THEMATIC ANALYSIS

A PRACTICAL GUIDE

VIRGINIA BRAUN AND VICTORIA CLARKE

UNIVERSITÄT ST. GALLEN
HOCHSCHULE FÜR WIRTSCHAFTS-,
RECHTS- UND SOZIALWISSENSCHAFTEN
BIBLIOTHEK

SAGE

Los Angeles | London | New Delhi
Singapore | Washington DC | Melbourne

SAGE

Los Angeles | London | New Delhi
Singapore | Washington DC | Melbourne

SAGE Publications Ltd
1 Oliver's Yard
55 City Road
London EC1Y 1SP

SAGE Publications Inc.
2455 Teller Road
Thousand Oaks, California 91320

SAGE Publications India Pvt Ltd
B 1/I 1 Mohan Cooperative Industrial Area
Mathura Road
New Delhi 110 044

SAGE Publications Asia-Pacific Pte Ltd
3 Church Street
#10-04 Samsung Hub
Singapore 049483

© Virginia Braun and Victoria Clarke 2022

Apart from any fair dealing for the purposes of research, private study, or criticism or review, as permitted under the Copyright, Designs and Patents Act, 1988, this publication may not be reproduced, stored or transmitted in any form, or by any means, without the prior permission in writing of the publisher, or in the case of reprographic reproduction, in accordance with the terms of licences issued by the Copyright Licensing Agency. Enquiries concerning reproduction outside those terms should be sent to the publisher.

Editor: Amy Maher
Editorial assistant; Esmé Carter
Assistant editor, digital: Sunita Patel
Production editor: Rachel Burrows
Marketing manager: Camille Richmond
Cover design: Wendy Scott
Typeset by: C&M Digitals (P) Ltd, Chennai, India
Printed in the UK

Library of Congress Control Number: 2021934969

British Library Cataloguing in Publication data

A catalogue record for this book is available from the British Library

ISBN 978-1-4739-5323-9
ISBN 978-1-4739-5324-6 (pbk)

At SAGE we take sustainability seriously. Most of our products are printed in the UK using responsibly sourced papers and boards. When we print overseas we ensure sustainable papers are used as measured by the PREPS grading system. We undertake an annual audit to monitor our sustainability.

3

EXPLORING THIS WORLD IN DETAIL

DOING CODING

Chapter Three overview

You've taken the lay of the (new) land you've found yourself in; you've had some initial observations about what things are like; you've spent some time reflecting on your impressions, both overall, and specific aspects you've encountered. Now is the time for a more detailed and systematic exploration. In reflexive TA terms, you're about to start *coding*. Coding is a process and practice across many forms of qualitative analysis, but one where there is quite a bit of variation, even across TA (in Chapter Eight, we discuss different conceptualisations of coding in TA). In the first half of the chapter, we explain how coding is understood within reflexive TA, and clarify some key concepts. In the second half, we discuss and demonstrate how you *do* coding in reflexive TA.

PREPARING FOR CODING

Before you get into coding, you need to understand some key things *about* codes and coding. In reflexive TA, a code is the smallest unit of your analysis. Your codes form the building blocks of your analysis; from these you will go on to develop your themes.

KEY CONCEPT Codes are the building blocks of analysis in reflexive TA, capturing meaning relevant to the research question.

Your codes capture specific and particular meanings within the dataset, of relevance to your research question. They have succinct *labels* that evoke the data content. In writing about reflexive TA, we've often conflated the concept of a **code *label*** and a code, through simply using the term code for both, but it's useful to be clear that a richer analytic idea often sits 'behind' the short code label, and this is what the code captures. Although codes in reflexive TA focus on singular ideas, these don't necessarily lack depth. This is because coding is more than just a way to *reduce down* the detail of the data. Codes often also provide a pithy take on what is of analytic interest **in the data** – they offer some interpretation. This means codes can range from the more **summative** or **descriptive** to the more interpretative or **conceptual**. Coding can capture a range of meaning abstraction, from the semantic or **manifest** content of the data, to latent or underlying meaning. The coding process can also be driven by a more inductive or a more deductive orientation to data. We will describe all these elements in a bit more detail, below; before we do, we briefly synopsise the three core concepts of coding for reflexive TA (see Box 3.1).

KEY CONCEPT A code label succinctly summarises the analytic ideas and data meanings captured by a code.

ALERT Codes both reduce the data content and provide an analytic take on what is of interest.

Box 3.1

Coding, codes and code labels in reflexive TA – a quick guide

Coding The *process* of exploring the diversity and patterning of meaning from the dataset, developing codes, and applying code labels to specific segments of each data item.

Code An *output* of the coding process; an analytically interesting idea, concept or meaning associated with particular segments of data; often refined during the coding process.

Code label An *output* of the coding process; a succinct phrase attached to a segment of data, as a shorthand tag for a code; often refined during the coding process.

CODING IS A SYSTEMATIC PROCESS

In reflexive TA, coding is a *process* – it's how you work with data in this phase – and codes and code labels are *outputs* of this process.[1] When you engage in this process in a thorough and rigorous way, your codes should set you up well for the next phase of initial theme development. Where familiarisation is engaged-but-not-yet-systematic, coding is engaged-*and*-systematic. The coding *process* involves reading each data item closely, and tagging all segments of the text where you notice any meaning that is potentially relevant to your research question with an appropriate code label. This means some segments of data will not be tagged with any codes, because there isn't anything of relevance to the research question.[2] In contrast, as the coding example in Table 3.3 later in this chapter shows, some segments might be tagged with many *different* codes – because a number of *different* meanings are evident in a particular segment of data. You use a different code label for each different meaning because coding is a process for parsing out diversity of meaning. Individual codes shouldn't capture multiple meanings. Sometimes, your code applies to just

ALERT Coding in reflexive TA is a *process* – an activity you are engaged in – with codes as the *outcome* of that process.

PRACTICE POINT Make sure each different meaning you identify has its own code, so you're using coding to parse out the diversity of meanings from the data; codes should not capture multiple meanings.

[1]Not all forms of TA conceptualise coding and codes as, respectively, process and outputs. We conceptualise themes in a similar way to codes, as meanings developed *through* our analytic engagement with our data. See 'Variation across TA approaches: core concepts' in Chapter Eight for more discussion of these different conceptualisations.

[2]For reflexive TA, we do *not* advocate line-by-line coding in the traditional, grounded theory-developed sense of coding each single line of text (see Glaser, 1978). Instead, you only need to code data relevant to the research question – broadly framed.

a few words in a data item; other times, you might affix a code to a whole paragraph, or an even longer segment of text. Often the meaning you identify will be very narrow or 'tight'; other times it might be broader or 'loose'. There's no *right* or specific level here, as exactly how you code is ultimately guided by your research question and purpose. But what doesn't vary is that, with coding, you're aiming to generate lots of different codes that differentiate between meanings. It's also crucial to remember that *codes are not themes*, and so you're trying to capture a singular or particular idea through coding, not a multi-faceted one.

Coding often helps us to shift to fully engaging with the data *as data* – as materials we are grappling with to make analytic sense of, to address a specific question – rather than straightforward sources of information. A systematic coding process is important for two reasons – insight and rigour:

Insight, because your analysis typically becomes deeper, more interesting, and less obvious through a repeated process of close engagement. Insight takes time, and this disciplined practice forces us to not 'leap ahead' into developing themes. We regard thinking about themes as very important to resist during coding, not only because themes are typically an *outcome* of later phases of the analytic process in reflexive TA, but because this risks foreclosing analysis (Braun & Clarke, 2021c). Your analytic insights will evolve throughout the phases of reflexive TA.

Rigour, because coding ensures a systematic engagement with meaning and patterning *across* the entire dataset, so theme development is based on a robust and detailed analytic interrogation. Systematic engagement also helps avoid the accusation of *cherry-picking*. This is the idea that qualitative analysts select 'patterns' in their data to fit their own predetermined ideas about what meanings are evident, through poor analytic practice and/or partial data engagement (Morse, 2010) (see also Box 4.5 in Chapter Four).

ALERT A 'quick and dirty' analysis can result in *analytic foreclosure* – stopping analysis after an only superficial engagement with the data and producing a set of themes that don't realise the full potential of the data.

CODING IS ORGANIC, EVOLVING AND SUBJECTIVE

Coding is an *organic* and *evolving* process, an open process, in reflexive TA. Coding begins without any list or set idea of what codes will be used.[3] Codes can *evolve* through the coding process – shifting as your understanding of meaning in relation to the dataset develops through coding. Codes can be sharpened or expanded, so that the meaning they capture is associated with more than just one segment of data. The evolution of codes and code labels

ALERT Codes should usually connect to more than one segment of data; coding is about starting to capture repetition of meaning.

[3]There is *no* codebook or framework that guides the process of coding in reflexive TA (as there is in some other forms of TA – we outline the different approaches in Chapter Eight).

might sound like a flaw in the process, but it reflects *good practice* in reflexive TA, for two reasons. First, as noted already, insight develops through analytic engagement. Earlier codes *may* lack nuance, subtlety or depth; evolving rather than fixing your codes early in – or even before – analysis ensures more nuanced coding, and that richer analytic insights are systematically captured. Second, as TA is about identifying *patterns* of meaning, your task in coding is not only to demarcate differences, but to start to notice shared or similar meaning. An evolving coding process supports this practice, through allowing you to shift and change the code labels to capture closely-related ideas or meanings within single codes, where appropriate. You're looking for some repetition in coding for most, but not necessarily all, codes. A code can still be useful and relevant to addressing your research question even if it occurs only once, since themes are typically developed from *multiple* codes that identify different facets of the meaning focus of a theme.

ALERT Codes and coding labels can shift and change throughout the coding process, to better evoke and differentiate between the range of meanings in the data.

We also view coding as a *subjective* process shaped by what we bring to it. Coding is a process of *interpretation* – or meaning-making[4] – and researcher subjectivity fuels that process. As discussed in Chapter One, in keeping with an overall Big Q qualitative orientation, we view analysis as a process of meaning-making rather than truth-seeking or discovery (Braun & Clarke, 2021c). This means subjectivity is a strength, rather than a weakness or a source of 'bias' (Nadar, 2014). The coding achieved through this open, organic process can be stronger or weaker, depending on the depth and rigour of engagement. But it cannot be understood simply as right or wrong. This is because of the subjective and situated nature of the analytic process in reflexive TA, which means that different coders will notice and make sense of data in different ways.[5] Having only one person coding – usually the researcher – is normal practice, and indeed good practice, for reflexive TA. Where multiple coders *can* be useful is in developing *richer* and more complex insights into the data. But it is *not essential*, or even more desirable than having a single coder. If multiple coders *are* part of your reflexive TA process, the purpose is to collaboratively gain richer or more nuanced insights, *not* to reach agreement about every code (we discuss this further in Chapter Eight).

PRACTICE POINT A single coder is normal – and good – practice in reflexive TA.

INDUCTIVE AND DEDUCTIVE ORIENTATIONS TO DATA CODING

In Chapter One, we signalled different variations in reflexive TA, including different orientations to data (see Table 1.2). It is during coding that these variations *start* to take shape.

[4]Meaning-making and interpretation are core in the reflexive TA analytic process, so important that we dedicate the whole of Chapter Seven to interpretation.

[5]This means we *do not* recommend using multiple coders as a way to guarantee a 'true' or 'accurate' analysis. Some other versions of TA do recommend this, which we discuss in Chapter Eight.

ALERT Coding doesn't have to be either inductive or deductive, although it can be; your analysis can have elements of both orientations. What's most important is that your coding orientation fits your purpose.

One really important dimension in coding and theme development is the way you tackle the question of where and how meaning is noticed: this ranges from inductive (data-driven) to deductive (researcher- or theory-driven) orientations.[6] As we noted in Chapter One, it's more a spectrum than a dichotomy, and coding of a dataset can encompass both types. What is important is to recognise how you're approaching the meaning-making (coding) process, and how that shapes the things that you notice about the data.

An *inductive* orientation takes the dataset as the starting point for engaging with meaning. At some 'pure' level, it would only capture that meaning – it's evoked by the idea that qualitative research can 'give voice' to participants and tell their stories in a straightforward way. In Big Q qualitative analysis, the subjective and embedded process makes pure induction impossible: we bring with us all sorts of perspectives, theoretical and otherwise, to our meaning-making, so our engagement with data is never purely inductive. We cannot simply give voice, because who we are always shapes what we notice about our data and the stories we tell about them (Fine, 1992). This is why reflexivity matters (as discussed in Chapter One). Consider your disciplinary training so far: even if you don't realise it, you will be starting to think like, and view the world as, psychologists, or sociologists, or economists, or whatever it is you might be, or be training to be – and unless you spend lots of time in an inter-disciplinary environment, it can be hard to notice how much that constrains and limits your sense-making.

ALERT In Big Q qualitative, an inductive orientation is never 'pure' because of what we bring to the data analytic process, as theoretically embedded and socially positioned researchers.

This is not to say an inductive orientation is not possible! In reflexive TA, your analytic process *can* emphasise the data meanings, and aim to be grounded in and depart from these meanings *as your starting point*. If you want to figure out whether an inductive orientation fits your project, a useful basic question is: am I interested in things like the experiences, perspectives, and meanings of the participants? If the answer is an emphatic *yes*, then you'll probably be working more inductively. Often this connects to your research question: for the childfree dataset, for instance, you might want to understand people's *experiences of* being childfree. Within the dataset, people's articulated experiences would then form the starting point for your coding and theme development.

[6]The meanings of the terms induction and deduction in a qualitative analytic context are different from the more traditional – quantitative – use of these terms, as representing research that is either theory generating (inductive) or theory testing (deductive; determining a hypothesis, making observations, and confirming or revising the theory). Another – less commonly used – analytic orientation you might encounter is *abduction*, where situated scholars use theory to make sense of curiosities in the data, and/or use data to extend, modify or dispute existing theory (e.g. see Blaikie, 2010; Collins, 2019; Kennedy, 2018).

What about a *deductive* orientation? In reflexive TA, a deductive orientation refers to a more researcher- or theory-driven approach, where the dataset provides the foundation for coding and theme development, but the research questions asked – and thus the codes developed – reflect theoretical or conceptual ideas the researcher seeks to understand through the dataset.[7] In some cases, existing theory and concepts might provide a lens through which a researcher interprets and makes sense of the data. As an example, Icelandic education researchers Kjaran and Jóhannesson (2013) drew on the concept of heterosexism to make sense of varied experiences of lesbian, gay, bisexual and transgender (LGBT) school pupils. Heterosexism captures the normative assumption that everyone is heterosexual, and the societal privileging of heterosexuality as normal and natural (Pharr, 2000). The researchers used this concept to make sense of students' experiences, including feeling different and less valued, and an implicit pressure to talk openly about a heterosexual sex life. Prior theory might also inform your coding, and subsequent theme development and interpretation, in a more precise way – perhaps to enrich the empirically based understanding of a theoretical concept (e.g. Beres & Farvid, 2010). Finally, TA can be more deductive when the researcher notices strong connections to theoretical ideas, early on in the process, and starts to code around such concepts.

ALERT In reflexive TA, a deductive orientation means theory provides an interpretative lens through which to code and make meaning of the data.

No matter whether your analysis is more inductive or deductive, or indeed a blend of both, *theory* is still important – it's what gives reflexive TA, and indeed all forms of qualitative analysis, both its foundation, and its analytic power (Chapter Six provides a deeper discussion of theory and TA).

ALERT Reflexive TA, like all forms of qualitative analysis, gets its analytic power from theory.

SEMANTIC TO LATENT CODING

A second consideration is the level at which you will code meaning. Reflexive TA can capture meaning from semantic (participant-driven, descriptive) to latent (researcher-driven, conceptual) levels. *Semantic* coding involves exploring meaning at the surface of the data. **Semantic codes** capture explicitly-expressed meaning; they often stay close to the language of participants or the overt meanings of data. **Latent codes** focus on a deeper, more *implicit*

ALERT The 'best' coding is the coding that best fits *your* purpose in analysing your data.

ALERT Coding in reflexive TA ranges from semantic to latent. These are again best thought of as two ends of a spectrum, and approaches that can be combined in any one analysis, rather than either/or choices.

[7]Some describe quasi-deductive rather than purely deductive orientation to coding (Jaspal & Cinnirella, 2012).

or conceptual level of meaning, sometimes quite abstracted from the obvious content of the data. Because it can be easier to code meaning at the semantic level, initial coding is often semantic. As you become more experienced, or your analysis develops, you may find it becomes easier to generate latent-level codes. What this does *not* mean is that latent coding is better – that idea is one of several misconceptions about semantic and latent codes we clear up in Table 3.1. Whether latent or semantic meaning is the most *relevant* depends on the aims of the project. Furthermore, in practice, the boundaries between semantic and latent codes are not always distinct. Semantic and latent codes are not a dichotomy, but instead represent ends of a continuum of ways of looking at data.

Table 3.1 Dispelling some misconceptions about semantic and latent coding

Misconception	Response
Coding is *either* semantic *or* latent	Semantic and latent are end points on a continuum, and your coding can sit at one or many points along this continuum.
Latent coding is *more sophisticated* and *better* than semantic	Latent coding can be harder to do than semantic coding, because the meaning might be less immediately obvious to you. This doesn't make it necessarily more sophisticated, or indeed better.
Semantic coding is *more respectful* than latent	This idea is based on two assumptions: (1) that if the analyst engages semantically, they are not shaping or reading into the data, but simply capturing *what is there*; (2) that to engage latently, and look at meaning beyond what is obviously stated, *is* disrespectful. But we always shape the analysis, no matter how we approach coding, and neither approach is more inherently respectful (to participants) or disrespectful. Respectful interpretation is *not* tied to coding level (for more on the politics of representation, see Chapter Seven).
Latent codes capture *unconscious* meaning	We hear this especially among the counselling and psychotherapy students we teach – reflecting the use in psychoanalytic theory of latent to refer to unconscious meaning and experience. In reflexive TA, latent is used in a broader sense to refer to non-obvious, *hidden* or *concealed* (that is, non-explicit) meaning. But *if* your theoretical lens allowed, latent codes could capture 'unconscious' meaning.
Semantic codes *have to* use participant words	Semantic codes are not about summarising the words participants say (or the words written in a text); they are about capturing explicitly-stated ideas, relevant to answering your research question.
Codes are latent if they use words that weren't in the data	Not using the words participants say (or the words written in a text) is *not* what makes a code latent. A code is latent if it captures ideas and concepts that are implicit – that is, sit behind or underneath, the obvious, surface-level meanings in the data.
Semantic codes *should not go beyond* the data content	As codes can offer a *take* on the data, codes – even semantic ones – can go beyond the data; they can have some aspect of interpretation in them, connected to why you think the meaning is relevant to your research question.

GENERAL GUIDELINES FOR CODES AND CODE LABELS IN REFLEXIVE TA

Our organic and open approach applies to the codes themselves – and the associated labels you tag segments of data text with. Codes are *your tools* for developing the analytic

depth you need, to do justice to the dataset in addressing your research question. Codes aren't (ontologically) real, in the sense concrete is real or a wheelchair is real. Codes are **heuristic devices** we use, to foster our engagement, to enrich understanding, and push ourselves into interrogating the dataset and our meaning-making with it. That sounds vague, but really the key point is not to worry about getting codes right or wrong. The aim of understanding different *ways* to approach coding is not about doing things perfectly, so much as understanding what you're doing, and why. The point of coding is to develop codes that help you understand meaning in relation to your dataset in a different way than you did before, and to gain insights that address your research question. Given the flexible and organic process for reflexive TA, you might shift your analytic interests as your analysis develops. That's absolutely fine – as long as your research question shifts too. Similarly, your coding might become more latent – or indeed more semantic. Keep in mind that codes should be specific and precise, as your aim is to demarcate and capture (with the code labels) a rich diversity of meaning within the dataset. This is to allow you the widest scope for theme development in subsequent phases: you might cluster code labels into patterns of meaning that aren't even apparent to you in the coding phase.

PRACTICE POINT Don't worry about whether you're getting semantic or latent coding right – use this distinction to reflect on how you're coding the data, and at what level you're working with meaning. If you're only or mainly working at one level, might the other have anything to offer your analysis?

As noted, the label you use for each code offers a shorthand for the broader idea you're working with. Good code labels are pithy and brief – they offer a quick *in* to what the code is about. Generally they are *your* words (but occasionally a particularly evocative pithy data quote might be used). There is no easy formula for what a code label could or should be, but we offer a few useful general guidelines in Table 3.2, with examples of code labels related to 'choice' to illustrate these.

Qualitative analysis is an interpretative, rather than mechanical, process, so absolute rules are impossible. We call these *general* guidelines because they do not have to apply for *every single code*. Understanding the logic behind the overall process and techniques provides the grounding for making *in situ* decisions around analysis.

PRACTICE POINT Reflexive TA offers much flexibility. Understanding purpose and principles behind the tools and techniques allows you to apply them as needed, during your own analysis.

DOING CODING (PHASE TWO)

With all that information to set you going, you are now ready to start coding! We'll explain the process *and* illustrate it with data extracts and codes from the childfree dataset. We start with the initial coding process, then talk about the different technologies you might use for doing coding, and then discuss the process around *developing* your coding to the point you're ready to move into theme generation.

Table 3.2 Some general guidelines for developing code labels

General guideline	Principle	Example
Don't just copy the data	A code label should reduce the mess of the data, and summarise the meaning you're identifying in the data extract.	A code label like 'choice is important' summarises an idea that is often more loosely or fluidly expressed in data (and from semantic to latent levels).
Identify the particular angle of the meaning	A good code label should not be too broad, and should contain some indicator of the specific meaning in the data.	A code label like 'choice' potentially captures very contradictory meanings related to choice, and is too broad to be effective. Better code labels would indicate what specifically was stated or evoked around choice in a particular segment of data – such as 'choice is important' or 'it's not always a choice'.
Indicate your analytic take somehow	A code label often contains some indication of your interpretative take – what you think is particularly important or interesting about this particular meaning.	Rather than being based in the meaning-frameworks expressed by the commentators, a code label 'having choice is what is *ultimately* important' captured *our* analytic take. Although the commenters rarely expressed it in quite these terms, *having* choice was regularly expressed in a way that framed it as the most important principle of all.

Activity pause

Before coding

Before we show you what coding looks like, read the excerpts in Box 3.2. Hopefully it's apparent that there are lots of assertions and contestations around the meaning of choosing not to have children and being a parent. There are lots of explicitly stated (or semantic) ideas, as well as different assumptions and concepts that underpin what the commenters write – the logic-frameworks behind what is claimed, that allow it to make sense (latent ideas). Think about what and how you might code it.

The process of coding involves systematically working through each data item and your entire dataset. So where do you start? Take your first data item. Start reading,[8] and stop when you think you have spotted something relevant to addressing your research question, even if

[8]We assume here an analytic process centred on textual data, but TA is starting to be applied to visual data – Boxes 8.3 and 8.4 in Chapter Eight offer some guidance.

Box 3.2

A selection of six extracts from childfree dataset

CHCA: Why do people assume that choosing to be childfree automatically means that you won't have children in your life? My partner and I have decided not to have kids for a range of personal, environmental and social reasons. But I am a Godmother, an aunt, an older cousin, and a friend to many children. Contrary to common misconception those who choose not to have kids are rarely lonely, just more self-aware. In many cases those who choose not to have kids have usually thought a lot more about that decision than those who reproduce

DARE: Very, very well said. Clearly you have never had baby brain.

MACL: And conversely, in many cases those who choose to have children have thought a lot more about that decision than those who choose not to have children. We're an ageing society... And I think parents are pretty self-aware as our children's behaviour is reflected right back at us. For better or for worse.

GRKO: lets not forget those who would die to have kids of their own, but for one reason or another cant. . .

DARE: Agreed. The pity is there's no shortage of people who shouldn't have kids. Should almost have to take a pill TO have a kid.

SHHA: I think you have to want to amd be prepared for the challenge ahead. Those who dont have kids might have other goals they want to achieve good for them im not hating and I wouldnt change my life for anything my daughter did it for me amd I am grateful :)

---

it's only *potentially* relevant. Each time you spot *something* interesting or potentially relevant, tag it with a code label. Each time you encounter some text you want to code, consider whether an existing code applies, or you need to develop a new code.

PRACTICE POINT Each time you notice something potentially relevant in your data, code it with an existing code, tweaked if necessary, or create a new code.

What might you *want to code*? We love sport psychologist Lisa Trainor's description of her approach to coding for her Master's research, which captured the idea of 'data of analytic interest' for us:

> I approached coding as a 'consciously curious' researcher, attended to the athletes' stories, open to hearing (and reporting) different experiences than mine, and looking to connect the data and place it in the box of 'currently know' but also make sense of the 'unknown' data outside the box. (Trainor & Bundon, 2020, p. 9)

We coded each of the data items[9] in Box 3.2 in multiple ways, using a mix of semantic and latent codes (see Table 3.3). When you look at these, you'll probably note that sometimes our code labels here contain information in brackets and/or contain a forward slash with different ideas included in the code label – the label *superiority/hierarchy (childfree on top)* illustrates both points. We do this to signal codes that aren't yet fully clarified or where the exact scope of a code is not settled on; we do this when we're still evolving and refining our codes, which we typically do for much of the coding phase. Remember, code labels are *working tools* for us to parse the meaning we notice. We find this strategy useful for not feeling like we have to fix all our ideas or have analytic clarity on everything too early on. We also often retain some of these features in our final coding, as even final coding is still provisional, in the sense that there is no absolute end point to coding; there is always potential for new understandings.

Let's look at some semantic-level codes first. In CHCA's post, the codes *childfree can be for personal reasons, childfree can be for environmental reasons* and *childfree can be for social reasons* all relate to what is explicitly expressed by CHCA. They state they decided not to have children for a mix of reasons. Note, however, that the code label does not simply summarise this with a *mixed reasons for not having children* code, but instead parses each one out into a separate code. Furthermore, the label phrasing 'childfree can be for...' isn't what CHCA stated. This code labelling was deliberately chosen – even though it wasn't how we initially named the code. We evolved this code label throughout the coding process, and eventually settled on this precise language, because it allowed us to code a wider range of places where a reason for being childfree was discussed. For instance, CHCA talked about their *own* decisions; other posters wrote about *other people's* decisions. In the environmentally orientated code, for instance, any environmental explanation around (not) having children could be included, even oblique references to overpopulation (e.g. PAMA: "the world is so over populated already"). This demonstrates the value of open and organic coding and a thorough code revision process, to settle on useful codes that balance precision and inclusivity.

The example of PAMA's reference to "overpopulation" demonstrates the ways codes *can* capture both quite semantic and quite latent expressions of an idea – while CHCA explicitly links not having children to the environment, PAMA only obliquely references the idea that environmental considerations (connected by some to planetary population) might motivate or justify not having children. Some codes may solely capture a semantically or latently expressed meaning, but qualitative analysis works with *messy* data and, as noted earlier in the chapter, many codes are unlikely to be purely semantic or (to a lesser extent) purely latent.

What might a more purely latent code look like? Consider the code label *good/bad parents* related to DARE's second extract in Table 3.3. This offers a good example of the way code labels operate as analytic shorthand, capturing concisely an idea, without spelling out all the analytic thoughts contained by the code. It might appear that this captures semantic content – there is talk of 'good' parenting and 'bad' parenting in the dataset. But the code label actually signals something deeper – related to hierarchy and legitimacy around parenting. When coding the dataset, we noticed that people often made claims about good and bad

[9]We're treating each comment as a data item.

Table 3.3 A selection of childfree comments data with code labels

Data	Codes
CHCA: Why do people assume that choosing to be childfree automatically means that you won't have children in your life? My partner and I have decided not to have kids for a range of personal, environmental and social reasons. But I am a Godmother, an aunt, an older cousin, and a friend to many children. Contrary to common misconception those who choose not to have kids are rarely lonely, just more self-aware. In many cases those who choose not to have kids have usually thought a lot more about that decision than those who reproduce	Offspring are not the only children in people's lives Childfree by choice Childfree can be for personal reasons Childfree can be for environmental reasons Childfree can be for social reasons Compensatory kids Relationships with other kids The childfree more enlightened than parents (hierarchies) Parents (often) don't make deliberate choices/aren't thoughtful Childfree as a thought-out choice (kids as non-thoughtful action) Superiority/hierarchy (childfree on top) Logic/rationality trumps emotion
DARE: Very, very well said. Clearly you have never had baby brain.	Children destroy (women's) rationality Logic/rationality trumps emotion
MACL: And conversely, in many cases those who choose to have children have thought a lot more about that decision than those who choose not to have children. We're an ageing society... And I think parents are pretty self-aware as our children's behaviour is reflected right back at us. For better or for worse.	Parents (often) *do* make deliberate choices/are thoughtful Childfree as not thoughtful Parents as self-aware Superiority/hierarchy Society needs children (ageing society) Having children as investment for future Parents produce society of tomorrow
GRKO: lets not forget those who would die to have kids of their own, but for one reason or another cant...	Childfree isn't always a choice The 'real' victims are those who can't have kids Some people are desperate for kids Childfree are selfish? (because the real victims are those who want but can't have kids) Genetic relatedness and offspring are important
DARE: Agreed. The pity is there's no shortage of people who shouldn't have kids. Should almost have to take a pill TO have a kid.	Good/bad parents Social engineering/forced sterilisation/eugenics Many/some people shouldn't have kids (the ones who often do)
SHHA: I think you have to want to amd be prepared for the challenge ahead. Those who dont have kids might have other goals they want to achieve good for them im not hating and I wouldnt change my life for anything my daughter did it for me amd I am grateful :)	Each to their own but... Good/bad parents – being prepared/deliberate Kids are a challenge Gratitude for having kids/wouldn't change anything Childfree have other goals Kids are life-fulfilling

parents, creating a hierarchy that seemed to denaturalise the idea that we should all be parents, because only some people are 'fit' for the role. In the early analytic phases (familiarisation, coding), we were not quite sure what was going on with this, and whether it would be relevant. So we developed this code to capture our analytic *questioning* around a common pattern that seemed to underpin statements that contained quite *different* obvious semantic content. This operates as a *latent* code, because it is getting at some pattern or idea *beyond* the explicit or obvious meaning.

Another example of a latent code is the code labelled *compensatory kids* associated with CHCA's extract in Table 3.3. Many of the commenters described relationships with

children in their lives who were not their biological children. The code labelled *relationships with other kids* captures this explicit or semantic expression. In contrast, the code *compensatory kids* captures our initial speculation that the commenters might somehow need to demonstrate that they are not emotionally cold-hearted child-haters, one of the negative stereotypes of childfree women, in particular (Hayfield et al., 2019; Rich et al., 2011). However, our analytic ideas developed even further with this code, captured by the word *compensatory* – as we worked through the dataset, it seemed as if not having children in your life suggested you might be in some way deficient as a human being. This analytic idea was, again, something we wanted to *explore*, but the ideas were still bubbling around, not yet very clearly formed; the label again succinctly summarised a bigger idea we were working with.

Both of these codes also help to illustrate the other code continuum discussed earlier in the chapter: from data-driven or *inductive* codes to researcher-driven or *deductive* codes. Although the code *good/bad parents* initially derived from the data, it reflected *our knowledge* of the parenting literature, the ways good and bad parents get constructed within society, and the work of legitimating and de-legitimation this does around parenting (Clarke, 2001; DiLapi, 1989). So the code captured an analytic idea that reflected *our* knowledge and skills, rather than more directly capturing meaning or interpretation as expressed by the commenters.

Similarly, the code *compensatory kids* captured an analytic idea that went beyond the data. While many non-parent commenters described relationships they had with other children, the code captured our analytic take around this as compensation – based in our knowledge that non-parents and non-normative family structures (e.g. single or lesbian parents) are often characterised as somehow deficient (Clarke et al., 2018; Lampman & Dowling-Guyer, 1995). The codes indicate that we want to do more with these data excerpts than just explore the semantic meaning – they hint at something deeper that can potentially offer important insight into how being childfree is made sense of (by those who are, as well as those who are not, childfree). As noted, at this stage, these analytic ideas are just that – *ideas* we're trying to capture through coding, to get a sense of potential prevalence and scope within the dataset, to assess whether they are worth exploring and developing further.

PRACTICE POINT Coding is quite exploratory, especially earlier in the process. Keep all potentially relevant ideas 'in play' at this point, as you don't know what themes you will develop later.

At the other end of the inductive–deductive spectrum, codes related to choice, such as *childfree by choice* (CHCA) or *childfree isn't always a choice* (GRKO) (see Table 3.3) offer examples of more data-driven codes. The dataset was permeated with *explicit* references to choice, in all sorts of different ways. These quite semantic codes reflect a prominent pattern, immediately obvious to us on a first read. Even though there is scope to engage with choice in a more theoretical way (see for instance Figure 4.7 and Box 4.9 in Chapter Four), these codes capture the dominant *explicit* meanings around choice within the dataset.

These two continua – semantic to latent; data- to theory-driven – are aligned but not perfectly: more theoretical codes are often more latent; more inductive codes are more often semantic (see Table 1.2 in Chapter One). Remember, though, that these are continua.

Most importantly, remember that types of codes are not *real things*, but rather tools and devices that we use to develop our analytic insights. Understanding the process and purpose of coding, and recognising *how* you've engaged with your data – for instance, in a primarily deductive or latent way – through coding and theme development, are what's important for later writing up and producing a cohesive, high quality, reflexive TA (we focus specifically on quality and TA in Chapter Nine).

Actually wrangling data and codes: Technologies of coding

But what about actually *doing* coding? You can use all sorts of technologies – in the broadest sense – to code your data. Ranging from totally manual to entirely electronic, the way you can label data segments includes:

- Handwriting code labels *on* the printed data – it's helpful to print with wide margins to facilitate this. You need to indicate what bit of data the code relates to. You can do that with circles, underlining, highlighting, etc. – whatever works for you.
- Writing code labels on sticky notes and attaching those to printed data. Again, you need a way to clarify which bit of the data the code relates to.
- Writing each new code label on a hard-copy file card and clearly noting where to find each associated extract of data (and where each extract starts/ends).
- Typing the code label beside the data in an electronic version of the dataset formatted into a two-column table – as in Table 3.3.
- Using the *comment* box in Microsoft Word to select a section of text and tag it with a code label (Box 3.4 contains a handy hack, if coding in this way).
- Attaching electronic sticky notes to a PDF version of the dataset.
- Using one of the many software programmes specifically designed to assist coding and analysis of qualitative data, originally collectively referred to as **CAQDAS** (computer assisted qualitative data analysis software), and now often just **QDAS** (see Box 3.3).
- Finally, if you've collected data using an online data *collection* programme, such as SurveyMonkey, you may be able to electronically tag data with code labels within the programme itself.

The way you code intersects with all sorts of things, including circumstances and ability. If you're new to coding, and you have options, play around to find what works best for you. Don't automatically assume that (various) electronic ways are better than hard-copy modes, or vice versa. Each presents opportunities and challenges or constraints, intersected by your circumstances. For most of our careers, we used a manual, handwriting approach. This approach *worked* for each of us – no doubt *partly* because of our age and our training, but also because we have found we think and engage with data differently when we read it as hard copy compared to electronically. We also liked to leave the screen behind, and to code in all sorts of different spaces – which

PRACTICE POINT There is no best or ideal way to manage the practicalities of coding. Do whatever works best for you, but be mindful of the opportunities and constraints different coding technologies offer.

can bring subtly different things to the process. You might notice we're writing this in the past tense. That's because this has changed for Victoria recently, as she no longer has full use of her right hand (she is right-handed), necessitating a shift to coding electronically using voice recognition software. This has caused us to reflect more on these processes. We now consider our previous advocacy for manually coding – or at least learning to code manually before using software, if using software at all – as reflecting an ableist worldview that assumes all people have the same opportunities for, and processes of, engagement.

PRACTICE POINT You don't need to stick to one coding technology – you can shift between different tech. Again, we emphasise doing what works best for you.

How you start to do coding does not lock you into an unbreakable contract! It's possible to start with one approach and move to another (e.g. see Box 6.5 in Chapter Six and Box 7.1 in Chapter Seven). For instance, you might start with some hard-copy coding first, then develop your coding further using QDAS. You might start with QDAS and switch to comment functions in Microsoft Word. With a larger dataset, you might start some initial coding based on semantic content using QDAS and develop a fuller and more latent coding, with the full dataset or a subset of the dataset, using a different technology. What's best might change with our circumstances, with each project, or indeed *within* a project. Despite that, we find ourselves increasingly encountering the view that QDAS makes for better or at least more thorough or rigorous coding and analysis (see Box 3.3). To give some *different* perspectives, we also asked two colleagues who have used and teach QDAS in TA to reflect on this – see Boxes 3.4 and 3.5.

Box 3.3

Is using QDAS better than coding in other ways?

QDAS programmes have evolved over more than three decades to provide often-sophisticated tools to assist coding, code development, and even pattern-development. These programmes offer varied resources for (TA) coding and analysis – and some even allow for some (online) data collection (Silver & Bulloch, 2017). Does this mean they are the best way to code? We want to emphasise there is no decontextualised best way! We share some of the concerns raised by others, about the ways programmes have the potential to embed or implicitly validate certain assumptions around the purpose of qualitative research, which can shape your research practice or outputs (e.g. Zhao, Li, Ross, & Dennis, 2016). Some programmes reflect the early dominance of **grounded theory** (GT); reflexive TA and GT do not share processes or (necessarily) assumptions (Braun & Clarke, 2021a). The language in the promotional material can tell us much about how qualitative research is imagined, reflecting implicit research values. The NVivo website, for instance, states: "Unlock insights from your research: What is your data really trying to show you?" This quotation offers a realist, extractive model of research, where the data reveal their (singular) truth, a truth that exists in there, waiting for you to find it – if you have the right tools. NVivo's claims for speed and efficiency in analysis ("Connecting the dots in your data is faster, easier and more efficient with NVivo") position these as ideal in qualitative researching.

But this is at odds with deep and questioning engagement, and allowing plenty of *time* for reflection and insight to develop (Braun & Clarke, 2013, 2016, 2021d).

Advocates of QDAS have argued that while there are important questions we should be engaging with, related to critiques of distancing, standardisation, dehumanisation, quantification, and decontextualisation (e.g. Jackson, Paulus, & Woolf, 2018), these critiques are themselves often decontextualised. Furthermore, changes in what QDAS offers and facilitates have shifted their possibilities and limits. Without a thorough and conceptual sense of what the purpose and process of reflexive TA is, and without an understanding of the foundations, principles and process of the software we use, we do, however, fear there is a risk that analytic development and depth within reflexive TA can be foreclosed. This might be through an over-emphasis on highly stratified and detailed coding, through over-focus on very semantic, and obvious expression of meaning, or developing themes too early (Braun & Clarke, 2021c). Relatedly, we have concerns about the way the user-engagement with QDAS and the tools the programmes offer *can* risk framing qualitative analysis instrumentally, treating both analysis and the technology as a neutral tool or technique you *apply to* data – rather than an interpretative subjective and reflexive process and practice developed through *engagement with* data (Zhao et al., 2016). The shift in the acronym to QDAS – the disappearance of "computer assisted" – risks implicitly promoting a view of the technology as something that *does* the analysis for you, or allows you to do it, something we find problematic. QDAS provides tools, but analysis is a *process* of mind, and it is important to remember this (Evers, 2018). Useful questions for anyone using QDAS include: how do we challenge our ideas once we're getting comfortable with analysis, and how might we move away from the screen/programme and create space for different ways of data engagement (Jackson et al., 2018), and (therefore) different analytic possibilities? This is not just a question of QDAS versus manual modes, but could be about switching between different digital modalities of engagement – such as using voice recorders to make voice memo notes.

Of course, such questions are important for any developing analysis, regardless of QDAS use – as we'll continue to show.

Researcher Reflection – Box 3.4

Thematic analysis and QDAS, by Gareth Terry

My training as a qualitative researcher tended toward the old school, and I still prefer manual approaches when I do (reflexive) TA, and I have the capacity to code in this way. I have dabbled in the use of QDAS, and like many doctoral students, attended an NVivo workshop when it came to analysis time. I hoped that the software would enhance my analytic processes and reduce the time burden – instead I found the opposite. I enjoyed aspects of the software as a data management tool but found that my preference toward flexible coding was strongly curtailed by features of the programme. The node structure nudged me toward clustering much earlier than I would normally want, which had the effect of creating notable superficiality in my analysis. In other words, my use of the software, the way I interacted with it, worked *against* both the style of analysis that I had been trained in and my existing analytical craft skills. This may have been a

(Continued)

consequence of not fully understanding NVivo, or a constraint I imagined rather than something more tangible, but at the time I found it counterproductive.

My current orientation, and one I use with my students, is to focus on the craft skills of qualitative analysis, using QDAS products only insofar as they *support* existing ways of working and capabilities. I have found a happy medium, where I code using the comment function under the review tab in Microsoft Word. This means my preference for coding multiple facets of various data extracts is not constrained. A colleague (Duncan Babbage at the Auckland University of Technology) has helped develop a bespoke macro for Microsoft Word that can then turn these comment codes into a Microsoft Excel spreadsheet, where each code is listed alongside the data.[10] This mechanism, while 'computer aided', allows me levels of engagement with data that I am invested in, reducing the writing (and reading) burden, while also providing opportunities to present lists of codes and data for other members of a research team to easily access. I am definitely not opposed to QDAS in principle, as long as it facilitates good research practices (e.g. immersion in data and flexible and ongoing coding, without premature clustering).

Researcher Reflection – Box 3.5

On using NVivo, by Alexandra Gibson

I can chart my use, and subsequent views, of NVivo in much the same way as other aspects of my research journey – with elements of curiosity, exploration, and always a healthy dose of critical scepticism. Coming from a strong background in critical qualitative research (see Chapter Six), I had developed firm opinions about various analytical approaches. This was paired with undergoing my research training in a context where QDAS was virtually unheard of and an anathema. So during my PhD, I baulked when I heard about peers having to use NVivo in their work as research assistants – virtually a job requirement for those working in multi-disciplinary teams in health research. I felt wary about the ramifications for the role of the researcher-as-analyst, the loss of interpretation, and possibly even a standardising effect of boxing data into pre-defined units. I had also heard of tools, such as counting instances of words; it concerned me that quantification might take precedence over meaning and context.

It was only post-PhD when I was faced with analysing much larger qualitative datasets, involving upwards of 50–200 participants and projects spanning multiple sites, that I started to think that there might be some benefit in having a programme that could help to store and organise what could be an unwieldy amount of data. In that particular context, using NVivo can be very helpful, in that you can organise data files according to participant groups, method of collection, or study site, making the data feel more manageable at the start of analysis. Having all of your coding in one place and being able to more easily re-code extracts are advantages

[10]The macro is *not* currently publicly available. If it does become available, we'll provide a link to it from the companion website.

I have found with NVivo compared to coding by hand or in a Microsoft Excel spreadsheet. More recently, I was in a role where I had to facilitate NVivo training. My co-facilitator and I always spent considerable time emphasising the importance of some initial hand-coding (if you can, or using the comments function in Microsoft Word), reflexivity, interpretation, and theory. Yet, even while having taught students how to use this programme, I remain sceptical of certain functions and promises that NVivo offers and make it clear to others that analytic rigour comes from knowing the foundations of qualitative research, not from the touch of a few buttons in a programme.

Evolving your coding

Because of the organic nature of coding in reflexive TA, your codes will likely evolve as your analytic insight develops. You may decide your first code was a bit too precise or narrow, and want to make it a bit broader, to capture more related data extracts. If you're finding each code is *unique*, and there's little repetition or patterning starting to happen as you work your way through your dataset, your developing coding is likely too **fine-grained**, particular, and ultimately fragmented. You will need to look for ways to broaden your codes. For instance, we noted the *childfree can be for environmental reasons* code label earlier (see Table 3.3), which expanded from a narrower scope into a broader one – encompassing own and others' reasons, and not just focusing on a *particular* environmental reason (e.g. overpopulation).

Conversely, your codes may be *too* broad or general, and you may hone them to identify a range of meanings related to the same general concept. Take codes around 'choice' that we've already noted. Even in the few extracts shown in Table 3.3, there are different codes related to ideas about choice: *childfree by choice*; *parents (often) don't make deliberate choices/ aren't thoughtful*; *childfree as a thought-out choice (kids as non-thoughtful action)* (all from CHCA); *parents (often) do make deliberate choices/are thoughtful; childfree as not thoughtful* (both MACL); and *childfree isn't always a choice* (GRKO). The whole dataset contained yet further ideas and codes related to 'choice' (as we will go onto discuss in later chapters).

Box 3.6

Codes as building blocks for analysis

At the start of this chapter, we identified codes as the building blocks of your themes; we've often used an analogy of building a dwelling – say a stone house – to illustrate the relationship between codes, themes, and a final analysis. Codes are like the stones you would combine with other stones to build bigger things – walls, for instance (where walls are akin to themes). In coding, you're trying to identify all the different forms of stones you might need, from your big

(Continued)

pile, for the structure you're building. For the sake of our analogy, imagine you have a whole lot of stones, with quite different features. Initially, you identify the basic size differentials as important, and you start to separate them into piles by size – small, medium and large. But as you start to work with the stone, you notice all sorts of other variations. You realise other features, like shape, colour and texture will *also* be relevant to the walls you're about to build. And so your sorting process becomes more refined as you prepare for the build. You end up with *many* different piles, based around relevant differentiations in the stones. These many, many piles will set you up to build good strong walls in a systematic and coherent way. Coding is somewhat akin to this – it is about demarcating the variation in the dataset, in order to develop themes robustly based on clusters of pertinent similar meaning.

The issue of coding precision and capturing diversity of meaning also highlights the value of good familiarisation. We started our coding of the dataset knowing there were lots of different expressions around choice that we wanted to capture; we knew a single code around choice would not be enough. Sometimes familiarisation sets you up well for this task. Sometimes, the close work of coding leads to the refined noticing of meaning in the dataset,[11] and the refinement of codes. Other times, later analytic phases reveal a need for even further refinement around coding. The importance of acuity for difference and shared meaning developed through coding is illustrated by the analogy in Box 3.6.

Refining your coding through multiple rounds

You systematically work your way through the dataset, more than once, when coding, to ensure rigour. Sometimes, you might refine codes across multiple data items 'as you go': "as I continued coding the interviews, I moved back and forth between interviews, making notes on previous interviews as well as the interview I was currently coding" (Trainor & Bundon, 2020, p. 9). But don't be haphazard! Be systematic and thorough. Because possible analytically-interesting meanings evolve *through* the coding process, we recommend going through your dataset *at least* twice, to ensure this process is thorough and rigorous – and there may be further value in additional coding 'runs'. It depends on how close and deep you get initially. We recommend that with each different coding run, you go through the dataset in a different order. If you have 20 data items, for instance, your first coding run might start with item 1 and work sequentially through each data item, finishing with item 20. For your second run, maybe start in the middle of the dataset (e.g. data item 10) and work

PRACTICE POINT Do at least two rounds of coding – to capture evolutions in your codes and coding labels – and vary the order in which you code the data items – to give you a fresh perspective on your dataset.

[11] As we noted in Chapter Two, the phrasing 'noticing of meaning in the dataset' does *not* imply the meaning is just there waiting to be found.

'backwards' – initially towards item 1, and then from item 20 down, finishing on item 11. We recommend mixing up the coding order for two reasons: (1) disrupting what can become a familiar 'flow' for the dataset; and (2) as your coding will continue to be refined, changing the order means certain (later) data items don't get a doubling up of extra depth of insight, while earlier ones miss out – risking an unevenly coded dataset.

Coding is never completed, because meaning is never final; you could (in theory) always notice something more or different in the dataset. In reflexive TA, you aim to reach a point at which you decide it's time to adventure forth to the next phase of your journey. At this point, in preparation for the next phase, you collate together all your codes (and relevant coded data). If you want an example of this, flick forward to Table 4.1 in Chapter Four. The data extracts for each of the codes illustrate how a code label tags similar but not identical content, similar but diverse articulations of the code (including some that are semantic and some latent).

PRACTICE POINT There isn't an absolute endpoint for coding – you make a subjective judgement about when to stop.

Can I stop coding yet?

How do you know if you're at the point when it is time to stop coding? Coding can be alluring – it can draw you in, and make you want to stay. It can tempt you to feel you *need* to go on coding, *ad infinitum*. Although it's important to be thorough, remember coding is only an *early stage* in the analytic process. You are not trying to get at everything that you will say about the dataset at this stage, or even develop the analysis. You're still effectively in the preparation stages. You're trying to parse out a whole range of possible meanings and ideas, which you'll then look at combining and evolving, and sometimes rejecting, as you develop your TA.

So when do you know your codes and code labels have done a *good enough* job of capturing and differentiating diverse meaning? Once you've gone through the dataset thoroughly a couple of times, and refined and finalised your code labels and checked coding for consistency and thoroughness, you're probably in a good position to stop – especially as you know you can come back to coding, if you need to.

PRACTICE POINT If you've reached the point when you're just tweaking your codes and coding labels, perhaps even over-tweaking, you've probably reached the point when it's time to stop and move on to theme development.

There's no absolute test for whether your reflexive TA coding is *good enough*, but we have found an exercise we call 'take away the data' useful for our students, to test both their developing coding and whether the code labels do a good job of capturing meaning (Terry, Hayfield, Clarke, & Braun, 2017). Imagine you had compiled all your code labels, but lost your dataset (Meth, 2017) – gaahh! Look at your list of code labels and ask whether they provide you with a summary of the *diversity* of meanings contained in the dataset? Do they also provide some indication of your analytic take on things? This activity should illustrate clearly why a code label 'choice', mentioned above, would be a weak code and

doesn't do what it needs to do. It doesn't parse out the diversity of meaning *in* the dataset, and it doesn't provide any indication of why or how you might have been interested in choice. If your code labels similarly don't deliver, you need to do more coding refinement before moving on. But if they do, you've probably done a good job of coding. Another way to look at this activity is to ask: can I make sense of the meaning richness, diversity and any contradiction in the dataset through the codes *alone*? This is important, because that is exactly how you start building your themes – initially working with the codes, not the full dataset.

CHAPTER SUMMARY

We have done two primary things in this chapter: (1) discussed key principles and concepts of coding for reflexive TA; and (2) discussed and illustrated the *process* of coding. We highlighted the way coding in reflexive TA is a process, with codes and code labels an outcome, a product, of the coding process. Moreover, coding is an organic and evolving process, capturing an interweaving of the knowledge, subjectivity and analytic skill of the researcher, engaging closely and systematically with the dataset. We discussed and illustrated the ways codes and coding can be more or less inductive ↔ deductive and semantic ↔ latent. We explored different technologies for coding and emphasised that no matter what technology you use, your thinking, your subjectivity and your analytic insights are the driving force of coding. Even though this is still effectively 'preparatory' work, by the end of coding you should feel that you're getting yourself ready for the theme development part of your adventure.

WANT TO LEARN MORE ABOUT...?

For a detailed worked discussion of ***coding in reflexive TA***, see: Terry, G. (2021). Doing thematic analysis. In E. Lyons & A. Coyle (Eds.), *Analysing qualitative data in psychology* (3rd ed., pp. 148–161). London: SAGE.

For all things ***QDAS***, see the University of Surrey's Computer Assisted Qualitative Data Analysis (CAQDAS) networking project: www.surrey.ac.uk/computer-assisted-qualitative-data-analysis.

For a ***critical and reflective evaluation*** of such software, see: Zhao, P., Li, P., Ross, K., & Dennis, B. (2016). Methodological tool or methodology? Beyond instrumentality and efficiency with qualitative data analysis software. *Forum Qualitative Sozialforschung/Forum: Qualitative Social Research, 17*(2). www.qualitative-research.net/index.php/fqs/article/view/2597.

ACTIVITIES FOR STUDENT READERS

Coding your own dataset: This activity works with the same *men and healthy eating* dataset you worked with in Chapter Two, downloadable from the companion website. Once you've completed familiarisation, you're ready to start this coding activity. Remember your research question is 'How is healthy eating for men represented online?'

- Systematically work through all four data items, following the coding processes outlined.
- Compile a list of all the code labels you have identified. For each, list the data items associated with each.
- Examine the list to determine that codes in general are evident across more than one data item, and to identify similarities, overlaps, or inconsistencies, that might benefit from refinement.
- Go back and recode the dataset, in reverse data item order, making any adjustments necessary.
- Compile a list of final code labels. For each, list the data items associated with each.

If you're teaching content related to this chapter...
Don't forget to check the companion website for a range of teaching resources.