Not So Short Introduction to R

A Practical Learning Journey from First Contact to Reproducible Data Systems

Daniel James

June 7, 2026

Table of contents

Not So Short Introduction to R

Dedication

For the learners who have carried data work by memory, patience, and trial and error, and who are ready to turn that effort into a clearer, repeatable way of working.

About This Book

This book is for people who want to learn R without being rushed past the part where confusion usually begins.

You may be a researcher, analyst, graduate student, lecturer, professional, or self-taught learner. You may already work with data in Excel, Google Sheets, SPSS, STATA, forms, dashboards, or exported files from different systems. You may have copied code from a tutorial or asked an AI tool for help, only to find that the code worked once but did not become a method you could fully explain.

You are not starting from nothing.

You already know that data work can be messy. You know that files arrive in different formats. You know that columns change names. You know that values look correct until they refuse to calculate. You know that a result is easier to trust when the path behind it is clear.

This book was built for that learner.

Not the person who wants to become a software engineer overnight.

Not the person looking for a thin list of shortcuts.

The serious beginner who wants to understand R properly, from first contact with the console to reproducible data systems.

Who This Book Is For

This book is written for learners whose work, study, or research depends on data, but who need a patient route into R.

You are likely a fit for this book if you:

- want to learn R from the ground up without being treated like you are unintelligent
- use Excel or other data tools but want a more repeatable workflow
- need to clean, transform, summarize, visualize, or report data
- have tried R before and felt lost after copying a few commands
- want to understand why code works, not only make it run once
- prefer a story-driven explanation that still respects technical accuracy
- want a practical path from first command to reproducible projects

You do not need a computer science background to begin.

You need patience, curiosity, and a willingness to let small ideas become clear before rushing into large ones.

What This Book Is Trying to Solve

Many introductions to R start too quickly.

They begin with syntax, packages, or datasets before the learner understands what kind of system R is. The beginner is asked to run commands before understanding where the commands live, what objects are, why types matter, or how the work will be saved and repeated.

That creates a familiar problem.

The learner may produce an output without building understanding.

A command works, but the meaning is unclear.

An error appears, but the cause is hidden.

A package is loaded, but the learner does not know what R itself is doing.

This book slows down at the right places.

It treats R as a working system, not only a language of isolated commands. It begins with first contact, then builds the ideas that make later data work possible: objects, types, structures, syntax, logic, functions, packages, cleaning, transformation, visualization, communication, reproducibility, collaboration, and scale.

The aim is not to make R look easy by hiding its structure.

The aim is to make R learnable by revealing the structure in the right order.

How to Read This Book

Do not read this book like a reference manual.

Read it like a learning journey.

Each chapter follows Leo, a capable beginner who wants to understand R without pretending that the early confusion is not real. Leo's questions are part of the method. When he pauses, the book slows down. When he meets a problem, the book uses that problem to introduce the next idea.

You will get the most from this book if you use it in three passes.

First, read for orientation. Let the story and explanations give you the mental map.

Second, read with R open. Run the code examples slowly and notice what changes.

Third, read for transfer. Start connecting the ideas to your own data work, research task, class activity, or reporting problem.

You do not need to master everything at once.

You need one clear concept, then another.

What You Will Need

You do not need to be a programmer before you begin.

You do need a few practical things:

- a laptop or desktop computer
- R installed from CRAN
- RStudio installed as the working environment
- a willingness to type commands rather than only read them
- a folder where you can keep scripts, data, and outputs as the book develops
- patience with early errors, because they are part of learning a language

Later in the book, you will work with the teaching dataset `df_r_migration`. It will help you practise real data ideas without jumping between unrelated examples.

For now, the most important tool is attention.

What You Will Learn to Do

By the end of this book, you should be able to:

- understand what R is and why it matters for data work
- use RStudio as a practical working environment
- create, inspect, and reuse objects
- understand data types and data structures
- recognize why values behave differently in R
- use logic and functions with confidence
- install and use packages with purpose
- clean and transform messy data
- build clear summaries and visualizations
- communicate insight through charts and written explanation
- create reproducible reports with Quarto
- organize projects so that files, scripts, data, and outputs make sense
- begin thinking about collaboration, reliability, automation, performance, and scale

The focus throughout is practical understanding.

This is not a book about sounding technical.

It is a book about becoming capable with R in a way that can survive real work.

What Makes This Book Different

This book does not begin by throwing the learner into a full dataset and hoping the concepts become clear later.

It begins with first contact.

The early chapters give Leo the mental ground: what R is, how the R ecosystem is organized, what objects are, why data types matter, and how structures shape behaviour. Only after that

foundation does the book move into data cleaning, transformation, visualization, reporting, and reproducible systems.

The teaching style is deliberate.

Small examples appear before larger ones.

Friction appears before the tool that resolves it.

Concepts are named after Leo has felt why they matter.

The book also uses one main teaching dataset, `df_r_migration`, so that the learning journey stays coherent. Leo does not jump from fruit examples to car examples to unrelated toy data. The same dataset grows with him as the work becomes more realistic.

You are not only learning commands.

You are learning how R thinks about data.

Book Roadmap

The structure follows a gradual movement from first contact to reproducible data systems.

- First, Leo understands why R changes the way data work is recorded and repeated.
- Then, he meets the R ecosystem: R, the console, RStudio, packages, scripts, projects, and the wider working environment.
- Then, he learns the foundation of R data: objects, atoms, containers, types, and structures.
- Then, he learns how R reads and executes instructions through syntax, logic, and functions.
- Then, he moves into real data workflow: coercion, cleaning, and transformation.
- Then, he learns to see and communicate insight through visualization and data storytelling.
- Finally, he builds toward reproducible reports, data connections, collaboration, automation, performance, and an end-to-end project.

This is not a race toward advanced code.

It is a steady movement toward control.

Optional Setup Check

The code blocks throughout the book are meant to be executable. This quick check confirms that your R session is working.

```
version$version.string
#> [1] "R version 4.5.2 (2025-10-31 ucrt)"
```

Introduction: The Day R Became Less Strange

Chapter Opening Story

You know the screen before you understand it.
Not the code. The silence.
The window opens with no spreadsheet grid, no ribbon, no familiar tabs, no rows running down the left side, no columns labelled A, B, C. There is no cell waiting for a value. No formula bar. No button that says sort, filter, clean, summarize, or chart.
Just a console.
A prompt waits on the screen.

```
>
```

It does not explain itself.
It does not ask what file you want to open.
It does not offer a tutorial.
It simply waits.
Leo looks at it longer than he expected.
He is not new to data. He has worked with spreadsheets. He has cleaned tables. He has copied values, checked formulas, dragged columns, fixed formats, removed duplicates, and wondered why the same file behaves differently on two different mornings.

Why This Book Exists

This book exists because many people are told to learn R before they are helped to understand R.

That is where the trouble often begins.

A beginner is handed syntax before meaning.

Functions before context.

Packages before the base system.

Data frames before data itself.

Plots before objects.

Projects before the learner understands where R is running.

The learner copies a line of code. It fails. They copy another line. It fails differently. They search online. They find three possible answers, none of which explains the original problem. Slowly, they begin to believe that R is only for people who were already technical before they began.

That belief is false.

R can be learned by serious beginners.

But it must be taught in the right order.

This book exists to give that order.

It does not treat the beginner as careless or incapable. It assumes the learner is new. That difference matters. A new learner does not need to be patronized. A new learner needs a path, patient explanation, honest friction, and enough repetition for ideas to settle.

Leo is the learner in this book.

He is intelligent.

He is curious.

He is cautious.

He is not stupid, and the book never treats him as if he is. He simply does not yet know how R works. His confusion is not a character flaw. It is part of the learning process.

Through Leo, this book teaches R as a movement from first contact to structured analytical work.

The journey begins with the raw R console because the learner needs to see the engine. Then the book allows the friction of that raw experience to appear. Only after that does RStudio become meaningful.

The same pattern runs through the whole book.

Exposure first.

Friction second.

Solution last.

That principle matters because tools make more sense when they arrive as answers to problems already felt.

RStudio matters after Leo experiences the limits of the console.

Packages matter after Leo sees that base R cannot comfortably do everything.

Data types matter after Leo sees that values which look similar may behave differently.

Coercion matters after Leo sees R silently change values to make them fit.

Pipelines matter after Leo feels the repetition of manual cleaning.

Visualization matters after Leo sees that tables can contain insight without making that insight visible.

Quarto matters after Leo understands that scripts alone do not fully communicate an analysis.

The book does not rush past confusion.

It uses confusion as the doorway into understanding.

What This Book Will Help You Do

By the end of this book, you will be able to move from isolated R commands to a complete, reproducible data workflow.

You will know what R is, why it exists, and why it is more than a calculator. You will know why R should be installed first and why RStudio becomes useful only after the raw console has shown its limits. You will know how to write commands, create objects, inspect values, and save code in a script instead of depending on memory.

You will learn to see R as a system.

R is the engine.

RStudio is the working environment.

Packages extend capability.

Scripts preserve thought.

Projects organize work.

Quarto turns analysis into a document that can be read, checked, rendered, and shared.

The emphasis here is understanding before speed.

This is not a trick-first book. It is not built around memorizing commands. It is built around the slow but powerful movement from confusion to control.

You will learn why R treats data the way it does.

You will learn that data begins as small pieces.

You will learn that those pieces have types.

You will learn that types affect behavior.

You will learn that values are placed into structures.

You will learn that structures determine what R can do easily and what it will resist.

You will learn why this works:

```
10 + 5
```

and why this does not behave the same way:

```
"10" + 5
```

You will learn why a column can look numeric and still refuse to calculate.

You will learn why missing values matter.

You will learn why `NA`, `NaN`, `Inf`, and `NULL` are not random annoyances but signals that must be understood.

Later, you will work with one teaching dataset: `df_r_migration`.

That dataset will carry the practical journey of the book. It includes learner-centred variables such as `learner_id`, `country`, `background`, `study_hours`, `confidence_score`, `project_score`, `first_script_completed`, `enrolment_date`, `first_login_date`, and `week`.

The book does not jump from one unrelated toy example to another.

One dataset grows with you.

At first, you will only hear about it. Then you will inspect it. Then you will clean it. Then you will transform it. Then you will visualize it. Then you will communicate insight from it. Finally, you will use it as part of a reproducible analytical system.

That is the real movement of this book.

From first contact to working structure.

From copied commands to understood code.

From scattered effort to reproducible workflow.

PhD Insight Made Simple

Many beginners think confusion means they are not capable.
In R, confusion often means the learner has reached a hidden rule.
The task is not to pretend the confusion is not there. The task is to expose the rule behind it. Once Leo understands the rule, the problem becomes less personal and more technical.
Technical problems can be inspected, tested, corrected, and practiced.
That is why this book slows down at the moments where many tutorials speed up.

First Contact Application

Before you continue, write down your honest starting point with R.
Answer these five questions:

1. Have you installed R before, or is this your first time?
2. When you see the R console, what feels most unfamiliar?
3. What kind of data work do you currently do most often?
4. What part of that work feels repetitive, fragile, or hard to reproduce?
5. What would you like R to help you do with more confidence by the end of this book?

Do not solve anything yet.
Do not try to sound advanced.
Just name your starting point clearly. Later, this becomes your evidence of progress.

Part 1: Foundations

Before the First Command: Why R Changes the Way Data Work Thinks

Opening Scene: Leo Before the Prompt

The problem is not that Leo cannot work with data.

He can.

He has cleaned spreadsheets late at night. He has copied formulas across hundreds of rows. He has renamed workbooks carefully, then still wondered which version was correct. He has opened files called `final`, `final_updated`, and `use_this_one`, knowing that none of those names guarantees safety.

He has repaired formulas that worked yesterday and failed today because one column moved. He has copied tabs from one workbook to another. He has made manual corrections that were obvious at the time, then difficult to explain a week later. He has watched data travel from file to file until the final result looked neat, while the route behind it became harder to defend.

That is the pressure of spreadsheet-heavy work.

The finished output may look controlled. The chart may be clean. The summary may be accepted. But underneath the result sits a chain of actions that only Leo remembers: the filter he applied, the rows he removed, the lookup he corrected, the formula he extended, the duplicate he fixed by hand.

And memory is a fragile place to store a method.

So when Leo hears people talk about R as powerful, difficult, and important for serious data work, he is not sure what to believe. Some people describe R as if it belongs only to statisticians, programmers, or people who have been coding since childhood.

Leo is not afraid of learning.

He is cautious.

That caution is reasonable. A new tool can make an intelligent person feel slow at first. R will ask Leo to work differently. It will not begin with rows, columns, ribbons, or a familiar grid. It will begin with instructions.

This chapter prepares Leo for that first contact.

He will not master R here. He will not clean a dataset. He will not build a chart. He will not write a full analysis.

Before the first command, he needs to understand why the first command matters.

> **Chapter Focus**
>
> This chapter is not about doing everything in R.
> It is about helping Leo understand what kind of tool R is, and why it changes the way data work is recorded, repeated, and defended.

The Problem Was Never Leo

Many people meet R after years of working with spreadsheets.

That history matters.

It means Leo is not arriving empty. He already understands more about data work than he may realize. He knows that data can be messy. He knows that categories can be inconsistent. He knows that dates can behave strangely. He knows that one small change in a source file can affect the whole report.

But he has learned those lessons inside tools that often hide the path behind the result.

A spreadsheet can show the final number without clearly showing every step that produced it. A workbook can contain formulas, hidden columns, copied sheets, manual corrections, and pasted values. Some of the logic is visible. Some of it is buried. Some of it exists only in the memory of the person who prepared the file.

That is not a personal weakness.

It is a workflow problem.

Leo's issue is not lack of ability. His issue is that his previous tools have allowed too many important steps to remain scattered, manual, or difficult to repeat.

R matters because it changes that relationship.

It gives Leo a way to write the work down.

Not just the answer.

The path.

What R Is

R is a programming language built for data work. It focuses on statistical computing, data analysis, and data visualization.

It was created in the early 1990s by Ross Ihaka and Robert Gentleman at the University of Auckland. The name "R" comes partly from the first letters of their names and partly from its relationship to the earlier S language.

For Leo, the simplest definition is enough:

> R is a language for giving data-related instructions to a computer.

That definition matters because it separates R from many tools Leo may already know.

R is not only an application that opens on the screen. It is not only a calculator. It is not only a statistics program. It is a language Leo can use to express data work clearly and repeatably.

In a spreadsheet, Leo often works by clicking, dragging, filtering, copying, and pasting. In R, he writes instructions. Those instructions can be saved, inspected, corrected, rerun, and shared.

This is the beginning of a different relationship with data.

Leo is no longer only manipulating what he sees on the screen. He is beginning to describe what should happen to the data.

Plain Definition

R is a language for working with data through written instructions.
Those instructions may begin with simple commands, but they can grow into complete workflows for cleaning, analysis, visualization, and reporting.

Why R Exists

R exists to solve one core problem: working with data efficiently and reproducibly.

Data work becomes fragile when it depends only on memory, manual calculations, copied formulas, spreadsheet tabs, and file versions with names like `final`, `final_updated`, or `use_this_one`.

Those habits may get work done for a while. But as the work grows, the risk grows with it. More files. More columns. More people. More updates. More chances for the method to drift away from the result.

R gives Leo a way to describe his work as code.

With R, Leo can eventually:

- store data in structured objects
- clean and transform data
- run statistical analysis
- create visual outputs
- save the instructions behind the result
- repeat the same workflow later

The main idea is simple:

> R does not only help Leo get an answer. It helps him preserve the path to the answer.

That path matters.

In serious data work, the result is only part of the story. Leo also needs to know how the result was produced. He needs to be able to return to the work, check it, revise it, and explain it to someone else.

R supports that kind of work because the steps can be written down.

This does not mean R removes every problem. No tool does. But R gives Leo a stronger way to control work that would otherwise depend too much on manual repetition.

Why R Feels Different from Excel

Leo is used to visible tools. In Excel, he can see rows, columns, cells, formulas, tabs, filters, charts, and menus. The workbook gives him a surface to touch. He can point to a cell, edit it, drag a formula, and see the change immediately.

R feels different because it begins with written instructions.

In a spreadsheet, Leo often sees the data before he writes the logic. In R, he often writes the logic that tells the computer what to do with the data.

That shift can feel strange at first. The screen may look plain. The instructions may feel strict. A missing quotation mark or a wrong capital letter may cause an error.

This does not mean Excel is bad or R is automatically better.

They solve different kinds of problems.

Excel is useful for quick inspection, manual editing, familiar business workflows, and situations where direct visual interaction is helpful.

R becomes useful when work needs to be repeated, checked, documented, scaled, or shared.

> **Excel Is Not the Enemy**
>
> Leo does not need to reject Excel in order to learn R.
> The goal is to understand when spreadsheet work is enough and when a repeatable code-based workflow becomes safer, clearer, and easier to maintain.

The Shift from Manual Work to Reproducible Work

The deeper promise of R is reproducibility.

A spreadsheet workflow may depend on memory:

- which file was opened
- which rows were copied
- which formulas were dragged
- which filters were applied
- which chart was updated manually

Those steps may be familiar, but they can also become invisible. If something changes, Leo may have to reconstruct the process from memory. If someone else asks how the result was produced, he may have to explain a long chain of manual actions that were never fully recorded.

This is where R changes the nature of the work.

R encourages Leo to write those steps down as instructions.

A written instruction can be saved. A saved instruction can be rerun. A rerun instruction can be checked. A checked instruction can be corrected. A corrected instruction can be shared. A shared instruction can be understood by someone else.

The work becomes less dependent on what Leo remembers and more dependent on what the workflow records.

The shift is not from “non-technical” to “technical.”

The shift is from hidden steps to visible steps.

Leo is learning a clearer way to express work he already understands.

> **The Real Shift**
>
> The real shift is not from Excel to R.
> The real shift is from work that depends on memory to work that can be repeated from saved instructions.

The First Lightbulb Moment

At some point, the reason for R becomes clearer.

R is not mainly about replacing Excel.

It is not mainly about looking more technical.

It is not mainly about writing clever code.

R matters because it makes the invisible steps of data work visible, repeatable, and defensible.

That is the lightbulb moment.

When Leo works only through manual actions, the result may survive, but the route can disappear. When Leo writes the work as instructions, the route remains. It can be read. It can be questioned. It can be improved. It can be run again.

This changes the kind of confidence Leo can have in his work.

He is no longer saying only:

> Here is the answer.

He is moving toward being able to say:

> Here is the answer, and here is the path that produced it.

That distinction matters in research. It matters in business. It matters in teaching. It matters anywhere results need to be trusted.

R gives Leo a way to make his data work accountable.

Not perfect.

Accountable.

What Leo Is Expected to Know at This Point

At this point, Leo does not need to know RStudio, packages, data frames, scripts, projects, or visualization.

He only needs to understand a few ideas:

- R is a language for data work
- R exists to make data work more repeatable
- R feels different because it uses written instructions
- R is useful when work needs to be checked, repeated, and improved

The hands-on part begins in Chapter 2.

This matters because beginners often feel pressure to understand everything immediately. Leo does not need that pressure here. He only needs enough orientation to make the first direct encounter with R meaningful.

> **Not Yet**
>
> Leo is not expected to understand RStudio, packages, data frames, projects, or reporting yet.
> Those ideas will come later. For now, he only needs to understand why R is worth meeting.

Chapter Outcome

By the end of this chapter, Leo has:

- understood what R is
- understood why R exists
- seen why R differs from spreadsheet-only work
- understood the value of reproducible data work
- prepared mentally for his first direct interaction with R

Leo has not written code yet.

That is deliberate.

Before learning commands, he needed to understand the problem those commands will help solve. He now sees that R is not simply another software package to click through. It is a language for giving data-related instructions to a computer. It helps turn hidden manual steps into visible, repeatable work.

That understanding is enough for now.

The first command will make more sense because of it.

Transition to Chapter 2

Leo is now ready to meet R directly.

In the next chapter, the idea becomes contact. He will install R, open the console, and face the plain prompt where every R user eventually begins:

```
>
```

That small symbol will ask him for one thing.

An instruction.

> **Ready for First Contact**
>
> Chapter 1 gives Leo the reason for learning R.
> Chapter 2 gives him the first direct experience of using it.

Leo has not typed anything yet.

But the system is waiting.