Coding agents: the basics

How an agent like Claude Code works and how to keep it under control, explained for healthcare professionals.

José Manuel Ramírez Sánchez

jmramirez.engineer · November 2026

THE CASE

A quarterly HbA1c report

A folder of anonymised, fictitious lab results. We ask the agent for the quarter's report, and we always do it in four steps.

1
Explore
Reads the folder and describes the data. Changes nothing.

→

2
Plan
Proposes the steps. You correct and approve them.

→

3
Execute
Writes and runs the scripts on copies.

→

4
Review
You check the result. The responsibility is yours.

In Claude Code 101 this flow is called explore → plan → code → commit.

What an agent is

A chat

Answers once. If something goes wrong, you have to notice.

An agent

Repeats a loop until the goal is met: it thinks, acts with a tool and observes the result.

Turn	Thinks	Acts	Observes
1	What files are there?	ls data/	12 lab-result CSVs
2	I'll merge them with a script	python merge.py	Error: no “Date” column
3	I'll fix the column name	python merge.py	merged_table.csv created

The working folder

```
hba1c-project/
├─ AGENTS.md rules
├─ data/
│  ├─ originals/ untouched
│  └─ working/ copies
├─ scripts/
└─ results/ delivered
```

It is the agent's desk

It only sees what is inside. To read or write outside, it asks you.

The first thing it reads: AGENTS.md

And the structure tells it what each thing is.

Originals apart, copies to work on, results apart.

Permission modes

Shift+Tab switches mode in the middle of the session.

Mode	What it does	When
Ask permission	Asks before editing a file or running a command	By default, and always at the start
Accept edits	Edits without asking; commands still ask for permission	Once you have approved the plan
Plan mode	Only reads and proposes a plan; edits nothing	Before tasks with several steps

With clinical data, start in Ask permission and read each action before approving it.

AGENTS.md and the SHORT rule

A text file with the project's rules. It is loaded **in full** at the start of **every** session.

It is context, not a guaranteed order: the shorter and more specific, the better it is followed.

Claude Code reads CLAUDE.md and, if it does not exist, AGENTS.md. /init creates a first one.

- **Specific:** instructions you can check
- **How-to:** what to do, not the backstory
- **Overhauled:** monthly, delete what is stale
- **Restricted size:** under 200 lines
- **Tidy:** clear headings; specifics in another file

An AGENTS.md for the case

```
# HbA1c project
## Rules
- Reply in English.
- Dates as YYYY-MM-DD.
- Do not modify data/originals/.
- Ask before deleting rows.
## Data
- Anonymised. Columns: DATA.md
## Outputs
- Tables and charts in results/.
## Compact Instructions
- Keep columns and decisions.
```

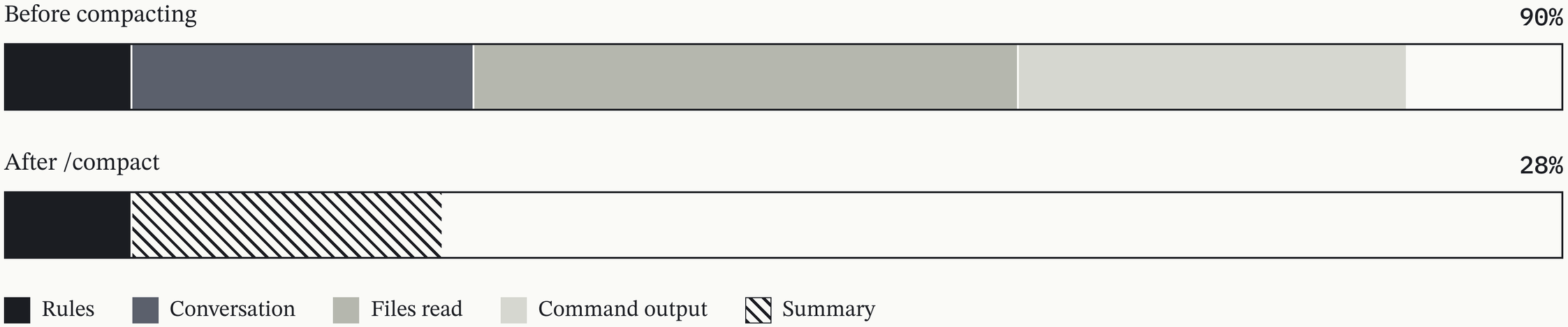
From vague to specific

Vague	Specific
“Sort the data properly”	Dates as YYYY-MM-DD
“Be careful”	Ask before deleting rows
“Be brief”	Replies under 10 lines

Never put patient data or passwords here: the file is sent to the model in every session.

The context window

Everything the model sees at once. It has a fixed size, measured in tokens.



Kept: the full rules, your requests and what matters.

May be lost: long instructions from the start and old results.

Short-term and long-term memory

The model remembers nothing between sessions. What lasts lives in files.

Moment	Window (short term)	AGENTS.md (long term)
Session 1	“Put the dates as YYYY-MM-DD”	No change
Session 1	“Note that in AGENTS.md”	+ Dates as YYYY-MM-DD
/clear	Emptied completely	Unchanged
Session 2	Starts empty	Loaded: it already knows the format

If you want it remembered tomorrow, write it in a file.

Five basic commands

Command	What it does	When
/init	Looks at the folder and creates a first rules file	When starting a project
/clear	New conversation; the rules are still there	When switching tasks
/compact	Summarises and frees window space; you can say what to keep	When the window fills up
/rewind	Goes back to an earlier point: code, conversation or both	If a step goes wrong
/context	Shows what is taking up the window	If it seems to ignore a rule

One session per task: /clear when switching, /compact when full, /rewind if something fails.

Shortcuts that save time and window

Shortcut	What it does
Esc	Stops the agent and keeps what is done; you redirect it without starting over
Shift+Tab	Switches permission mode, including plan mode
@file	Points to one specific file: it reads only that one, not the whole folder
/btw	Side question that does not enter the history or take up window
claude --resume	Resumes an earlier session; --continue opens the last one

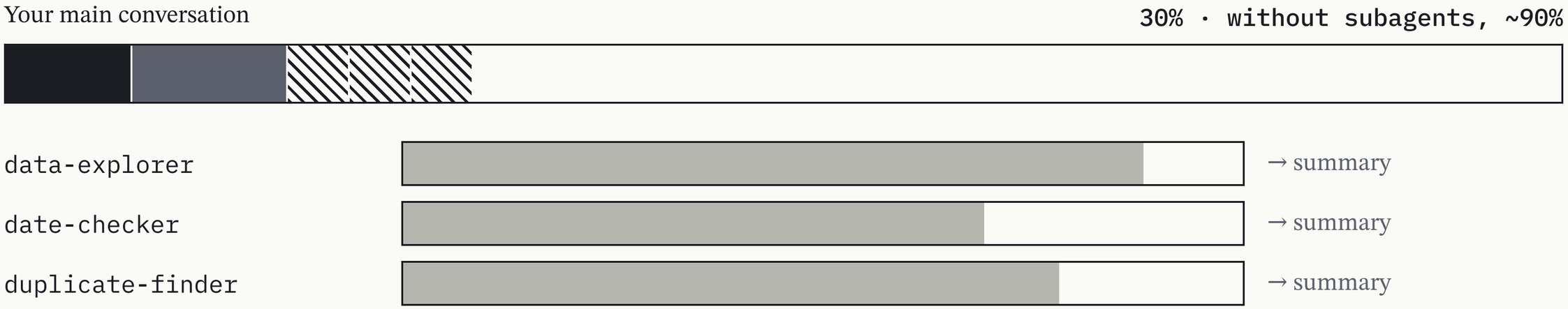
If it still fails after two corrections: /clear and write a better request.

THE CASE, IN ONE SESSION

Explore, plan, execute, review

Step	What you ask	What happens	What you do
Explore	“Read data/working/ and describe the columns. Change nothing.”	Summarises 12 CSVs	Check it understands the data
Plan	Shift+Tab, plan mode: “Propose how to do the report”	Four-step plan	Correct it and approve it
Execute	“Go ahead with the plan”	Deletes rows that were not duplicates	/rewind and “exact duplicates only”
Execute	Moves on to the chart	The window reaches 85%	/compact keep the columns
Review	“Compare the totals with the originals”	They match	You review it; it goes to results/

Subagents: helpers with their own window



How it works: each one reads a lot in its own window and only a summary reaches yours.

When: tasks that read a lot. For a small task it is not worth it.

A subagent is an .md file

```
.claude/agents/date-checker.md
---
name: date-checker
description: Checks date
formats in data/working/
tools: Read, Grep
model: haiku
---
Check that the dates are in
YYYY-MM-DD. Return a summary
of 10 lines at most.
Do not modify any file.
```

Field	What for
description	When to delegate to it
tools	What it can do: here, only read and search
model	A small model is enough for reading
instructions	What to do and how much to return

Minimum permissions: a checker only needs to read, not write. And always ask it for a short summary.

09 · REUSE

Skills: recipes loaded on demand

A folder with a SKILL.md and, if needed, scripts and templates. Only its description is always in view.



Anatomy of a skill

```
.claude/skills/hba1c-report/
├─ SKILL.md
├─ template.xlsx
└─ scripts/chart.py
---
name: hba1c-report
description: Creates the
quarterly HbA1c report. Use
it when that report is asked.
---
1. Read the table in results/.
2. Compute mean and median.
3. Run scripts/chart.py.
```

Where it lives	Who uses it
~/.claude/skills/	You, in all your projects
.claude/skills/	Anyone working on that project

It activates on its own if your request matches the description, or by hand with /hba1c-report.

Short SKILL.md: under 500 lines; the detail, in separate files.

The description decides everything: say what it does and when to use it.

08 + 09 · COMPARISON

AGENTS.md, skill or subagent?

Piece	When it loads	What it is for	Example
AGENTS.md	Always, in full	Project rules	“Dates as YYYY-MM-DD”
Skill	On demand, if it matches	A repeatable procedure	The quarterly HbA1c report
Subagent	When delegated to	Heavy work done apart; returns a summary	Check 40 CSVs without filling your window

What always applies, to AGENTS.md. What you do sometimes, to a skill. What clutters the window, to a subagent.

Local or cloud

Aspect	Local	Cloud
Where it runs	Your computer	An isolated Anthropic machine
Files	Your own, the real ones	A copy of your GitHub repository
Laptop closed	It stops	It keeps working
Review	You see each change as it happens	You review at the end, even from your phone
Best for	Data and interactive tasks	Code and long tasks

Availability depends on version and plan. With HIPAA or zero-retention setups, cloud sessions may not be available.

LOCAL AND CLOUD ALIKE

What the agent reads is sent to the model.

Anonymise first. Patient data: local and anonymised. In the cloud, only code or fictitious data.

Which model to use: three rules

1

The smallest one that solves the task

Start with Haiku or Sonnet. If it fails twice, move up to Opus or Fable with /model.

2

Writing costs more than reading

In Claude, each output token costs 5 times an input token. Ask for short answers.

3

Do not use old models

Each generation gives more for less, and old ones are retired: a script tied to a fixed model stops working.

As of 11 November 2026: Haiku, Sonnet, Opus and Fable, from fastest to most capable. Current prices at anthropic.com/pricing.

The HbA1c case, with the right model

1 · EXPLORE	2 · EXTRACT	3 · DRAFT	4 · EXECUTE	5 · REVIEW
Read the folder and list the columns (subagent)	**Pull the HbA1c out of 200 reports**	**Quarterly summary from the table**	**Full report with scripts and charts**	**Do the conclusions hold up?**
■□□□	■□□□	■■□□	■■■□	■■■■
Claude: **Haiku**	Claude: **Haiku**	Claude: **Sonnet**	Claude: **Opus or Sonnet**	Claude: **Fable or Opus**
`Open: 1–8B`	`Open: 1–8B`	`Open: 8–32B`	`Open: 70B+`	`Open: 70B+, with doubts`

AND THEN The clinical decision is yours: no level of the thermometer reaches it.

Indicative, as of 11 November 2026 · open = run locally, size in parameters · test with your task

PRACTICE

Four challenges with the sample folder

```
agents-practice/
├─ AGENTS.md
├─ DATA.md
├─ data/originals/
├─ data/working/
├─ results/
└─ .claude/
├─ agents/date-checker.md
└─ skills/hba1c-report/
```

1. Open the folder with Claude Code and type /context. Was AGENTS.md loaded?
2. In plan mode, ask for a plan for the report. Correct it before approving.
3. Delegate the date check to the subagent and compare your window before and after.
4. Run /hba1c-report and check the totals against the originals.

SUMMARY

Eight takeaways

1. **An agent acts in a loop**
   You set the goal and review.

2. **A tidy folder and permissions**
   Originals apart; approve each action at first.

3. **A SHORT AGENTS.md**
   Checkable rules, under 200 lines.

4. **The window is finite**
   What must last goes in a file.

5. **Explore, plan, execute, review**
   With /clear, /compact, /rewind and Esc at hand.

6. **Subagents and skills**
   Heavy work is delegated; repeated work is reused.

7. **Always anonymise**
   Local or cloud, what it reads goes to the model.

8. **The smallest model that works**
   And never an old one.