Build the dataset

Coding Lesson 2 · Math Camp 2026

Rony Rodriguez-Ramirez

Harvard Kennedy School · API 209

Wednesday, August 19, 2026

Lesson map

From a question to a table
Define the object

R verbs and functions

Reshaping and joining tables

Build and record the table

From a checked table to a defensible handoff

Begin with the policy question

Imagine that a development-policy team is preparing a short briefing on economic resources and child survival. It wants to know where higher income coincides with lower mortality—and which economies do not follow the broad pattern. To represent every economy–year once, the briefing table should contain one row for each economy and year.

How is national income associated with under-five mortality across economies from 2000 to 2022?

Two minutes with someone near you

Suppose Kenya–2022 appears in two rows of the draft analysis table. What could have produced two rows for the same economy and period? Why could that change the briefing's table, plot, or conclusion?

The goal for Lesson 2

Turn one policy question into a checked analysis table.

The table should have a clear population, unit, key, period, variables, and record of the checks we performed.

One investigation across two days

Today: learn the workflow

Translate the policy question into a table, construct it with readable R verbs, and inspect the evidence after each step.

Tomorrow: perform the workflow

Build the same common table from a scaffold, keep a build record, and explain which decisions remain yours.

Today is a guided construction. Lab 2 is the supervised practice.

What I want you to do by the end

1. Explain what one row represents before writing a pipeline.
2. Distinguish a legitimate repeated country–year from a broken key.
3. Read an R sentence: object, assignment, pipe, function, and arguments.
4. Use data verbs for stated structural reasons.
5. Test the unit and key before and after a reshape or join.
6. Rebuild and independently verify a saved analysis table.

Route for the lesson

Block 1 · 00–60		Block 2 · 60–120	
00–10	Working setup	60–82	Mutate + summarise
10–25	Question + table	82–100	Reshape + join
25–50	R grammar + verbs	100–112	Build + verify
50–60	Break	112–120	Evidence + handoff

Three reasoning pauses ask students to explain the sample, diagnose a join, and decide what the completed table can honestly support.

Working-session setup: open the shared project

Open `math-camp-2026.Rproj`, then open `code/lesson-2-walkthrough.R`. Run Section 0:

```
source("code/00-setup.R")
wdi <- read_csv(data_file, show_col_types = FALSE)
nrow(wdi)
```

We continue when

R finds the project files and reports 4,991 rows in the frozen teaching dataset. If it does not, stop here and repair the shared starting point.

The walkthrough's section numbers follow the lesson, so we can move between the deck and RStudio without reconstructing code from the slides.

A question implies a data object

Our question relates two measurements observed for the same economy and year:

- GDP per capita, PPP;
- under-five mortality per 1,000 live births.

To compare them directly, both values must be attached to the same economy–year observation.

One row is not a formatting preference. It is a claim about what counts as one observation in the analysis.

The source can legitimately have two Kenya–2022 rows

Country	Year	Indicator	Value
Kenya	2022	GDP per capita, PPP	5,492
Kenya	2022	Under-five mortality	40.5

Here one row means an economy–year–indicator. The complete key is:

```
iso3c + year + indicator
```

Two Kenya–2022 rows are expected because the indicator distinguishes them.

The analysis table needs one Kenya–2022 row

Country	Year	GDP per capita, PPP	Under-five mortality
Kenya	2022	5,492	40.5

Here one row means an economy–year. The complete key is:

```
iso3c + year
```

The same evidence is now arranged for the comparison in our policy question.

A duplicate is defined relative to a key

Long source table

Two country–year rows can be correct when `indicator` is part of the unit and key.

Wide analysis table

Two country–year rows violate the table plan when `iso3c + year` is supposed to identify each observation once.

The useful question is not simply “Are there duplicate rows?” It is “Does the proposed key uniquely identify the unit we intend to analyze?”

Return to the opening question

What could produce two Kenya–2022 rows, and why could the briefing change?

1. **Long source.** One row per indicator is legitimate when `indicator` belongs to the key; reshape before comparing the measures.

Return to the opening question

What could produce two Kenya–2022 rows, and why could the briefing change?

1. **Long source.** One row per indicator is legitimate when `indicator` belongs to the key; reshape before comparing the measures.
2. **Unnamed dimension.** Sex, unit, or version may distinguish observations that the draft table plan has incorrectly combined.

Return to the opening question

What could produce two Kenya–2022 rows, and why could the briefing change?

1. **Long source.** One row per indicator is legitimate when `indicator` belongs to the key; reshape before comparing the measures.
2. **Unnamed dimension.** Sex, unit, or version may distinguish observations that the draft table plan has incorrectly combined.
3. **Nonunique join key.** A repeated Kenya record on the right can multiply every Kenya year and give those observations excess weight.

Return to the opening question

What could produce two Kenya–2022 rows, and why could the briefing change?

1. **Long source.** One row per indicator is legitimate when `indicator` belongs to the key; reshape before comparing the measures.
2. **Unnamed dimension.** Sex, unit, or version may distinguish observations that the draft table plan has incorrectly combined.
3. **Nonunique join key.** A repeated Kenya record on the right can multiply every Kenya year and give those observations excess weight.
4. **Repeated record.** Importing or appending the same evidence twice can duplicate a point and alter a summary, plot, or model.

Return to the opening question

What could produce two Kenya–2022 rows, and why could the briefing change?

1. **Long source.** One row per indicator is legitimate when `indicator` belongs to the key; reshape before comparing the measures.
2. **Unnamed dimension.** Sex, unit, or version may distinguish observations that the draft table plan has incorrectly combined.
3. **Nonunique join key.** A repeated Kenya record on the right can multiply every Kenya year and give those observations excess weight.
4. **Repeated record.** Importing or appending the same evidence twice can duplicate a point and alter a summary, plot, or model.

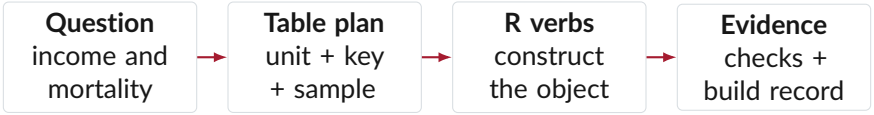
Two rows are not inherently wrong. They are wrong when they violate the unit and key required by the policy question. Diagnose the difference before using `distinct()`.

Write down the table we need

We will call this short list our table plan:

Population	Economy–years with both measures observed and positive income.
Unit	One economy–year.
Key	`iso3c + year`.
Period	2000–2022.
Outcome	Under-five mortality per 1,000 live births.
Comparison	GDP per capita, PPP.

The table plan guides the code



Every consequential line of R should connect to one decision in the table plan or to evidence that the constructed table follows it.

Read the frozen file

```
source("code/00-setup.R")

wdi <- read_csv(
  data_file,
  show_col_types = FALSE
)

glimpse(wdi)
```

The checked-in teaching file is already wide: one row is one economy–year, with indicators stored in separate columns. Keep it unchanged and build forward with code that can be rerun (Wickham et al., 2023).

Lesson map

From a question to a table

R verbs and functions

Predict, run, check, explain

Reshaping and joining tables

Build and record the table

From a checked table to a defensible handoff

Read one R sentence

```
health_selected <- wdi |>
  select(iso3c, year, gdp_per_capita_ppp,
         under5_mortality)
```

Object `wdi` is the input table; `health_selected` is the output.

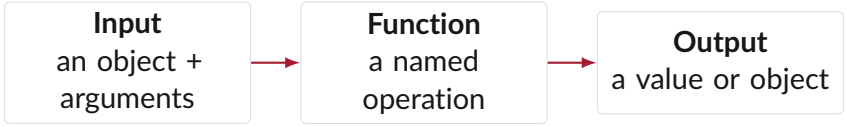
Assignment `<-` gives the result a name.

Pipe `|>` passes the object on the left into the next function.

Function `select()` performs an action.

Arguments The names inside `select()` tell it which columns to keep.

A function connects an input to an output



R functions do not all change data. Some inspect, some construct, some test, and some save. We choose a function because of the evidence we need next.

Inspection functions help us ask before changing

`glimpse()`	What columns and types are present?
`names()`	Which variables are available?
`count()`	How often does a proposed key or category occur?
`n_distinct()`	How many different economies or years appear?
`anyDuplicated()`	Does the proposed key identify every row once?
`stopifnot()`	Which expectation must be true before continuing?

These functions turn “the data look fine” into visible evidence.

Source check 1: how many rows?

Start with one question and one function:

```
nrow(wdi)
```

```
[1] 4991
```

Input The complete `wdi` table.

Function `nrow()` counts its rows.

Evidence The frozen source contains 4,991 economy–year records.

Source check 2: how many economies?

Now select one column with `$` and count its distinct values:

```
n_distinct(wdi$iso3c)
```

```
[1] 217
```

Input `wdi$iso3c`, the economy-code column.

Function `n_distinct()` counts different codes.

Evidence The source contains 217 economies or reporting entities.

Source check 3: which years are covered?

Apply `range()` to the year column:

```
range(wdi$year)
```

```
[1] 2000 2022
```

Input `wdi$year`, the year column.

Function `range()` returns its minimum and maximum.

Evidence The frozen teaching period is 2000–2022.

Source check 4: is the proposed key unique?

Select the two key columns and test them together:

```
anyDuplicated(wdi[c("iso3c", "year")])
```

```
[1] 0
```

Input A two-column table containing the proposed key.

Function `anyDuplicated()` returns zero when no key repeats.

Evidence Each economy–year appears once in the frozen source.

Four verbs change the current table directly

`select()`	Keep the columns required by the question and key.
`filter()`	Keep the observations belonging to the stated population.
`mutate()`	Create or transform a variable while keeping the rows.
`arrange()`	Change the display order without changing the unit.

Before each verb, predict whether rows, columns, values, order, or the unit should change.

Other verbs can change the unit or combine tables

`group_by()` Declare which observations belong together.

`summarise()` Return one row per group and therefore define a new unit.

`pivot_wider()` Move indicator names from rows into columns.

`left_join()` Add columns by matching keys while preserving left rows.

These operations deserve an explicit unit and key check because the output table may represent a different object.

Use one recurring move

This loop is more important than memorizing every function name.

`select()` defines the analytical scope

```
health_selected <- wdi |>
  select(
    iso3c, country, region, income_level_current,
    year, gdp_per_capita_ppp, under5_mortality
  )
```

Predict: Columns leave; rows and the economy–year unit do not.

```
names(health_selected)
stopifnot(all(c("iso3c", "year") %in%
              names(health_selected)))
```

`filter()` changes the population

```
health_filtered <- health_selected |>
  filter(
    between(year, 2000L, 2022L),
    !is.na(gdp_per_capita_ppp),
    gdp_per_capita_ppp > 0,
    !is.na(under5_mortality)
  )
```

Predict: Rows and possibly whole economies leave; columns and the meaning of one remaining row do not change.

Check what the filter excluded

```
tibble(
  stage = c("source", "health sample"),
  rows = c(nrow(wdi), nrow(health_filtered)),
  economies = c(
    n_distinct(wdi$iso3c),
    n_distinct(health_filtered$iso3c)
  )
)
```

The common sample contains **4,296 rows** from **188 economies**. The 695 excluded rows and 29 economies are part of the result, not housekeeping.

Checkpoint 1: name who left

The analysis table requires both income and mortality to be observed. The filtered table is smaller than the frozen source.

Decide together:

1. Does the result still represent “all economies”?
2. What evidence would distinguish missing years from missing indicators?
3. What sample sentence should accompany a later figure or model?

2 minutes discuss $\longrightarrow$ **2 minutes share**

Break · 10 minutes

When we return, we will transform the checked sample with `mutate()`, `arrange()`, `group_by()`, and `summarise()`.

Please keep `math-camp-2026.Rproj` and `code/lesson-2-walkthrough.R` open.

`mutate()` creates a value and an interpretation

```
health_mutated <- health_filtered |>
  mutate(
    log_income = log(gdp_per_capita_ppp)
  )
```

Predict: One column is added; rows and the economy–year unit remain.

The logarithm compresses the upper tail and changes the interpretation of a later coefficient. It is an analytical choice, not merely new syntax.

Check the transformed value

```
health_mutated |>
  summarise(
    min_income = min(gdp_per_capita_ppp),
    missing_log = sum(is.na(log_income)),
    all_finite = all(is.finite(log_income)),
    recomputes = isTRUE(all.equal(
      log_income, log(gdp_per_capita_ppp)
    ))
  )
```

The positive-income condition makes the logarithm defined. The recomputation check ties the derived value to its stated formula.

`arrange()` makes inspection easier

```
health_analysis <- health_mutated |>
  arrange(iso3c, year)
```

Predict: Order changes; rows, columns, values, unit, and key do not.

```
stopifnot(nrow(health_analysis) ==
          nrow(health_mutated))
```

An orderly table is easier to inspect, but sorting does not make a key unique.

Grouping and summarising create a new unit

```
regional_year <- health_analysis |>
  group_by(region, year) |>
  summarise(
    mean_mortality = mean(under5_mortality),
    economies = n_distinct(iso3c),
    .groups = "drop"
  )
```

One output row now represents a region–year, not an economy–year. This summary serves a new descriptive purpose; it does not replace the analysis table required by the original question.

Predict the regional output

Suppose the common table contains seven regions and 23 years.

1. What is the largest possible number of region–year rows?
2. Why might the observed count be smaller?
3. What does the `economies` column reveal that the mean cannot?

The maximum is 161 rows. Uneven country coverage can make a regional mean comparable in syntax but different in composition.

Lesson map

From a question to a table

R verbs and functions

Reshaping and joining tables

Protect the unit

Build and record the table

From a checked table to a defensible handoff

Long and wide can both be correct

Start from the real Kenya–2022 row and temporarily place its two indicators underneath one another:

```
kenya_2022_long <- kenya_2022_wide |>
  pivot_longer(
    cols = c(gdp_per_capita_ppp,
             under5_mortality),
    names_to = "indicator",
    values_to = "value"
  )
```

Long data use `iso3c + year + indicator` as the key. The repeated country–year is informative because the indicator differs.

`pivot_wider()` changes the row meaning

```
kenya_2022_rebuilt <- kenya_2022_long |>
  pivot_wider(
    names_from = indicator,
    values_from = value
  )

stopifnot(!anyDuplicated(
  kenya_2022_rebuilt[c("iso3c", "year")]
))
```

The two indicator rows become two columns on one economy–year row. Reshaping is correct because it moves from the source unit to the unit required by our policy question.

A join can multiply rows quietly

Every value can look plausible while each Kenya–year receives twice the weight. The structure of the right-side key determines the result.

Build and test the right-side table first

```
country_metadata <- wdi |>
  distinct(
    iso3c, country, region, income_level_current
  )

stopifnot(
  !anyDuplicated(country_metadata["iso3c"])
)
```

Here one metadata row should represent one economy. If `iso3c` is not unique, a country–year row on the left can match several rows on the right.

Join only after the relationship is understood

```
health_core <- health_analysis |>
  select(-country, -region, -income_level_current)

health_joined <- health_core |>
  left_join(
    country_metadata,
    by = "iso3c"
  )
```

Predict: Metadata columns return; the number and meaning of the left rows should not change.

Leave a join record

```
stopifnot(nrow(health_joined) == nrow(health_core))
stopifnot(!anyDuplicated(
  health_joined[c("iso3c", "year")]
))

unmatched <- anti_join(
  health_core, country_metadata, by = "iso3c"
)

stopifnot(nrow(unmatched) == 0)
```

Successful syntax is not the completion criterion. The evidence must show that the join preserved the left-side object described by the table plan.

Diagnose before repairing

The row count increased after `left_join()`.
Which action should come first?

1. Delete duplicated rows in the output.
2. Test the right-side key for repeated values.
3. Change to `inner_join()`.
4. Ask an agent to rewrite the complete pipeline.

Begin with number 2. The other actions can hide the mechanism that changed the analytical weight of the observations.

Checkpoint 2: approve or stop the join

A join runs without an error. Output rows rise from 4,296 to 4,319, while the number of distinct economy–year keys remains 4,296.

Decide together:

1. What is the likely relationship between the left and right keys?
2. Which table should we inspect first?
3. What could happen to a later mean or regression if we continue?
4. Why would `distinct()` be an inadequate diagnosis?

2 minutes discuss $\longrightarrow$ **2 minutes share**

Lesson map

From a question to a table

R verbs and functions

Reshaping and joining tables

Build and record the table

One common path

From a checked table to a defensible handoff

One R script follows the entire lesson

Open `code/lesson-2-walkthrough.R` in RStudio.

Move with the deck Each numbered section matches the teaching sequence.

Run one step Select a section and use Command+Enter on macOS or Ctrl+Enter on Windows.

Inspect objects Use the Console and Environment panes after each verb.

Rebuild everything Use Source to run the complete script from the protected input through independent verification.

```
Rscript code/lesson-2-walkthrough.R
```

The script uses the same names, examples, checks, and output paths as the deck.

One complete health pipeline

```
health_analysis <- wdi |>
  select(
    iso3c, country, region, income_level_current, year,
    gdp_per_capita_ppp, under5_mortality
  ) |>
  filter(
    between(year, 2000L, 2022L),
    !is.na(gdp_per_capita_ppp),
    gdp_per_capita_ppp > 0,
    !is.na(under5_mortality)
  ) |>
  mutate(log_income = log(gdp_per_capita_ppp)) |>
  arrange(iso3c, year)

stopifnot(!anyDuplicated(health_analysis[c("iso3c", "year")]))
```

Read the pipeline as decisions

1. **Population:** economy–years without both measures leave the sample.
2. **Scope:** selected columns define the health question.
3. **Transformation:** logged income stays beside the original measure.
4. **Structure:** the country–year key is tested in code.

Each important line should correspond to one decision in the table plan.

Use current metadata within its time meaning

`income_level_current` is the classification attached when this snapshot was assembled.

It is useful for a current grouping. It does not prove that a country belonged to that income group in 2000.

A historical classification needs a separate country–year source, a retrieval date, and documentation.

The build record

Rows and key
Input and output rows; country–year uniqueness.

Missingness
Observed and missing values by variable.

Matches
Unmatched and multiplied join keys.

Years
First year, last year, and coverage.

Ranges
Plausible values in documented units.

Decision note
What changed and why it belongs.

The record is useful when every check points back to the table plan.

Run the saved workflow from a clean process

In the RStudio Terminal, not from a console full of objects:

```
Rscript code/02-build-analysis.R
```

It creates two separate artifacts:

- `outputs/02-health-analysis.csv`: the analysis table;
- `outputs/02-health-build-record.csv`: evidence about how the table meets its stated requirements.

Successful execution is necessary, but it is not yet independent evidence that the saved object is correct.

Verify the saved object independently

Run a second script that reads the frozen source and the saved table afresh:

```
Rscript code/02-verify-analysis.R
```

The verifier checks columns, expected rows, key uniqueness, period, observed values, plausible signs, and whether `log_income` recomputes from the original measure.

The visible completion signal

```
RESULT: PASS. The saved health table meets its stated requirements.
```

The verifier is separate so it does not merely trust objects left behind by the builder.

Four levels of verification

1. **Execution:** Did the process finish and create the named files?
2. **Structure:** Do columns, rows, unit, key, and period match the table plan?
3. **Values:** Do missingness, ranges, and recomputed variables follow their documented definitions?
4. **Interpretation:** Does the policy statement respect the sample limits and the associational design?

An agent can help at every level; it cannot collapse all four into “the code ran.”

Lesson map

From a question to a table

R verbs and functions

Reshaping and joining tables

Build and record the table

From a checked table to a defensible handoff

Review the claim, not just the code

Goal for the live demonstration

We began with a policy question and built one economy–year table. The final task is to decide what that completed object can support now—before anyone makes a plot or estimates a model.

The draft statement we have been asked to review

"Across 217 economies from 2000 to 2022, higher income reduced under-five mortality."

We will ask Codex to audit that sentence against the same files we created, then decide whether its revision deserves to enter a briefing.

Step 1: establish the evidence ourselves

Rebuild the table and independently inspect the saved object:

```
Rscript code/02-build-analysis.R
Rscript code/02-verify-analysis.R
```

Analysis table Economy–year records available for later plots and models.

Build record Included and excluded rows, economies, years, missingness, ranges, and key status.

Verifier A separate process that checks the saved table against its stated construction rules.

Passing checks establish a usable table. They do not establish the substantive relationship in the opening policy question.

Step 2: separate three kinds of statements

The draft sentence mixes claims that require different evidence:

Already documented The analytical sample size, period, unit, key, and variable definitions.

Not yet calculated The direction, size, and uncertainty of the association between income and mortality.

Not supported by this design A causal claim that changing national income reduced mortality.

The live demonstration asks the agent to keep these categories separate. We are not asking it to alter the table or verifier.

Step 3: give Codex one bounded review task

Begin with the decision, artifacts, and draft statement:

```
We need to decide whether a draft sentence is ready for a policy
briefing. Review the sentence against the checked Lesson 2 artifacts.

Draft: "Across 217 economies from 2000 to 2022, higher income
reduced under-five mortality."

Read:
- outputs/02-health-build-record.csv
- outputs/02-health-analysis.csv
- data/documentation/indicator-dictionary.csv
- code/02-build-analysis.R
- code/02-verify-analysis.R
```

The prompt points to the source of truth instead of asking for a general opinion about whether the sentence sounds plausible.

Step 3 continued: require an auditable answer

Complete the same prompt with limits and a response contract:

```
Work in read-only mode. Do not edit files, browse the web, or run a
new statistical model. For each part of the draft, classify it as:
(a) supported by an artifact, (b) not yet calculated, or
(c) not supported by this descriptive design.

Cite the exact file and value used for every supported statement.
Then write a two-sentence handoff note that states what table we
built and what analysis must happen next. Separate facts found in
files from your inferences. If an item cannot be verified, say so.
```

In RStudio, print `bounded_agent_request` to copy the complete prompt into Codex as one message.

Step 4: inspect the answer against the artifacts

Do not accept the prose because it is fluent. Check:

1. Did it use 188 analytical-sample economies rather than 217 source economies?
2. Did it distinguish 4,296 retained rows from 695 excluded rows?
3. Did it avoid claiming that every economy is observed in every year?
4. Did it reserve the association for the next analytical step?
5. Did it reject causal language rather than merely soften the verb?

The class checks the agent's trace against the table plan and build record.

Final synthesis: write the handoff

Classify each part of the draft as **supported**, **not yet calculated**, or **not supported by this design**. Then write two sentences for the analyst who will make the first plot.

Your handoff must state:

1. what one row represents and how large the analytical sample is;
2. which rows were excluded by the table plan;
3. what relationship still needs to be examined;
4. which causal conclusion remains outside the evidence.

2 minutes discuss $\longrightarrow$ **2 minutes share**

A defensible handoff at the end of Lesson 2

Supported by the checked artifacts

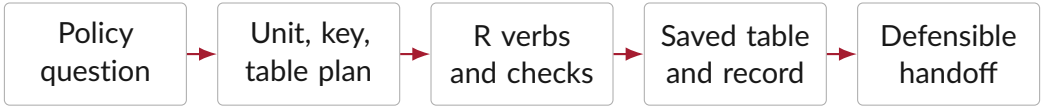
We constructed a table of 4,296 economy–year observations from 188 economies between 2000 and 2022, retaining records with observed under-five mortality and observed positive GDP per capita, PPP. The table excludes 695 source rows and has one row per economy–year.

What happens next

The table is ready for a descriptive comparison of income and mortality. We have not yet estimated that association, and this table alone cannot establish that a change in income causes mortality to change.

This is the handoff from Lesson 2 to Lesson 3: from a checked object to a carefully bounded analysis of patterns.

The thread from question to handoff



Each stage produces evidence needed by the next. No stage is equivalent to “the code ran,” and no agent response substitutes for the final judgment.

Decisions the analyst keeps

- The meaning of one row.
- The population represented by a filter.
- Whether a transformation fits the policy question.
- Whether current metadata can be interpreted historically.
- Whether missing coverage weakens the comparison.
- Whether a proposed sentence stays inside the available evidence.

The agent can organize and challenge evidence. The analyst owns the claim.

Updating public data is an extension

Extending one indicator beyond 2022 can be useful. It also adds:

- an internet and API dependency;
- retrieval dates and version changes;
- possible revisions to earlier years;
- new metadata and comparability questions.

First build and explain the frozen table. Then extend it as a separate, documented investigation.

What we learned in Lesson 2

- Define the table we need before using data verbs.
- Every verb should have a reason and a check.
- A join must preserve the intended key and unit.
- The analysis table should rebuild from a protected source.
- A separate verifier should inspect the saved object.
- A bounded agent review can test whether prose follows from artifacts.
- A checked table is the beginning of analysis, not its conclusion.

Tomorrow: Lab 2

Build one economy–year table.

Thursday, August 20 · 4:00–5:00 p.m.

Start from a scaffold, test the table's stated requirements, and leave a clear handoff for the person who will analyze the first pattern.

Lab 2 route

Minutes	Work	Result
00–05	**Reopen project**	Project, source, and scaffold agree.
05–12	**Table plan**	Population, unit, key, period, and variables.
12–30	**Build**	Select, filter, mutate, arrange, and save.
30–40	**Record**	Rows, key, years, missingness, and ranges.
40–47	**Verify**	Every named independent check passes.
47–57	**Claim review**	One read-only audit tied to the saved artifacts.
57–60	**Handoff**	One supported fact and one remaining limitation.

Your Lab 2 launch sequence

Open `math-camp-2026.Rproj` and `code/lab-2-starter.R`.

1. Confirm that Section 0 loads the 4,991-row frozen source.
2. Fill in the table plan before writing a data verb.
3. Build and save `outputs/02-health-analysis.csv`.
4. Compare your work with `code/02-build-analysis.R` only after making a genuine attempt.
5. Run the independent verifier from the RStudio Terminal:

```
Rscript code/02-verify-analysis.R
```

Then use the verified artifacts for the read-only claim review and write the Lesson 3 handoff.

The common path and the extensions

Required in 60 minutes

- begin from `lab-2-starter.R`;
- select, filter, mutate, and arrange;
- test the economy–year key;
- record coverage and ranges;
- review one claim against the artifacts;
- complete independent verification.

Extension after the common path

- summarize by region;
- diagnose a metadata join;
- adapt another policy track;
- retrieve a newer public indicator.

Everyone leaves with the same checked object. Extensions deepen the work without changing the shared foundation.

The lab is done when

- The script rebuilds the table from the frozen file.
- The economy–year key is unique.
- The build record includes rows, years, missingness, and ranges.
- Every named independent verification check passes.
- You can state one supported fact and one unanswered question without relying on the agent.

The lab has no submission or grade. Keep the script and build record because Lesson 3 will use this table to examine the first pattern.

References I

Wickham, Hadley, Mine Çetinkaya Rundel, and Garrett Grolemund, *R for Data Science*, 2 ed., O'Reilly Media, 2023.