London: Falmer Press.

Schweizer, H. (2004). Preparations for the Redemption of the World: Distribution of Words and Modalities in Chapter I of Don Quixote. In M. Kiegelmann & L. Gürtler (Eds.), *Research Questions and Matching Methods of Analysis* (pp.71-108). Tübingen: Ingeborg Huber Verlag.

Shelly, A. L. (1986). The stages of qualitative data analysis. In A. L. Shelly, R. A. Archambault, C. Sutton & P. Tinto (Eds.), *The stages of qualitative research: Researcher thinking facilitated by logic programming.* CASE Technical Report No. 8607, Syracuse, NY: Syracuse University, The Center for Computer Applications and Software Engineering.

Shelly, A. L., & Sibert, E. E. (1985). *The QUALOG user's manual.* Syracuse, NY: Syracuse Univer sity, School of Computer and Information Science.

Shelly, A. L., & Sibert, E. E. (1992). Qualitative Analyse: Ein computerunterstützter zyklischer Prozeß. In G. L. Huber (Hrsg.), *Qualitative Analyse. Computereinsatz in der Sozialforschung* (S. 71- 114). München: Oldenbourg.

Shulman, L. S. (1981). Disciplines of inquiry in education: An overview. *Educational Researcher, 10*, 5-12.

Smith, J. K. (1983). Quantitative versus qualitative research: An attempt to clarify the issue. *Educational Researcher, 12(3)*, 6-13.

Smith, J. K. & Heshusius, L. (1986). Closing down the conversation: The end of the quantitative-qualitative debate among educational researchers. *Educational Researcher, 15(1)*, 4-12.

Strauss, A., & Corbin, J. (1990). *Basics of qualitative research. Grounded theory procedures and techniques.* Newbury Park, CA: Sage.

Tashakkori, A., & Teddlie, C. (1998). *Mixed methodology: Combining qualitative and quantitative approaches* (Applied Social Research Methods, No. 46). Thousand Oaks: Sage.

Tashakkori, A., & Teddlie, C. (Eds.) (2003). *Handbook of mixed methods in social and behavioral research.* Thousand Oaks: Sage.

Tesch, R. (1990). *Qualitative research. Analysis types and software tools.* New York: Falmer.

Tesch, R. (1992). Verfahren der computerunterstützten qualitativen Analyse. In G. L. Huber (Hrsg.), *Qualitative Analyse. Computereinsatz in der Sozialforschung* (S. 43-70). München: Oldenbourg.

Thomas, W. I., & Znanecki, F. (1918). *The Polish peasant in Europe and America.* Boston: Badger.

Tuthill, D. & Ashton, P. (1983). Improving educational research through the development of educational paradigms. *Educational Researcher, 12(10)*, 6-14.

Villar, L. M. (Ed.) (1981). *Las prácticas de enseñanza.* Sevilla: ICE de la Universidad.

Villar, L. M., & Marcelo, C. (1992). Kombination qualitativer und quantitativer Methoden. In G. L. Huber (Hrsg.), *Qualitative Analyse. Computereinsatz in der Sozialforschung* (S. 177-218). München: Oldenbourg.

Weber, M. (1988). *Gesammelte Aufsätze zur Wissenschaftslehre.* Tübingen: Mohr.

Weber, R. P. (1985). *Basic content analysis.* Sage University Paper series on Quantitative Applications in the Social Sciences, series no. 07-049. Beverly Hills: Sage.

Yin, R. K. (1989). *Case study research. Design and methods.* Applied Social Research Methods Series, Vol. 5. Newbury Park, CA: Sage.

Zabalza, M. A. (1991). *Los diarios de clase. Documento para estudiar cualitativamente los dilemas prácticos de los profesores*. Barcelona: Promociones y Publicaciones Universitarias, S.A.

Subject Index

Author Index