ICT for Civic Data — Crash Course 2026

Foundations

Self-Paced Review — Part 0

civicliteraci.es

The Data Pipeline

> The data pipeline has seven steps in two groups

The data pipeline structures all data work from beginning to end.

Acquisition	Processing
Define — formulate the question, hypothesis, and horizon table	**Verify** — check that the data is what you think it is
Find — locate data sources that could answer the question	**Clean** — fix errors, standardise, consolidate
Get — retrieve the data into your workspace	**Analyse** — answer the questions you defined at the start

Present cuts across both groups. It is about communication: maps, charts, reports, dashboards. The output format depends on the audience and the story.

> The pipeline is iterative

Two loops appear in practice:

Define ↔ **Find** ↔ **Get**: discovering what data exists changes what questions you can answer. Retrieval issues may send you back to Find, and what you find may change the question itself.

Verify ↔ **Clean** ↔ **Analyse**: during cleaning you may discover quality issues that require re-verification. During analysis, something may look wrong and send you back to cleaning.

The pipeline is a living process, not a rigid sequence. Returning to an earlier step is a feature, not a failure.

AI and the Pipeline

> The pipeline as a harness

AI is powerful but walks while looking at its feet. It does not have a world model. It needs fences to stay on track.

The data pipeline IS the harness:

- Each step constrains what the AI does
- Each step produces a verifiable output
- Each boundary is a checkpoint where you inspect the work

Just as a saddle and reins guide a horse, the pipeline guides AI. Without it, AI will jump from Get to Analyse without human verification, and you will not know where it went wrong.

> AI concentrates at Get and Clean

Pipeline step	Human only	AI can help	AI can do it
Define	Research question, editorial framing	Brainstorming	
Find		Source discovery	
Get			Extraction, scraping
Verify	Data quality judgment	Cross-referencing	
Clean			Tidying, formulas
Analyse	Hypothesis testing, interpretation		
Present	Editorial message	Visualisation drafts	

Get and Clean are mechanical, repetitive, and text-based: exactly what AI is good at. Define, Verify, and Analyse require judgment about meaning: exactly what AI is bad at.

> More automation means less visibility

Level	Method	Example
Manual	Copy-paste	Select table, paste in spreadsheet
Semi-manual	Built-in functions	IMPORTHTML in Google Sheets
Tool-assisted	Visual, no code	WebScraper.io Chrome extension
AI-assisted	Chatbot writes code	"Write me a script to scrape this"
AI-driven	Agent does everything	AI navigates, extracts, saves

Each level gives you more power but **less visibility** into what is happening. As automation increases, the burden on **Verify** increases proportionally.

Working With AI

> Structure your prompts with five elements

When a task has multiple steps, the prompt needs structure. Five elements:

1. **Objective** — what are you trying to achieve?
2. **Steps** — what should the AI do, in what order?
3. **Reasoning** — why each step? (so the AI does not skip or improvise)
4. **Output shape** — what should the result look like?
5. **Human verification** — how will you check the result?

A prompt without these elements is an invitation for the AI to fill in the blanks, with hallucinations.

The more complex the task, the more important it is to think through all five. When prompts fail, come back to this framework to diagnose what is missing.

> Tasks stay within pipeline steps

A task must live entirely within **one** pipeline step. If it crosses step boundaries, it is two tasks.

Wrong: "Find the data and make a map of it."

Right: Two tasks:

1. **Find** — produce a list of data sources. You review the list.
2. **Get** — download data from a verified source. You check the file.

Why: if you ask AI to do both at once and the map shows wrong data, you cannot tell whether the AI found the wrong source, fetched bad data, or rendered it incorrectly. Separating steps = separating failure modes.

> Trust the code, not the reasoning

Trust	Don't trust
The AI's **technical capability**: it knows how to write code, query APIs, convert formats.	The AI's **reasoning about data content**: it does not understand what the data means.
Scripts are **deterministic**: the same script on the same data produces the same result every time.	Never ask the AI to directly modify the content of your data. Ask it to write a script that does it.

Exception: AI writing a formula inside a spreadsheet cell IS trustable because you see the result immediately in the cell and can verify it in place.

> AI risk depends on what the code does

Lower risk: functional code

AI writes code for a website menu, a scraper, a map. The goal is **software that works**.

You can test it: does the menu open? Did the scraper get the right rows? Does the map render?

Verification is **direct and immediate**.

Higher risk: data processing code

AI writes code that ranks, compares, or classifies data. The goal is **decisions about the world**.

How do you know the ranking logic is correct, unless you can read the code and understand every choice it made?

Professional Practice

> Core vs peripheral competencies

Core (build skill here)

- > Data cleaning and validation
- > Formula logic and spreadsheet work
- > Judging whether results make sense
- > Understanding what the data represents

These require **your judgment**. AI cannot replace it.

Peripheral (AI can handle)

- > Writing HTML and CSS
- > Memorising git commands
- > Producing visual polish
- > Code syntax

These are mechanical. AI does them well.

Best AI use: ask **"what function would I use for this?"**, not "do it for me."

> Build good taste in data

If you rely on AI to do all the data work, you will never develop **intuition** for what right data looks like and what wrong data looks like.

- Sort the column. Scroll through. Does anything stand out?
- Check the min and max. Do they make sense?
- Count the categories. Are there too many?

This is why we do manual spreadsheet work, even when AI could do it faster. **You are building judgment**, not just producing output.

AI gets lazy with large files: it reads the first 50 rows carefully, assumes the rest are similar, and may make up or skip data afterwards. Verify values near the end of outputs, not just the top.

Quick Reference

> The pipeline at a glance

Step	Key question
Define	What specific question am I trying to answer?
Find	Where is the data that could answer it?
Get	How do I retrieve it into my workspace?
Verify	Is this data what I think it is?
Clean	Is it ready for analysis?
Analyse	What does the data say about my question?
Present	How do I communicate this to my audience?

> The prompt framework at a glance

« What you want to achieve. What the AI should do, in what order. What the result should look like. Why: explain your approach. »

Objective Steps Output shape Reasoning

The fifth element, **human verification**, is how you will check the result. Design this before you start, not after.