

1 Introduction to Data Science for the Health Sciences

What is Data Science, and Why Does Health Need It?

Data science is the disciplined practice of extracting reliable knowledge and defensible decisions from data. It sits at the intersection of three older disciplines — statistics, computer science, and domain (subject-matter) expertise — and adds to them a fourth ingredient that is often underestimated: the engineering discipline required to move data from a messy, real-world source into a form that can be analysed, modelled, and acted upon reproducibly. In the health sciences, this fourth ingredient is not a luxury. A clinician who trusts a p-value produced by a script that silently dropped half the missing records, or a policymaker who allocates malaria commodities on the basis of a chart built from an unversioned spreadsheet, is exposed to a specific and preventable kind of error: an error not of statistical theory, but of data handling.

This textbook treats data science as a single connected pipeline rather than a collection of unrelated tools. That pipeline runs from problem formulation, through data collection and cleaning, to exploratory analysis, formal statistical inference, predictive modelling (including machine learning), and finally to communication and decision support. Every chapter in this book

will return to that pipeline and show where its topic sits within it, because a technique learned in isolation — a regression coefficient, a clustering algorithm, a neural network — is of little use to a health system unless the analyst also understands where the data came from, what selection processes shaped it, and who will act on the result.

Health data differs from data in many other domains in ways that matter for how we teach and practise data science. It is frequently incomplete by design (a patient who does not return for a scheduled visit is missing not at random). It is subject to strict ethical and legal constraints (informed consent, the National Health Act 2014, and institutional review board approval in the Nigerian context). It mixes structured records (a DHIS2 aggregate indicator, a laboratory result field) with unstructured ones (a clinician's free-text note, a caregiver's verbal description of symptoms). And, critically, the consequences of getting the analysis wrong are not abstract: a miscalibrated risk score can mean a child is not referred for emergency care, and an over-optimistic forecast of vaccine demand can mean a stock-out in a rural facility. Data science for the health sciences must therefore be learned with an unusual level of rigour and humility.

The Data Science Pipeline: From Question to Decision

A useful mental model for the whole of this book is the pipeline shown below. It is deliberately drawn as a loop rather than a straight line, because in practice health data science is iterative: an unexpected pattern found during exploratory analysis routinely sends the analyst back to the field to ask a better question or collect an additional variable.

- Problem formulation: translating a health system question (“are we seeing more severe malaria cases in the rainy season in Kwara State than the dry season?”) into an answerable data question.

- Data collection and provenance: identifying which data source(s) — HMIS/DHIS2 registers, a household survey such as the Nigeria Demographic and Health Survey (NDHS), a cohort dataset, or a purpose-built study instrument — can answer the question, and documenting exactly how each variable was captured.
- Data cleaning and management: detecting and handling missing values, duplicate records, implausible entries (a reported age of 200 years, a temperature of 15 degrees Celsius in a febrile patient), and reconciling identifiers across linked datasets.
- Exploratory data analysis (EDA): summarising, tabulating, and visualising the cleaned data to generate and refine hypotheses before any formal model is fitted.
- Formal analysis and modelling: applying the statistical or machine learning method appropriate to the question, the data structure, and the assumptions that can reasonably be defended.
- Validation and communication: checking model assumptions and predictive performance, and translating results into a form that a Local Government Area (LGA) disease control officer, a hospital administrator, or a Federal Ministry of Health (FMoH) technical working group can act on.
- Decision and feedback: the point at which the analysis changes a policy, a clinical protocol, or a resource allocation — and where new data begins to accumulate that will eventually be fed back into the pipeline.

Throughout this book, worked examples will be anchored to Nigerian health data contexts that recur across the chapters: routine health facility data reported through DHIS2/NHMIS, malaria diagnostic surveillance data from Global Fund-supported

states, neglected tropical disease (NTD) surveillance (in particular cutaneous and visceral leishmaniasis), adolescent health cohort data, respondent-driven samples of key populations, and dermatoglyphic (fingerprint pattern) data collected in partnership with clinical referral centres. These are not hypothetical illustrations; they are the kind of dataset a Nigerian health data scientist will actually be handed.

Figure 1.1 — The Data Science Pipeline as a Loop

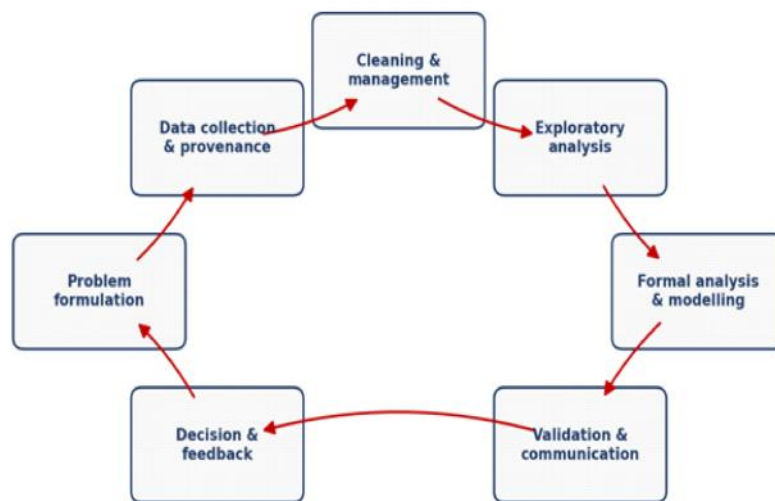


Fig. 1.1 — The data science pipeline, drawn as a loop (Section 1.2).

Who Does Data Science in a Health System, and What Do they Need to Know?

It is useful to distinguish three overlapping roles that this book serves simultaneously, because the balance of statistical theory, programming, and domain framing differs across them.

- The biostatistician or epidemiologist who needs the machine learning and big-data chapters of this book to extend a classical training in inference to modern predictive and unsupervised methods.

- The health informatics or monitoring-and-evaluation (M&E) professional who works daily with DHIS2 and survey data and needs a rigorous statistical foundation to move beyond descriptive dashboards.
- The computer scientist or data engineer entering the health sector who needs a grounding in epidemiological reasoning, study design, and the ethical framework of health research before their technical skills can be trusted with patient-level data.

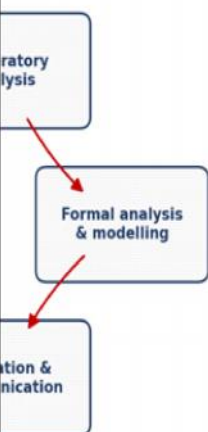
A reader from any of these three backgrounds should be able to follow this book, because each chapter opens with the conceptual foundation before introducing computational implementation in R and Python, the two languages that dominate modern health data science and that are used interchangeably by working analysts in Nigerian research institutions, universities, and public health agencies.

Case Vignette

Why the Pipeline Matters

Consider a real class of problem that recurs in Nigerian malaria control: comparing the test positivity rate (TPR) derived from routine HMIS reporting against the TPR verified from source registers and laboratory logs across the states supported by the Global Fund. On the surface, this looks like a simple two-sample comparison. In practice, a naive analyst who simply merges the two spreadsheets and runs a t-test will be misled, because the two data streams differ systematically in when they aggregate cases (weekly versus monthly), how they treat clinically diagnosed (unconfirmed) cases, and how consistently facilities report in a given month. A rigorous treatment — of the kind developed in Chapters 4, 5, and 8 of this book — reveals a consistent and policy-relevant finding: HMIS-derived TPR tends to run systematically higher than archive-verified TPR across states, a discrepancy with

as a Loop



direct implications for commodity forecasting and for how confidently a State Malaria Elimination Programme can rely on routine data alone. This example illustrates the central argument of this chapter: the statistical technique is often the easiest part of the problem; understanding and correcting for how the data was generated is what makes the analysis trustworthy.

Worked Example

Comparing Two ‘Equivalent’ Numbers that are Not Equivalent

Suppose a facility’s monthly HMIS summary report shows 340 malaria tests performed and 210 confirmed positive, for an HMIS-derived TPR of 61.8%. A retrospective audit of the same month’s laboratory register at the same facility counts 402 tests performed and 218 confirmed positive, for a register-verified TPR of 54.2%. The two numbers differ by 7.6 percentage points, and the difference is not explained by a data-entry typo: the HMIS summary form, in this example, was completed before the last five days of the month were transcribed into the register, so it systematically under-counted the (mostly negative) tests performed in the final week, inflating the reported positivity rate. This is a textbook illustration of the point made in Section 1.4: the two ‘TPR’ figures are not measuring the same underlying quantity with different amounts of noise; they are measuring the numerator and denominator of the same rate at two different, non-random points in the reporting cycle. A naive analyst comparing these two numbers across dozens of facilities and concluding that ‘HMIS over-reports positivity by roughly 7 percentage points on average’ would be correct about the direction and rough size of the bias, but should still trace at least a sample of the discrepancy to its root cause (as done here) before generalising the finding to a policy recommendation.

Quantity	HMIS summary report	Register audit	Difference
Tests performed	340	402	-62 (-15.4%)
Confirmed positive	210	218	-8 (-3.7%)
Test positivity rate	61.8%	54.2%	+7.6 pp

A First Look at Code: Reading and Inspecting a Health Dataset

Before any of the statistical machinery introduced later in this book can be applied, an analyst must be able to load a dataset and perform the most basic checks on it: how many rows and columns does it have, what are the variable names and types, and are there any glaringly implausible values? The two code blocks below perform this first inspection in R and in Python on a simulated facility-level malaria dataset of the kind introduced in this chapter, and they will be extended throughout the book as the running example dataset for many end-of-chapter exercises.

R example

```
# R: first inspection of a facility-level malaria dataset
malaria <- read.csv("hmis_malaria_monthly.csv")

dim(malaria)           # number of rows and columns
str(malaria)           # variable names and types
summary(malaria$tests_performed)
summary(malaria$confirmed_positive)

# Quick plausibility check: TPR should lie between 0 and 1
malaria$tpr <- malaria$confirmed_positive / malaria$tests_performed
range(malaria$tpr, na.rm = TRUE)
sum(malaria$tpr > 1, na.rm = TRUE) # count of facility-months with an impossible TPR
```

PYTHON example

```
# Python: equivalent first inspection
import pandas as pd

malaria = pd.read_csv("hmis_malaria_monthly.csv")

print(malaria.shape)      # (rows, columns)
print(malaria.dtypes)     # variable types
print(malaria[["tests_performed", "confirmed_positive"]].describe())

malaria["tpr"] = malaria["confirmed_positive"] / malaria["tests_performed"]
print(malaria["tpr"].min(), malaria["tpr"].max())
print((malaria["tpr"] > 1).sum()) # facility-months with an impossible TPR
```

A Note on this Book's Structure and How to Read It

Each chapter in this book follows a consistent structure that mirrors the pipeline described in Section 1.2: conceptual foundations first, followed by one or more fully worked numerical examples, followed by code demonstrations in R and/or Python, and closing with a chapter summary, a mini-project, and a set of end-of-chapter exercises with model answers collected in the appendix at the back of the book. Readers who are already comfortable with a chapter's foundational material are encouraged to move directly to its worked examples and mini-project, since these are where the book's Nigerian health data framing is most concretely demonstrated and where the most durable learning tends to occur. Readers newer to a topic are encouraged to work through a chapter's exercises before checking the appendix answer, since the productive struggle of attempting an answer first is a well-established aid to retention that mere reading does not provide.

Extended Case Study**Two Analysts, One Dataset, Two Different Answers**

To make the abstract argument of this chapter concrete, consider a scenario that recurs, in one form or another, in almost every Nigerian health data science team: two analysts are independently given the same raw facility-month malaria dataset and asked to answer the same question — ‘did test positivity fall after the state's indoor residual spraying (IRS) campaign?’ Analyst A loads the spreadsheet, computes the mean TPR for the three months before and the three months after the campaign's stated start date, finds a fall from 38% to 29%, and reports this as clear evidence of programme impact. Analyst B, working from the same raw file, first checks the number of facilities reporting in each month (finding that reporting completeness rose from 61% to 84% of facilities over the same period, likely for reasons unrelated to

the spraying campaign itself), checks whether the campaign's actual roll-out date varied by LGA rather than occurring uniformly on the stated state-wide date, and re-examines the pre-period for a seasonal decline that was already underway before the campaign began (the pre-period spans the tail end of the rainy season, when transmission is naturally beginning to fall regardless of any intervention).

After this additional work, Analyst B's more careful reconstruction shows that roughly half of the apparent 9-percentage-point fall is attributable to normal seasonal decline that would have occurred regardless of the IRS campaign, and a further portion is confounded by the change in which (generally lower-transmission, better-resourced) facilities were reporting in the later period. The genuine, defensible estimate of the campaign's effect — after adjusting for season and reporting-completeness changes using the regression methods of Chapter 8 — is a more modest but still meaningful 3-4 percentage point reduction. Both analysts used entirely correct arithmetic; the difference between their answers is entirely a matter of data-generating-process discipline of exactly the kind this chapter has argued for. When this dataset was actually used to brief a State Ministry of Health technical working group, it was Analyst B's more cautious, adjusted estimate that was ultimately presented, and it changed the programme's messaging from 'IRS delivered a dramatic 24% relative reduction in positivity' to the more defensible and, in the long run, more credible 'IRS appears to have delivered a modest but real reduction in positivity, on top of an already-expected seasonal decline' — a less dramatic headline, but one that would not need to be walked back under later scrutiny.

Building a Personal Habit of Provenance-First Analysis

The single most transferable skill this opening chapter can offer is not any specific technique but a habit: before opening a dataset

in R, Python, or Excel, pausing to write down, in one or two sentences, exactly what population and time period the data claims to describe, how it was collected, and who would be harmed by a wrong conclusion drawn from it. This costs perhaps two minutes and yet reliably prevents a large fraction of the most common and most damaging errors in applied health data science — the kind illustrated throughout this book, from the malaria TPR discrepancy of Section 1.5 to the reverse-causation pitfall of Section 8.10. Analysts who adopt this habit early in their careers report that it becomes automatic within a few months, at which point it stops feeling like an extra step and starts feeling like an inseparable part of what ‘starting an analysis’ actually means.

A simple way to operationalise this habit, useful for individuals and teams alike, is a short standing checklist kept visible during any analysis session: What exactly does each row represent? What time period and geography does this cover? What is the known or suspected direction of any bias in how this was collected? Who will read or act on this analysis, and what would they need to know about its limitations to use it responsibly? Answering these four questions in writing, however briefly, before any code is run, converts an implicit and easily-skipped judgment call into an explicit, reviewable step — precisely the discipline that separates the confident but potentially misleading Analyst A of Section 1.8 from the more careful and ultimately more useful Analyst B.

The Cost of Analysis Paralysis

Balancing Rigour and Timeliness

Everything argued for in this chapter — provenance checks, careful data-generating-process scrutiny, resisting premature conclusions — could, taken to an extreme, produce an analyst who never delivers a timely answer because every dataset can always be interrogated further. A State Ministry of Health facing an active outbreak genuinely cannot wait three weeks for a

perfectly validated, triangulated, publication-grade analysis before deciding whether to deploy emergency supplies; a real-time decision often has to be made on the best currently available evidence, explicitly flagged with its limitations, rather than withheld until an ideal analysis is complete. The skill this book aims to build is not an unconditional demand for perfect data before any conclusion is drawn, but a calibrated judgment: understanding which limitations are severe enough to invalidate a conclusion outright, which are manageable with an explicit caveat, and which can reasonably be set aside given the time pressure and stakes of a specific decision — a judgment that itself improves with experience and that this book’s worked examples are intended to help develop.

Building a Personal Portfolio from this Book’s Mini-Projects

A practical suggestion for any reader working through this book systematically: keep every mini-project’s code, output, and written reflection in a single, well-organised personal repository, rather than treating each as a disposable classroom exercise. By the time a reader reaches Chapter 18, this repository will contain a genuine, demonstrable body of applied work spanning data cleaning, exploratory analysis, regression and survival modelling, machine learning, spatial analysis, mechanistic disease modelling, and a complete Python outbreak investigation — precisely the kind of portfolio that a prospective employer, PhD supervisor, or research collaborator finds far more convincing than a list of course titles or software proficiencies alone. Structuring this repository with a clear top-level README describing each mini-project in one or two sentences, and linking to the relevant code and output, costs perhaps an hour of additional effort across the whole book but meaningfully increases its long-term professional value.

A Worked Example

When Two Reasonable Analysts Disagree, and How to Adjudicate

Two analysts independently examine the same facility-month malaria dataset. Analyst A excludes facility-months with fewer than 10 tests performed, arguing these produce unstable TPR estimates; Analyst B retains them, arguing exclusion introduces selection bias against small, often rural, facilities. Both positions are individually reasonable, and this is a genuinely common situation in applied health data science rather than a sign that one analyst has made an error. The correct way to adjudicate is not to seek a single ‘right’ answer but to make both approaches explicit and run the key analysis under both, checking whether the substantive conclusion changes. If both approaches converge on the same policy-relevant conclusion, the disagreement is methodologically interesting but practically immaterial; if they diverge, the disagreement itself is the finding, and it should be reported transparently rather than resolved by silently picking whichever approach produces the more convenient result.

Refining the Inspection Code

Defensive Programming for Production Use

The inspection code in Section 1.6 is a reasonable first pass, but a version intended for repeated, unattended use (for example, run automatically every month against a new HMIS export) should fail loudly and informatively rather than silently producing a wrong or incomplete result when the input does not match expectations. Two specific improvements are worth making standard practice: validating that expected columns actually exist before referencing them, and returning a structured summary object rather than only printing to the console, so that the result of the check can itself be logged, tested, or fed into a downstream automated pipeline (a pattern extended further in Section 4.13’s validation rules).

PYTHON example

```
# Python: a defensive, reusable dataset inspection function
from dataclasses import dataclass, field

@dataclass
class InspectionReport:
    n_rows: int
    n_cols: int
    missing_columns: list = field(default_factory=list)
    implausible_tpr_rows: int = 0
    ok: bool = True

def inspect_malaria_dataset(df, required_cols=("tests_performed", "confirmed_positive")):
    """Validate a facility-month malaria dataset before any analysis proceeds.

    Returns an InspectionReport rather than printing directly, so the
    result can be logged, unit-tested, or used to halt a pipeline.
    """
    missing = [c for c in required_cols if c not in df.columns]
    if missing:
        return InspectionReport(len(df), len(df.columns), missing_columns=missing, ok=False)

    tpr = df["confirmed_positive"] / df["tests_performed"].replace(0, float("nan"))
    implausible = int(((tpr < 0) | (tpr > 1)).sum())
    return InspectionReport(len(df), len(df.columns), implausible_tpr_rows=implausible,
                           ok=(implausible == 0))

report = inspect_malaria_dataset(malaria_df)
if not report.ok:
    raise ValueError(f"Dataset failed inspection: {report}")
```

R example

```
# R: the equivalent defensive inspection function
inspect_malaria_dataset <- function(df, required_cols = c("tests_performed", "confirmed_positive")) {
  missing_cols <- setdiff(required_cols, names(df))
  if (length(missing_cols) > 0) {
    stop("Missing required columns: ", paste(missing_cols, collapse = ", "))
  }
  tpr <- df$confirmed_positive / ifelse(df$tests_performed == 0, NA, df$tests_performed)
  implausible <- sum(tpr < 0 | tpr > 1, na.rm = TRUE)
  list(n_rows = nrow(df), n_cols = ncol(df), implausible_tpr_rows = implausible,
       ok = implausible == 0)
}

report <- inspect_malaria_dataset(malaria_df)
if (!report$ok) stop("Dataset failed inspection: ", report$implausible_tpr_rows, " implausible rows")
```

Chapter Summary

- Data science for health is a connected pipeline — from problem formulation to decision — not a bag of disconnected techniques.
- Health data carries distinctive constraints: ethical oversight, structural missingness, mixed structured/unstructured formats, and high-stakes consequences for error.
- Nigerian health data sources used throughout this book include DHIS2/NHMIS, NDHS, malaria diagnostic surveillance, NTD (leishmaniasis) surveillance, adolescent health cohorts, respondent-driven samples, and clinical cohort studies.
- Comparing two seemingly equivalent data streams (e.g., HMIS-reported vs. archive-verified test positivity rate) can reveal systematic bias that only careful attention to data provenance uncovers.
- The first step of any analysis is a basic, scripted inspection of dataset dimensions, types, and plausibility — before any statistical model is fitted.

Mini-Project

Mini-Project 1: Building a Data Provenance Log

Choose any dataset you currently work with (or the simulated `hmis_malaria_monthly.csv` used in this chapter). Produce a one-page 'data provenance log' documenting its full lifecycle, and use it to identify one concrete risk of a naive analysis.

Step 1. State, in one sentence, the exact question the dataset is meant to help answer.

Step 2. For each variable, record: how it was captured (paper form, electronic device, laboratory instrument), by whom, and at what frequency.

Step 3. Identify at least one point in the data's journey (collection, transcription, aggregation, transmission) where systematic — not random — error could be introduced.

Step 4. Using the R or Python code from Section 1.6 as a template, load the dataset and report its dimensions, variable types, and any implausible values you find.

Step 5. Write two to three sentences explaining how the risk identified in Step 3 could bias a naive comparison (e.g., a before/after comparison, or a comparison across facilities/states).

Deliverable: A short written provenance log (half a page) plus the annotated inspection code and its output.

End-of-Chapter Exercises

1. List three sources of routinely collected health data available in your own state or institution. For each, identify one plausible way its data-generating process could bias a naive analysis.
2. A colleague reports that ‘malaria cases rose 40% this quarter’ using only HMIS aggregate totals. Write three questions you would ask before accepting this claim.
3. Sketch the data science pipeline (Section 1.2) for a question of your choice drawn from your own research area, naming a specific Nigerian data source at the collection stage.
4. Using the worked example in Section 1.5, calculate the percentage-point difference in TPR if the register audit had instead found 430 tests performed and 215 confirmed positive. Comment on whether the direction of the HMIS bias is the same as in the original example.
5. Explain, in two or three sentences, why ‘a large sample size’ does not by itself protect an analysis from the kind of systematic bias illustrated in Section 1.5.
6. Run the R or Python code in Section 1.6 on a dataset of your choice (or a simulated one) and report the number of rows, columns, and any implausible TPR values found.
7. A donor agency asks you to justify why a full data provenance log (Mini-Project 1) is worth the time it takes to produce, given that ‘the numbers are already in the spreadsheet.’ Write a short, persuasive paragraph in response.
8. Describe, in your own words, how the pipeline in Section 1.2 is better thought of as a loop than a straight line, using an example of when a later stage might send you back to an earlier one.

9. Which of the three professional roles described in Section 1.3 most closely matches your own background, and which of the other two skill sets do you most need to strengthen?
10. Explain briefly why reading a chapter's worked example before attempting its exercises (Section 1.7) might be less effective for long-term learning than attempting the exercises first.
11. In the Section 1.8 case study, explain in your own words why Analyst B's estimate, though less dramatic, was ultimately the more useful one for the Ministry of Health.
12. Identify which specific step in Analyst B's investigation (reporting completeness, campaign timing, or seasonal decline) you think would be easiest to check first with the least additional data, and explain why.
13. If reporting completeness had instead stayed constant across the pre/post period in Section 1.8, would you expect the seasonal-decline confound to fully explain the observed fall on its own? Explain your reasoning.
14. Describe one way a health programme could design its own monitoring so that Analyst A's kind of naive before-after comparison is less likely to be misleading in the first place.
15. Write a two-sentence explanation, suitable for a non-technical programme manager, of why 'the numbers went down after we started the programme' is not by itself proof the programme caused the improvement.
16. Using the four-question checklist in Section 1.9, apply it briefly to a dataset you have worked with recently, writing one short answer to each question.
17. Explain why answering these four questions in writing, rather than just 'thinking about it,' makes a meaningful difference in practice.

18. Which of the four checklist questions in Section 1.9 do you think is most often skipped in real practice, and why?
19. Propose one way a team lead could build this habit into a team's standard workflow, rather than relying on each individual analyst to remember it.
20. Explain briefly why this habit is described as something that becomes 'automatic' over time rather than remaining an effortful extra step indefinitely.
21. Using Section 1.10, describe a situation where waiting for a perfectly validated analysis would itself cause harm, and explain what a reasonable compromise would look like.
22. Explain the difference between a limitation severe enough to invalidate a conclusion outright and one that can be managed with an explicit caveat, using an example.
23. Why does this book argue that calibrated judgment, not a fixed rule, is the right way to balance rigour and timeliness?
24. Propose one way an analyst could flag 'best current evidence, still being validated' clearly to a decision-maker without either overstating or hiding the analysis's limitations.
25. How might this balance differ between a routine annual report and an active outbreak response? Explain your reasoning.
26. Summarise, in a single paragraph, the single most important idea from Chapter 1 that you would want a brand-new research assistant to understand on their first day.
27. Explain how the malaria TPR example in Section 1.5 and the IRS campaign case study in Section 1.8 both illustrate the same underlying lesson, despite being different scenarios.

28. Why does this chapter argue that programming skill alone, without epidemiological or provenance awareness, is not sufficient for responsible health data science?
29. Propose one question you would ask a data provider before accepting a new dataset for analysis, based on this chapter's arguments.
30. If you had to remove one section from this chapter to make room for something else, which would you keep no matter what, and why?
31. In the disagreement between Analyst A and Analyst B (Section 1.12), propose a third possible approach neither analyst considered, and explain its trade-offs.
32. Explain why 'running the analysis both ways and checking if the conclusion changes' is a more useful response to analyst disagreement than trying to determine a single objectively correct choice.
33. Describe a disagreement you have witnessed or could imagine among colleagues analysing the same dataset, and how Section 1.12's adjudication approach would apply.
34. Why might silently picking whichever approach gives the more convenient result be especially tempting, and especially dangerous, under time pressure?
35. If your reported finding depended entirely on one debatable analytic choice (such as an exclusion threshold), what should you disclose in your written report?