Structuring a Team-Based Approach to Coding Qualitative Data

Lindsay Giesen1 and Allison Roeser1

Abstract
Improvements to qualitative data analysis software (QDAS) have both facilitated and complicated the qualitative research process. This technology allows us to work with a greater volume of data than ever before, but the increased volume of data frequently requires a large team to process and code. This paper presents insights on how to successfully structure and manage a team of staff in coding qualitative data. We draw on our experience in team-based coding of 154 interview transcripts for a study of school meal programs. The team consisted of four coders, three senior reviewers, and a lead analyst and external qualitative methodologist who shepherded the coding process together. Lessons learned from this study include: 1) establish a strong and supportive management structure; 2) build skills gradually by breaking training and coding into “bite-sized” pieces; and 3) develop detailed reference materials to guide your coding team.

Keywords
teamwork strategies, qualitative analysis, management strategies, qualitative data analysis software, QDAS

Date received: January 3, 2020. Received revised September 22, 2020; Accepted: September 28, 2020

Background
The use of qualitative data analysis software (QDAS) has expanded over the last two decades, allowing researchers to code and analyze large volumes of data at a faster pace than they used to by hand (Woods et al., 2016). These software products also allow multiple staff to code data simultaneously, and then seamlessly merge their work for additional analysis. While the use of such software has increased dramatically, few empirical studies have looked at how to successfully manage a research team that uses QDAS to code and analyze large volumes of data (Fielding & Lee, 1998, 2002; Gilbert, 2002; Mangabeira et al., 2004; Marshall, 2002; Silver, 2010).

Despite the efficiencies generated by new software, we have found that teams using QDAS to code qualitative data require substantial upfront training and ongoing support in order to accurately and reliably apply codes. Such training may include a pilot testing phase where staff apply codes to sample data and then discuss their work to identify areas of agreement and divergence in coding. Those discussions can then lead the team to refine and improve the coding scheme and/or the coding process. As discussed later, we treated the coding scheme and codebook as living documents; though we initially trained the team to use a set coding scheme developed by our lead analyst and an external qualitative methodologist, new insights from staff discussions of the data led to changes (and improvements) over time.

The qualitative methods literature provides some guidance on what to consider when building a team for qualitative coding and analysis, but few specific examples of the mechanics of managing such a team. Bazeley and Jackson (2013) pose useful questions for teams to consider early in the process, such as: “What work will be done independently, in pairs, or in groups?” and “What decision-making models (e.g., consensus, democratic vote, or team leader directives) are in play to guide team progress during coding?” However, they do not explain the benefits or drawbacks of the different possibilities to help project managers determine the best path forward.

We also have some examples in the literature of how to structure a team of coders. For a study of Don’t Ask, Don’t Tell, the 1994 law banning gays and lesbians from openly...
serving in the U.S. military, Robins and Eisen (2017) describe how they assigned staff to each of three primary tasks—qualitative data collection, analysis, and management—and the pros and cons of that approach. Their model differed from those reviewed by Cornish et al. (2013) in their high-level discussion of different ways to structure teams for qualitative coding. One of the models they summarized is the one presented by Hall et al. (2005). That model most closely resembles our own in that their team had a hierarchy with layers of support for staff, but did not make decisions about the coding scheme or process in a top-down manner. Instead, both were continuously refined through ongoing team reflection and discussion.

While these resources provide insight into the possible structures, benefits and challenges of team-based coding, the predominantly high-level discussions still leave questions about how to actually manage a team of coders. The example provided by Hall et al. began to answer some of those questions, and this article attempts to build on that by providing additional detail on how to successfully manage and train a team with clear roles and timelines and ongoing support.

In this article we draw on our experiences with a research study conducted on behalf of the U.S. Department of Agriculture’s Food and Nutrition Service (FNS), which explored how schools, school districts, and State agencies collect data on the National School Lunch Program (NSLP) and School Breakfast Program (SBP). The NSLP and SBP are Federally-funded meal programs providing lunch and breakfast every school day to millions of students across roughly 100,000 public and nonprofit private schools and residential childcare institutions (RCCIs) around the country (Economic Research Service, 2020). The primary research objective was to document the processes and challenges for collecting and submitting data on these school meal programs. Through site visits, we conducted 154 in-person interviews across four States from February through May 2018. Respondents included: 1) staff from four NSLP State agencies, 2) staff from four Supplemental Nutrition Assistance Program (SNAP)5 State agencies, 3) food service directors at 39 school districts, and 4) school food managers representing 119 schools.5 Data collection ended in May 2018, and our team had to complete all coding within 6 weeks, by early July.

We opted for a team-based approach to coding for this study in response to timing constraints and the large quantity of data to be processed; one person could not code all 154 transcripts within 6 weeks. We estimated that the work required four coders, each one assigned to code all interviews from a different State in the study. We selected our coding team from among the staff who conducted site visits because they were already familiar with the study objectives and the types of data collected. The junior members of the site visit teams coded the interview transcripts of the State they visited, while three of the site visit leads reviewed the coders’ work for quality.3 This team-based approach came with the challenge of ensuring that staff agreed on which codes to apply to each segment of text. However, the study benefitted from the different perspectives and suggestions that staff brought to the work, and we used those suggestions to refine the coding scheme and the coding process along the way.

In the 6 weeks following data collection, our team coded all 154 interview transcripts in NVivo4 using a complex, two-tiered coding scheme. The software allowed the team to code a large quantity of data in a short amount of time, and staff received substantial upfront and ongoing training, support, and oversight to produce the strongest output possible. This article does not discuss how to develop a coding scheme to organize qualitative data or how to measure and facilitate intercoder reliability,5 because much has been written about those topics already (Burla et al., 2008; Campbell et al., 2013; Cohen, 1960; McHugh, 2012). Comparatively little has been written about how to manage a team-based coding effort, and so we offer three lessons learned from our experience on this study: the importance of 1) establishing a strong and supportive management structure; 2) building skills gradually; and 3) developing detailed reference materials to guide the coding team.

Lesson Learned #1: Establish a Strong Management Structure

Our management structure and approach was key to our success and possessed three critical elements: 1) clear roles and responsibilities; 2) regular communication and oversight; and 3) an overall timeline with intermediate milestones. In this section we discuss how each element facilitated our success.

Clear Roles and Responsibilities

While working on other studies, we witnessed the confusion and mistakes that happen when coding as part of a team. First, work may not be adequately coordinated across staff, leading to confusion about who is responsible for coding each piece of data. Second, teams may have a group of coders but no one to review the quality of the work. Third, teams of coders may not have a clear lead analyst to make final determinations about changes to the coding scheme or coding process. Any of these scenarios can result in inefficient and inadequate coding.

We created four roles on our team to provide layers of support and a clear division of responsibility:

1. Lead Analyst.

This person consulted with an external qualitative methodologist to create the coding scheme and all reference materials, manage the NVivo database,6 and train the coding team. In addition, they provided ongoing guidance and support to all staff throughout the coding process. The lead analyst had over a decade of experience with analyzing qualitative data and using NVivo, and possessed a thorough understanding of the study objectives and the data collected. This person also served
as one of the original site visit leads as well as a senior reviewer.

2. External Qualitative Methodologist.

This person had no prior involvement with the research study, and was engaged to objectively review the coding scheme and provide guidance on the coding process. The team recruited this person to the study immediately after data collection ended to help define the details of the coding and analysis process. They collaborated with the lead analyst to develop the coding scheme and reference materials. The external qualitative methodologist also helped lead the staff training and provided ongoing support and feedback to the team.

3. Coders.

These four staff coded all data, and had varying levels of experience with qualitative coding. As noted earlier, study data were collected from four different States, and each of the four coders was assigned to code all data from one of the four States in the study; no two staff coded data from the same State.

4. Senior Reviewers.

The three senior reviewers on our team were subject matter experts who understood the study objectives, the data collected, and the final products we needed to create using the coded data. Each senior reviewer was paired with a coder to review their work and provide feedback and guidance (one reviewer was paired with two coders; two reviewers were each paired with a single coder). The lead analyst served as one of the three reviewers.

Having four distinct and defined roles proved invaluable during the coding process, because it established layers of support and clarity on staff responsibilities. Each team member had a specific assignment as well as a primary point of contact for support. To the coders, their senior reviewer was their primary point of contact for any questions. In turn, the senior reviewers contacted the lead analyst for guidance; the lead analyst consulted with the external qualitative methodologist.

Regular Communication and Oversight

Communicating early and often is crucial during a team-based coding effort. We met as a group roughly every 10 days throughout the 6-week coding process. These group meetings occurred by phone, and they were scheduled in addition to any meetings between the coder-senior reviewer pairs as part of the arbitration process (discussed in greater detail in Lesson #2).

We held two types of regular group meetings, both facilitated by the lead analyst: 1) training meetings prior to the start of coding each type of transcript, and 2) check-in meetings early in the coding process for each type of transcript. Training meetings occurred as the team was ready to move on to coding the next type of transcript. These meetings began with a reflection on what the team learned from coding the earlier transcripts and identifying any themes they saw in the data. Similar to what Hall et al. (2005) described, these team reflections often led to revisions to the coding scheme to better capture what the coders and senior reviewers observed. For example, when the coder-reviewer pairs expressed that they felt they were over-utilizing a particular code, the team discussed the code and realized that the code definition was too general, and staff were applying it to more pieces of text than was intended. With the team’s input during the call, the lead analyst made two improvements: 1) split the code into several discrete sub-codes to better capture nuances in the text; and 2) revised the codebook definitions to provide greater clarity on when to apply those codes. In this way, the coding scheme served as a living document that was continuously revised with the team’s input throughout coding. The balance of each training meeting was spent reviewing the coding scheme, reference materials, and data for the new type of transcript to be coded.

The check-in meetings occurred within 2–3 days of the training on each type of transcript. During these meetings the team discussed any challenges staff faced in coding their first few transcripts. The meetings helped the lead analyst to uncover and adjudicate coding differences early on in the process. We also used the time to determine whether the timeline was realistic given the amount of time it took the coders to code their first few transcripts, and adjusted deadlines as needed.

Set an Overall Timeline With Intermediate Milestones

Our task was to code 154 interviews within 6 weeks. Sensing that the big picture deadline was both nebulous and daunting, we broke the timeline into intermediate milestones. Rather than tell the team to code everything within 6 weeks, we revised the message to: “Code all of the school transcripts in the next 10 business days.” That short-term deadline was easier for staff to grasp. When that deadline came, we set a new short-term deadline for the school district transcripts, and later for the State transcripts. Setting those milestones for the completion of each phase of coding helped to maintain momentum and created a sense of accomplishment when we reached them. Those intermediate milestones also helped to pace the team’s work and ensured that no one fell behind.

Lesson Learned #2: Build Skills Gradually

The primary challenge in structuring the training was conceptualizing how best to prepare the team to understand and apply 118 codes to the four types of respondent transcripts. Further complicating matters, only a subset of the codes applied to each transcript (e.g., codes pertaining to school-level processes applied only to the school transcripts). Yet another challenge
was that several coders had never used NVivo, and needed to learn the basics of the software. We designed the training to address these challenges, and also required the attendance of the senior reviewers to ensure that everyone began with a similar understanding and skillset. The lead analyst and the external qualitative methodologist designed and facilitated the training.

Ultimately, we broke the training and coding into “bite-sized” pieces, beginning with a one-day, in-person training. Table 1 displays our training agenda and discusses the value of each session. One of the most important goals of this training was to imbue the team with a conceptual framework for when, why, and what to code (and when, why, and what not to code). Qualitative data often contains a lot of “noise,” or unnecessary details, and we strove to teach our team to ignore the noise and locate the critical data. To that end, we reviewed the study objectives and discussed the written products that we would later create with the results of the coding and analysis. We believed that understanding the study’s goals and deliverables would help the team to be discerning when coding the interview transcripts.

After reviewing the study objectives, we reviewed the codes for - and practiced coding - the school transcripts, which were the shortest and most straightforward of the interviews. Starting with the simplest transcripts proved useful in building the team’s understanding of the project, as well as their confidence with the coding process, before moving to the more complicated school district and State transcripts.

Our experiences on other studies taught us that teaching coders how to code using software without first teaching them when and why to code leads to inadequate work, particularly among novice coders. How to code and the strategy behind coding are two separate cognitive tasks, and teaching both at once can be overwhelming for staff. For that reason, the team first practiced applying the codes to a paper copy of a transcript, which taught them the strategy behind coding and forced them to slow down and thoughtfully consider which codes to apply. The coders wrote on paper transcripts which codes they believed applied to each section of text, and then we went through the transcripts and coding as a team. At the end of this initial training day, we demonstrated how to navigate NVivo, and gave the team time to practice applying the same codes to the same transcript from within the software.

Following both practice coding sessions, on paper and within NVivo, we debriefed as a group. The lead analyst walked the team through the transcript one section at a time, and asked staff whether the segment of text needed to be coded or was just “noise,” and what codes, if any, coders applied to the data. Those discussions revealed the team’s lingering questions and uncertainties about both the coding process and the software, and provided an opportunity to offer them clarity and support. Where staff applied different codes to a piece of text, the lead analyst facilitated a discussion to gather the team’s input on which was the most appropriate code. The senior reviewers’ and lead analyst’s preferences regarding the application of codes carried the greatest weight because they would perform the subsequent analysis and reporting, but all opinions were heard.

The feedback from the team strengthened the coding scheme and reference materials. Both the coders and senior reviewers used their recollections of the interviews they conducted during site visits to suggest revisions to the coding scheme and codebook. For example, some schools in very low-income areas offered free meals to all students without requiring a school meal application from their parents. The coders and senior reviewers who visited those schools pointed out that some of the codes about school meal applications would not apply to those cases, and the lead analyst added notes to the codebook to explain when a code may not apply.

At the end of this one-day training we assigned “homework.” We gave the coders two days to independently code one of their assigned transcripts in NVivo, and
asked the senior reviewer paired with the coder to assess the work and provide feedback. The coder-reviewer pairs deliberated over whether codes were applied correctly and whether any text needed to be un-coded or re-coded in what we called the “arbitration process.” The pairs largely reached consensus on their own, and noted any areas of disagreement.

The entire team reconvened two days after the initial training for a check-in meeting facilitated by the lead analyst. The coder-reviewer pairs shared their perspectives on the aspects of the coding and arbitration process that went smoothly or proved challenging, and any areas of confusion or disagreement. The latter highlighted the need for additional revisions to the coding scheme and codebook to provide greater clarity to the team and produce a more comprehensive list of codes to capture valuable data; the lead analyst made those changes following the check-in meeting. For example, some coders and senior reviewers disagreed about whether a code about school meal processes applied to manual processes, automated processes, or both. Following a discussion, the team decided it was important to have different codes to capture manual and automated processes, and the lead analyst made those changes to the coding scheme and codebook. After the meeting, the coders worked through the remaining school transcripts on their own, and the senior reviewers checked an early sample of each coder’s work to ensure codes were being applied correctly and consistently.

As Figure 1 illustrates, we repeated this train-practice-code sequence for each type of transcript (i.e., after the team coded all school transcripts, they were trained on the school district transcripts; after they coded the school district transcripts, they were trained on the State transcripts). After these trainings, the team followed the same process as before: the coders had two days to practice coding independently, the senior staff reviewed the work, and the coder-reviewer pairs went through the arbitration process. Breaking the process into manageable pieces cultivated the team’s skills over time. An unexpected benefit was that it familiarized the coders with the data collected from each type of transcript to where they could more easily identify trends and outliers.

We continued to hold check-in meetings two days after each training because they served as an invaluable source of learning. Staff listened to each other’s questions, and learned from the answers provided. As the team grew more comfortable with the data and the coding process, they volunteered thoughtful solutions to their colleagues’ problems. Those check-ins also allowed us to identify and resolve problems early on, and revise the codebook as needed.

**Lesson Learned #3: Develop Detailed Reference Materials**

When multiple staff are responsible for coding, project leaders need to ensure a shared understanding of when and how to
apply codes. In preparation for training the team on the coding process for this study, the lead analyst developed the following reference materials:

1. Detailed codebook;
2. Blank interview guides with codes applied; and
3. Interview transcripts with codes applied.

These materials proved critical to fostering a shared understanding of the coding scheme and measurably improved inter-coder reliability.7

The codebook served as the primary reference document. The lead analyst built the codebook in Microsoft Excel8 and included the following fields:

- Code name;
- Definition;
- The respondent types for whom we expected to apply each code (e.g., school respondents);
- Source question from the interview guide (when applicable); and
- Notes, including guidance on when to double-code and when not to apply a code.

Creating the codebook in Excel allowed coders to filter the fields and display only the codes that pertained to a particular respondent type (e.g., codes about State-level processes would apply only to the State transcripts). This codebook was treated as a living document; we revised the codebook throughout the train-practice-code process to capture patterns in the data and to incorporate feedback from our team. For example, during a check-in meeting staff commented that two codes were so similar that they had trouble differentiating between them, and so we merged them to minimize confusion.

After developing the codebook, the lead analyst applied the codes to the blank interview guides for each respondent type. For example, we took the interview guide for the school interviews and used the comment feature in Microsoft Word to indicate which codes were expected to apply to each question or section of that school guide; we did the same for the school district interview guide and State interview guides for each of those respective trainings. We provided these templates to the coders each time they began coding a new type of interview transcript so that they had an additional visual aid to illustrate which code applied to each interview question or section.

Finally, we provided a sample interview transcript with the codes applied. This gave the coders a sense of when to apply codes to a real transcript, and helped them learn that a code may be applied multiple times to a transcript if the conversation touched on a subject more than once. It also allowed us to illustrate the concept of double- and triple-coding, wherein more than one code is applied to a section of text.

Regardless of the number of coders and their levels of experience, teams need detailed reference materials to code data accurately and consistently (Bazeley, & Jackson, 2013). The materials not only guide the coders, but also the senior staff reviewing each coder’s work. Without this guidance, even a well-trained team will apply codes inconsistently, thereby creating analytic challenges later on.

**Conclusion**

When operating under tight deadlines, it may not seem feasible to hold multiple trainings or regular check-in meetings. Our team-based coding effort proved that not only is it possible to build those into a short timeline, but that providing ongoing training, oversight and communication can actually facilitate a timely completion rather than hinder it. Too often, those leading a team in coding qualitative data mistakenly assume that a team simply needs a codebook and a tutorial on using qualitative data analysis software to be successful. While that is a useful starting point, it is far from adequate. No two coders think exactly alike, regardless of their years of experience, and the more complicated the data or the coding scheme is, the more room there is for inconsistency and error. Providing a strong management structure, ongoing training and oversight, and developing detailed reference materials all help to foster a shared approach and understanding of the material.

**Authors’ Note**

These insights were previously presented at the Ethnographic & Qualitative Research Conference, February 25–26, 2019, Las Vegas, Nevada, United States.

**Acknowledgments**

The authors would like to thank Melissa Rothstein and Cynthia Robins for their leadership and guidance during this research study, and for their support during the development of this article.

**Declaration of Conflicting Interests**

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

**Funding**

The author(s) received no financial support for the research, authorship, and/or publication of this article.

**ORCID iD**

Lindsay Giesen https://orcid.org/0000-0003-1927-7408

**Notes**

1. SNAP is the program formerly known as “food stamps.”
2. Some school food managers represented more than one school.
3. Staff with less than 10 years of experience in on-site qualitative data collection and qualitative analysis were the junior site visit staff, and subsequently served as coders. Staff with 10 or more years of relevant experience served as site visit leads and senior reviewers.
4. NVivo 11 was the qualitative data analysis software used in this study.
5. The application of the same codes to the same pieces of text by different staff.
6. We do not currently use an NVivo server, which means that the lead analyst created individual NVivo databases for each coder and later merged them.

7. We used Cohen’s kappa as our measure of intercoder reliability.

8. A codebook can also be programmed into NVivo.

9. Some coding topics may arise at any point in any transcript, while others are only expected to arise in response to particular questions.

References

Bazeley, P., & Jackson, K. (2013). *Qualitative analysis with NVivo* (2nd ed.). Sage.

Burla, L., Knierim, B., Barth, J., Liewald, K., Duetz, M., & Abel, T. (2008). From text to codings: Intercoder reliability assessment in qualitative content analysis. *Nursing Research, 57*(2), 113–117.

Campbell, J. L., Quincy, C., Osserman, J., & Pedersen, O. K. (2013). Coding in-depth semistructured interviews: Problems of unitization and intercoder reliability and agreement. *Sociological Methods and Research, 42*(3), 294–320. https://doi.org/10.1177/0049124113500475

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement, 20*, 37–46.

Cornish, F., Gillespie, A., & Zittoun, A. (2013). Collaborative analysis of qualitative data. In U. Flick (Ed.), *The Sage handbook of qualitative data analysis* (pp. 79–93). Sage.

Economic Research Service. (2020, October 1). *National School Lunch Program*. Retrieved October 14, 2020, from https://www.ers.usda.gov/topics/food-nutrition-assistance/child-nutrition-programs/national-school-lunch-program/

Fielding, N., & Lee, R. (1998). *Computer analysis and qualitative research*. Sage.

Fielding, N., & Lee, R. (2002). New patterns in the adoption and use of qualitative software. *Field Methods, 14*, 197–216.

Gilbert, L. S. (2002). Going the distance: ‘Closeness’ in qualitative data analysis software. *International Journal of Social Research Methodology, 5*(3), 215–228.

Hall, W., Long, B., Bermbach, N., Jordan, S., & Patterson, K. (2005). Qualitative teamwork issues and strategies: Coordination through mutual adjustment. *Qualitative Health Research, 15*(3), 394–410. https://doi.org/10.1177/1049732304272015

Mangabeira, W., Lee, R., & Fielding, N. (2004). Computers and qualitative research: Adoption, use, and representation. *Social Science Computer Review, 22*, 167–178.

Marshall, H. (2002). Horses for courses: Facilitating postgraduate research students’ choice of computer assisted qualitative data analysis. *Contemporary Nurse, 13*, 29–37.

McHugh, M. L. (2012). Interrater reliability: The kappa statistic. *Biochemia Medica, 22*(3), 276–282.

Robins, C., & Eisen, K. (2017). Strategies for the effective use of NVivo in a large-scale study: Qualitative analysis and the repeal of Don’t Ask, Don’t Tell. *Qualitative Inquiry, 23*(10), 768–778. https://doi.org/10.1177%2F1077800417731089

Silver, C. (2010). Research design for longitudinal case-study project: Tracking the use of software in real projects using different methodologies (QUIC Briefing Paper #1). Qualitative Innovations in CAQDAS (QUIC).

Woods, M., Paulus, T., Atkins, D., & Macklin, R. (2016). Advancing qualitative research using qualitative data analysis software (QDAS): Reviewing potential versus practice in published studies using ATLAS.ti and NVivo, 1994–2013. *Social Science Computer Review, 34*(5) 597–617.