

# **Health Data Science: AI and Knowledge Translation**

M. Ehsan. Karim      AI-KT team

2026-09-15

# Table of contents

|                                                        |               |
|--------------------------------------------------------|---------------|
| <b>The Project</b>                                     | <b>30</b>     |
| What We Aim to Achieve . . . . .                       | 30            |
| Dive into Our Modules . . . . .                        | 31            |
| How Our Content is Presented . . . . .                 | 32            |
| Open Copyright License . . . . .                       | 32            |
| How to Cite . . . . .                                  | 32            |
| <br><b>About the Course</b>                            | <br><b>33</b> |
| <br><b>Course Overview</b>                             | <br><b>34</b> |
| <br><b>Course Overview</b>                             | <br><b>35</b> |
| What you'll learn . . . . .                            | 35            |
| Who it's for . . . . .                                 | 36            |
| How the course works . . . . .                         | 36            |
| What the course is built around . . . . .              | 37            |
| Weekly topics . . . . .                                | 37            |
| About the developer . . . . .                          | 38            |
| <br><b>Week 0: Asynchronous Onboarding</b>             | <br><b>39</b> |
| <br><b>Week 0: Asynchronous Onboarding</b>             | <br><b>40</b> |
| Purpose . . . . .                                      | 40            |
| Required Tasks . . . . .                               | 41            |
| GitHub Account . . . . .                               | 41            |
| GitHub Education Benefits: Apply If Eligible . . . . . | 43            |
| Existing-Account Help . . . . .                        | 45            |
| Course Repository Entry Point . . . . .                | 45            |
| Support Path . . . . .                                 | 48            |
| What To Bring To Week 1 . . . . .                      | 48            |
| Definition Of Done . . . . .                           | 49            |

|                                                                          |           |
|--------------------------------------------------------------------------|-----------|
| <b>Week 1: Health Data, KT &amp; Ethics</b>                              | <b>50</b> |
| <b>Health Data &amp; Ethics</b>                                          | <b>51</b> |
| Overview . . . . .                                                       | 51        |
| Learning Objectives . . . . .                                            | 51        |
| Connection . . . . .                                                     | 52        |
| Case Study Data Spine . . . . .                                          | 52        |
| In-Class Activity . . . . .                                              | 54        |
| Open Data Portals . . . . .                                              | 54        |
| Provenance . . . . .                                                     | 55        |
| Stewardship . . . . .                                                    | 55        |
| Privacy And Security . . . . .                                           | 56        |
| Responsible AI . . . . .                                                 | 56        |
| Indigenous Data Sovereignty . . . . .                                    | 57        |
| Formats And Metadata . . . . .                                           | 58        |
| KT Framing . . . . .                                                     | 58        |
| What Students Leave With . . . . .                                       | 59        |
| <b>Activity: Data Intake Card</b>                                        | <b>60</b> |
| Goal . . . . .                                                           | 60        |
| Approved Portals . . . . .                                               | 60        |
| Student Template . . . . .                                               | 60        |
| Completed Example . . . . .                                              | 61        |
| Completion Checklist . . . . .                                           | 62        |
| Scoring Guide . . . . .                                                  | 63        |
| Submission . . . . .                                                     | 63        |
| Reproducibility Check . . . . .                                          | 64        |
| <b>Week 2: Modern Workflows</b>                                          | <b>65</b> |
| <b>Modern Workflows</b>                                                  | <b>66</b> |
| Overview . . . . .                                                       | 66        |
| Connection . . . . .                                                     | 67        |
| Case Study Data Spine . . . . .                                          | 67        |
| In-Class Activity . . . . .                                              | 69        |
| workflows1.qmd The Analogy: Public Computer and<br>Cloud Drive . . . . . | 69        |
| workflows2.qmd Getting Started: GitHub Account<br>and Fork . . . . .     | 70        |
| workflows3.qmd Launching Your Codespace . . . . .                        | 72        |

|                                                                       |            |
|-----------------------------------------------------------------------|------------|
| workflows4.qmd Tour of VS Code: The “Big Three”<br>Zones . . . . .    | 73         |
| workflows5.qmd Concepts: IDE, Notebook, Script . .                    | 74         |
| workflows6.qmd Concepts: Literate Programming . .                     | 75         |
| workflows7.qmd Project Structure: Files and Folders                   | 76         |
| workflows8.qmd Tutorial: Your First Quarto Docu-<br>ment . . . . .    | 76         |
| workflows9.qmd Tutorial: Stage, Commit, and Sync .                    | 80         |
| workflows10.qmd Stopping Your Codespace (Saving<br>Credits) . . . . . | 82         |
| workflows11.qmd Troubleshooting . . . . .                             | 82         |
| workflows12.qmd Knowledge Check . . . . .                             | 83         |
| workflows13.qmd Assignment 1: Your First Quarto<br>Report . . . . .   | 84         |
| workflows14.qmd Quick Reference . . . . .                             | 85         |
| <b>The Analogy: Public Computer vs Cloud Drive</b>                    | <b>87</b>  |
| Repository (Cloud Drive) . . . . .                                    | 87         |
| Codespace (Public Computer) . . . . .                                 | 88         |
| <b>Tour of the Repository</b>                                         | <b>91</b>  |
| Step-by-step . . . . .                                                | 91         |
| Task . . . . .                                                        | 93         |
| <b>Opening Your First Codespace</b>                                   | <b>94</b>  |
| Steps . . . . .                                                       | 94         |
| The “Boot Up” Phase . . . . .                                         | 96         |
| <b>Reopening a Codespace</b>                                          | <b>98</b>  |
| Steps . . . . .                                                       | 98         |
| <b>VS Code Orientation</b>                                            | <b>100</b> |
| Explorer (left) . . . . .                                             | 100        |
| Editor (center) . . . . .                                             | 101        |
| Terminal (bottom) . . . . .                                           | 102        |
| <b>Files vs Folders</b>                                               | <b>103</b> |
| The Golden Rule of Health Data . . . . .                              | 103        |
| Files vs. Folders (The Mental Model) . . . . .                        | 103        |
| Create a folder . . . . .                                             | 104        |
| Typical structure . . . . .                                           | 106        |



|                                                         |            |
|---------------------------------------------------------|------------|
| <b>The Magic Moment: Render Quarto</b>                  | <b>107</b> |
| Why do we “Render”?                                     | 107        |
| Step 1 — Open the file                                  | 107        |
| Step 2 — Edit name (The YAML Header)                    | 108        |
| Step 3 — Render                                         | 108        |
| If it fails (Troubleshooting)                           | 109        |
| <b>Commit Your Work</b>                                 | <b>111</b> |
| Steps                                                   | 111        |
| Verify (Trust but Verify)                               | 114        |
| <b>Troubleshooting: Rebuild Container</b>               | <b>115</b> |
| When to use this?                                       | 115        |
| Steps                                                   | 115        |
| <b>Stop Your Codespace</b>                              | <b>117</b> |
| Correct shutdown                                        | 117        |
| <b>Final Checklist</b>                                  | <b>120</b> |
| Reflection                                              | 120        |
| <b>Week 3: R with AI</b>                                | <b>121</b> |
| <b>Week 3: R with AI</b>                                | <b>122</b> |
| Overview                                                | 122        |
| Connection                                              | 123        |
| Case Study Data Analysis                                | 123        |
| In-Class Activity                                       | 125        |
| Packages: Opening the Toolbox                           | 125        |
| Core Constructs: Boxes, Toasters, and the Assembly Line | 127        |
| Data Import and First Contact                           | 129        |
| Data Transformation: The Five <code>dplyr</code> Verbs  | 130        |
| Control Flow: Auditing AI Decisions                     | 132        |
| Writing a Simple Function                               | 133        |
| Debugging: Reading the Red Text                         | 134        |
| AI Safety: The Audit Checklist                          | 136        |
| Exporting Clean Data                                    | 137        |
| Literate Analysis in Quarto                             | 137        |
| Knowledge Check                                         | 138        |
| Assignment 2: R With AI                                 | 139        |

|                                                       |                |
|-------------------------------------------------------|----------------|
| Quick Reference . . . . .                             | 140            |
| <b>The Setup (The Cockpit)</b>                        | <b>142</b>     |
| Verify Your AI Extensions . . . . .                   | 142            |
| Pick a Chat Model . . . . .                           | 143            |
| Check the Course Package Environment . . . . .        | 144            |
| Interface Review: Where does R actually live? . . . . | 144            |
| Interface Review: Working with .qmd and .Rmd files .  | 145            |
| <b>Packages (The Toolbox)</b>                         | <b>146</b>     |
| <b>Basics: Boxes &amp; Toasters</b>                   | <b>147</b>     |
| Variables (The Box) . . . . .                         | 147            |
| Functions (The Toaster) . . . . .                     | 147            |
| <b>Control Flow</b>                                   | <b>148</b>     |
| The Pipe %>% (The Assembly Line) . . . . .            | 148            |
| If / Else (The Fork in the Road) . . . . .            | 148            |
| What happens . . . . .                                | 149            |
| Loops (The Treadmill) . . . . .                       | 149            |
| Example . . . . .                                     | 149            |
| What happens . . . . .                                | 150            |
| Audit Check (Important) . . . . .                     | 150            |
| <b>The Loading Dock (Data Import)</b>                 | <b>151</b>     |
| Mini-Examples: Read In, Write Out . . . . .           | 152            |
| <b>The Detective (Debugging)</b>                      | <b>153</b>     |
| <b>AI Safety: The Audit Checklist</b>                 | <b>154</b>     |
| <b>The Paper Trail (Comments)</b>                     | <b>155</b>     |
| <b>Assignment 2: R With AI</b>                        | <b>157</b>     |
| <b>Final Summary Checklist</b>                        | <b>158</b>     |
| <br><b>Week 4: Git &amp; Collaboration</b>            | <br><b>159</b> |
| <b>Git &amp; Collaboration</b>                        | <b>160</b>     |
| Overview . . . . .                                    | 160            |
| Connection . . . . .                                  | 160            |

|                                                                      |                |
|----------------------------------------------------------------------|----------------|
| Case Study Data Analysis . . . . .                                   | 161            |
| In-Class Activity . . . . .                                          | 162            |
| git1.qmd Concepts: The “Save Game” of Reproducible Science . . . . . | 163            |
| git2.qmd Tutorial: Time Travel in Codespaces . . . . .               | 163            |
| git3.qmd Repo Hygiene: .gitignore and Relative Paths . . . . .       | 164            |
| git4.qmd Branching: Safe Experimentation . . . . .                   | 166            |
| git5.qmd Pull Requests: The Peer Review Gateway . . . . .            | 167            |
| git6.qmd AI Auditing for Version Control . . . . .                   | 168            |
| git7.qmd Setting Up Your Group Repository . . . . .                  | 169            |
| git8.qmd Knowledge Check . . . . .                                   | 170            |
| git9.qmd Practice Exercise: The Full Workflow Drill . . . . .        | 171            |
| git10.qmd Quick Reference: Common Git Actions in VS Code . . . . .   | 171            |
| <b>Git Basics</b>                                                    | <b>173</b>     |
| <b>Time Travel</b>                                                   | <b>176</b>     |
| <b>Repo Hygiene</b>                                                  | <b>180</b>     |
| <b>Branching</b>                                                     | <b>183</b>     |
| <b>Pull Requests</b>                                                 | <b>185</b>     |
| <b>AI Auditing</b>                                                   | <b>188</b>     |
| From Prompts to Submission Files . . . . .                           | 188            |
| <b>Setting Up Your Group Repository</b>                              | <b>190</b>     |
| <b>Knowledge Check</b>                                               | <b>191</b>     |
| <b>Practice Exercise: Full Workflow</b>                              | <b>192</b>     |
| <b>Quick Reference</b>                                               | <b>193</b>     |
| <br><b>Week 5: Polyglot Awareness &amp; R Deepening</b>              | <br><b>195</b> |
| <b>Polyglot Awareness &amp; R Deepening</b>                          | <b>196</b>     |
| Overview . . . . .                                                   | 196            |
| Learning Objectives . . . . .                                        | 196            |

|                                                                   |            |
|-------------------------------------------------------------------|------------|
| Connection . . . . .                                              | 197        |
| Case Study Data Analysis . . . . .                                | 197        |
| R Deepening Studio . . . . .                                      | 199        |
| In-Class Activity: R First, Python Second . . . . .               | 199        |
| Page Map . . . . .                                                | 200        |
| Assignment 4 . . . . .                                            | 201        |
| Knowledge Check . . . . .                                         | 201        |
| <b>R Deepening and Parity Check</b>                               | <b>202</b> |
| Learning Goals . . . . .                                          | 202        |
| Dataset . . . . .                                                 | 202        |
| Part A: R Pipeline . . . . .                                      | 203        |
| Part B: R Function . . . . .                                      | 203        |
| More R Mastery Exercises . . . . .                                | 204        |
| AI Audit Prompt . . . . .                                         | 204        |
| Part C: Python Translation . . . . .                              | 205        |
| Part D: Parity Audit . . . . .                                    | 205        |
| Student Output . . . . .                                          | 206        |
| <b>Python vs. R</b>                                               | <b>208</b> |
| The Course Position . . . . .                                     | 208        |
| Strengths At A Glance . . . . .                                   | 208        |
| Notebooks vs. Scripts . . . . .                                   | 209        |
| Check Your Understanding . . . . .                                | 209        |
| <b>Python Environment</b>                                         | <b>210</b> |
| Create A Notebook . . . . .                                       | 210        |
| Use The Variables Pane . . . . .                                  | 210        |
| Load A Tiny DataFrame . . . . .                                   | 211        |
| Common Fixes . . . . .                                            | 211        |
| <b>Packages and Reproducibility</b>                               | <b>213</b> |
| R <code>library()</code> vs. Python <code>import</code> . . . . . | 213        |
| Why <code>requirements.txt</code> Matters . . . . .               | 213        |
| Course Rule . . . . .                                             | 214        |
| R Connection . . . . .                                            | 214        |
| <b>Translation Traps</b>                                          | <b>215</b> |
| Trap 1: Indexing . . . . .                                        | 215        |
| Trap 2: Indentation . . . . .                                     | 216        |
| Trap 3: Assignment vs. Comparison . . . . .                       | 216        |
| Trap 4: Missing Values . . . . .                                  | 216        |

|                                                    |            |
|----------------------------------------------------|------------|
| Audit Habit . . . . .                              | 217        |
| <b>Load and Inspect Data</b>                       | <b>218</b> |
| Load The NHANES Snapshot . . . . .                 | 218        |
| Inspect Before Summarising . . . . .               | 218        |
| Compare To R . . . . .                             | 219        |
| Student Check . . . . .                            | 219        |
| <b>Python Tracebacks</b>                           | <b>220</b> |
| The Bottom Line Is Usually At The Bottom . . . . . | 220        |
| Common Week 5 Errors . . . . .                     | 220        |
| AI Debugging Prompt . . . . .                      | 221        |
| Student Check . . . . .                            | 221        |
| <b>One Step, One Cell</b>                          | <b>222</b> |
| Why This Rule Exists . . . . .                     | 222        |
| Good AI Prompt . . . . .                           | 222        |
| Risky AI Prompt . . . . .                          | 222        |
| Parity Audit Checks . . . . .                      | 223        |
| Notebook Habit . . . . .                           | 223        |
| <b>Mixing R and Python</b>                         | <b>224</b> |
| Why Interoperability Matters . . . . .             | 224        |
| Minimal Demo . . . . .                             | 224        |
| Health Data Example . . . . .                      | 224        |
| When To Use It . . . . .                           | 225        |
| Check Your Understanding . . . . .                 | 225        |
| <b>Reusable Python Scripts</b>                     | <b>226</b> |
| Notebook or Script? . . . . .                      | 226        |
| Create <code>helpers.py</code> . . . . .           | 226        |
| Import It In A Notebook . . . . .                  | 227        |
| Reproducibility Check . . . . .                    | 227        |
| Student Output . . . . .                           | 227        |
| <b>Git Progress</b>                                | <b>228</b> |
| Commit The Audit Trail . . . . .                   | 228        |
| Notebook Output Hygiene . . . . .                  | 228        |
| Example Commit Messages . . . . .                  | 228        |
| Student Check . . . . .                            | 229        |

|                                                  |                |
|--------------------------------------------------|----------------|
| <b>Proposal Readiness</b>                        | <b>230</b>     |
| Why This Belongs In Week 5 . . . . .             | 230            |
| Proposal Readiness Checklist . . . . .           | 230            |
| Feasibility Check . . . . .                      | 230            |
| Student Output . . . . .                         | 231            |
| <b>Knowledge Check</b>                           | <b>232</b>     |
| Short Applied Check . . . . .                    | 232            |
| <b>Bilingual Translation</b>                     | <b>233</b>     |
| Purpose . . . . .                                | 233            |
| Required Task . . . . .                          | 233            |
| Parity Table Template . . . . .                  | 233            |
| AI Prompt . . . . .                              | 234            |
| Definition of Done . . . . .                     | 234            |
| <b>R To Python Cheatsheet</b>                    | <b>235</b>     |
| Translation Rule . . . . .                       | 236            |
| <br><b>Week 6: Data Visualization</b>            | <br><b>237</b> |
| <b>Week 6: Data Visualization</b>                | <b>238</b>     |
| Overview . . . . .                               | 238            |
| Objectives . . . . .                             | 238            |
| Connection . . . . .                             | 239            |
| Case Study Data Analysis . . . . .               | 239            |
| Reading order . . . . .                          | 241            |
| Class plan . . . . .                             | 241            |
| Student output . . . . .                         | 241            |
| Links . . . . .                                  | 241            |
| Definition of done . . . . .                     | 242            |
| What students leave with . . . . .               | 242            |
| <b>Why Visualize, and What For</b>               | <b>243</b>     |
| Goal . . . . .                                   | 243            |
| The four-step pipeline you will follow . . . . . | 243            |
| What a “good” descriptive plot does . . . . .    | 243            |
| Where AI fits . . . . .                          | 244            |
| What you produce this week . . . . .             | 244            |
| Where this goes next . . . . .                   | 245            |

|                                                          |                |
|----------------------------------------------------------|----------------|
| <b>Grammar of a ggplot, and Your First Plot</b>          | <b>246</b>     |
| Where this fits . . . . .                                | 246            |
| The five moving parts . . . . .                          | 246            |
| Setup . . . . .                                          | 247            |
| Load the cached NHANES CSV . . . . .                     | 247            |
| Your first plot . . . . .                                | 248            |
| Three things to check on every plot . . . . .            | 248            |
| Where this goes next . . . . .                           | 249            |
| <br><b>Worked Example: Descriptive BMI Visualization</b> | <br><b>250</b> |
| Where this fits . . . . .                                | 250            |
| Goal . . . . .                                           | 250            |
| Prepare a Descriptive Summary . . . . .                  | 251            |
| Plot . . . . .                                           | 252            |
| Optional Plotly Check . . . . .                          | 253            |
| Caption Template . . . . .                               | 254            |
| Mini Visual Audit Checklist . . . . .                    | 254            |
| Where this goes next . . . . .                           | 255            |
| <br><b>How a Plot Can Mislead</b>                        | <br><b>256</b> |
| Where this fits . . . . .                                | 256            |
| Six failure modes . . . . .                              | 256            |
| Seeing it: a before/after pair . . . . .                 | 258            |
| Why six and not sixty . . . . .                          | 261            |
| A quick audit habit . . . . .                            | 261            |
| Where this goes next . . . . .                           | 262            |
| <br><b>AI Prompts You Can Defend</b>                     | <br><b>263</b> |
| Where this fits . . . . .                                | 263            |
| Why this matters . . . . .                               | 263            |
| Three prompt templates that work . . . . .               | 263            |
| Auditing what the model gives you . . . . .              | 264            |
| The paper-trail rule (still in force) . . . . .          | 265            |
| A note on what NOT to ask the AI . . . . .               | 265            |
| Where this goes next . . . . .                           | 265            |
| <br><b>In-Class Studio: AI Visual Audit</b>              | <br><b>266</b> |
| Where this fits . . . . .                                | 266            |
| Before you start . . . . .                               | 266            |
| Goal . . . . .                                           | 266            |
| Inputs . . . . .                                         | 267            |

|                                                         |            |
|---------------------------------------------------------|------------|
| Activity Steps . . . . .                                | 267        |
| Audit Categories . . . . .                              | 267        |
| Required Outputs (studio draft) . . . . .               | 268        |
| Completion Check . . . . .                              | 269        |
| Where this goes next . . . . .                          | 269        |
| <b>Assignment 5 Walkthrough and Reference</b>           | <b>270</b> |
| Where this fits . . . . .                               | 270        |
| Deadline and link to M1 . . . . .                       | 270        |
| The seven A5 deliverables, explained . . . . .          | 271        |
| Reproducibility checklist for A5 . . . . .              | 274        |
| Accessibility (good practice, not a graded deliverable) | 274        |
| ggplot2 quick reference . . . . .                       | 274        |
| Visual audit checklist . . . . .                        | 275        |
| Five quick knowledge checks . . . . .                   | 276        |
| Where this goes next . . . . .                          | 276        |
| <b>Week 7: EDA, Table 1 &amp; AI Auditing</b>           | <b>277</b> |
| <b>Week 7: EDA, Table 1 &amp; AI Auditing</b>           | <b>278</b> |
| Overview . . . . .                                      | 278        |
| Objectives . . . . .                                    | 279        |
| Connection . . . . .                                    | 279        |
| Case Study Data Analysis . . . . .                      | 279        |
| Reading order . . . . .                                 | 281        |
| Class plan . . . . .                                    | 281        |
| Student output . . . . .                                | 282        |
| Links . . . . .                                         | 282        |
| Definition of done . . . . .                            | 282        |
| What students leave with . . . . .                      | 282        |
| <b>EDA as Evidence Stewardship</b>                      | <b>283</b> |
| Goal . . . . .                                          | 283        |
| The four artifacts this week produces . . . . .         | 283        |
| Why this matters for AI work . . . . .                  | 284        |
| What “descriptive, unweighted, non-causal” means . .    | 284        |
| Where AI fits . . . . .                                 | 285        |
| What you produce this week . . . . .                    | 285        |
| Where this goes next . . . . .                          | 285        |



|                                                                         |            |
|-------------------------------------------------------------------------|------------|
| <b>Cohort and Missingness</b>                                           | <b>286</b> |
| Where this fits . . . . .                                               | 286        |
| Why cohort definition comes first . . . . .                             | 286        |
| Documenting the cohort in code . . . . .                                | 287        |
| Why missingness comes before the summary . . . . .                      | 287        |
| Counting missingness . . . . .                                          | 288        |
| What “complete-case” means and why it can bias the<br>summary . . . . . | 288        |
| A short audit habit . . . . .                                           | 289        |
| Where this goes next . . . . .                                          | 289        |
| <b>Worked Example: EDA and Table 1</b>                                  | <b>290</b> |
| Where this fits . . . . .                                               | 290        |
| Goal . . . . .                                                          | 290        |
| Define the Analysis Cohort . . . . .                                    | 291        |
| Missingness Check . . . . .                                             | 291        |
| Table 1-Style Summary . . . . .                                         | 292        |
| Methods Note Template . . . . .                                         | 293        |
| Stewardship Note Template . . . . .                                     | 293        |
| Survey Design Note . . . . .                                            | 293        |
| Reproducibility Notes . . . . .                                         | 293        |
| Where this goes next . . . . .                                          | 294        |
| <b>What Each Statistic Claims (and Does Not)</b>                        | <b>295</b> |
| Where this fits . . . . .                                               | 295        |
| The five common statistics . . . . .                                    | 295        |
| When to prefer mean(SD) vs median(IQR) . . . . .                        | 296        |
| What “N” alone does not tell you . . . . .                              | 296        |
| Three common beginner mistakes . . . . .                                | 297        |
| Small-cell suppression: when not to report a number . . . . .           | 298        |
| The audit habit for descriptive statistics . . . . .                    | 298        |
| Where this goes next . . . . .                                          | 298        |
| <b>Four Audit Categories for AI-Drafted EDA</b>                         | <b>299</b> |
| Where this fits . . . . .                                               | 299        |
| The four categories . . . . .                                           | 299        |
| Beginner-level examples for each . . . . .                              | 300        |
| What goes in an audit note . . . . .                                    | 301        |
| A worked audit note line . . . . .                                      | 302        |
| Stewardship recap: what goes in the provenance<br>statement . . . . .   | 302        |

|                                                           |            |
|-----------------------------------------------------------|------------|
| The audit habit . . . . .                                 | 303        |
| Where this goes next . . . . .                            | 303        |
| <b>In-Class Studio: Planted-Error Audit</b>               | <b>304</b> |
| Where this fits . . . . .                                 | 304        |
| Before you start . . . . .                                | 304        |
| Goal . . . . .                                            | 304        |
| Inputs . . . . .                                          | 305        |
| Activity Steps . . . . .                                  | 305        |
| Required Outputs (studio draft) . . . . .                 | 305        |
| Minimum Table 1 Requirements . . . . .                    | 306        |
| Completion Check . . . . .                                | 306        |
| Where this goes next . . . . .                            | 306        |
| <b>Assignment 6 Walkthrough and Reference</b>             | <b>307</b> |
| Where this fits . . . . .                                 | 307        |
| Deadline and link to M2 . . . . .                         | 307        |
| The six A6 deliverables, explained . . . . .              | 309        |
| Reproducibility checklist for A6 . . . . .                | 311        |
| A methods-note template you can reuse . . . . .           | 311        |
| A provenance/stewardship sentence you can reuse . . . . . | 311        |
| Five quick knowledge checks . . . . .                     | 312        |
| EDA quick reference . . . . .                             | 312        |
| Where this goes next . . . . .                            | 313        |
| <b>Week 8: Dashboard Prototypes for KT</b>                | <b>314</b> |
| <b>Week 8: Dashboard Prototypes for KT</b>                | <b>315</b> |
| Overview . . . . .                                        | 315        |
| Objectives . . . . .                                      | 316        |
| Connection . . . . .                                      | 316        |
| Dashboard Case Study: NHANES Equity . . . . .             | 316        |
| Reading order . . . . .                                   | 317        |
| Class plan . . . . .                                      | 317        |
| Student output . . . . .                                  | 318        |
| Links . . . . .                                           | 318        |
| Definition of done . . . . .                              | 318        |
| What students leave with . . . . .                        | 318        |
| <b>What a Dashboard Is, and Three Pathways</b>            | <b>319</b> |
| Goal . . . . .                                            | 319        |

|                                                                       |            |
|-----------------------------------------------------------------------|------------|
| A dashboard is not a report . . . . .                                 | 319        |
| Three pathways, three roles in this course . . . . .                  | 320        |
| Decision rule for your term project . . . . .                         | 320        |
| What “dashboard-style” means for Quarto . . . . .                     | 321        |
| What “demonstration only” means for Shiny . . . . .                   | 321        |
| Where this goes next . . . . .                                        | 322        |
| <b>From One Finding to a Prototype: The 1-1-1-1 Rule</b>              | <b>323</b> |
| Where this fits . . . . .                                             | 323        |
| The 1-1-1-1 rule . . . . .                                            | 323        |
| A worked example using your Week 7 Table 1 . . . . .                  | 324        |
| How to choose each field for your own project . . . . .               | 324        |
| What to write down before you build . . . . .                         | 326        |
| A note on scope creep . . . . .                                       | 326        |
| Where this goes next . . . . .                                        | 326        |
| <b>Starter Dashboard</b>                                              | <b>327</b> |
| Where this fits . . . . .                                             | 327        |
| Goal . . . . .                                                        | 327        |
| Audience Question . . . . .                                           | 327        |
| Setup . . . . .                                                       | 327        |
| Control . . . . .                                                     | 328        |
| Summary . . . . .                                                     | 329        |
| Visualization . . . . .                                               | 329        |
| Interpretation . . . . .                                              | 330        |
| Limitation . . . . .                                                  | 331        |
| Testing . . . . .                                                     | 331        |
| Privacy-Safe Publishing . . . . .                                     | 332        |
| Where this goes next . . . . .                                        | 332        |
| <b>Audience-Facing Text: What You Can and Cannot</b>                  |            |
| <b>Change for A7</b>                                                  | <b>333</b> |
| Where this fits . . . . .                                             | 333        |
| The rule, in one sentence . . . . .                                   | 333        |
| Allowed changes (text-only) . . . . .                                 | 333        |
| Not allowed for A7 (analytic-only changes) . . . . .                  | 334        |
| Before / after examples . . . . .                                     | 335        |
| What to record in your A7 <code>before-after-text.md</code> . . . . . | 336        |
| Where this goes next . . . . .                                        | 336        |

|                                                             |                |
|-------------------------------------------------------------|----------------|
| <b>Testing, Privacy-Safe Publishing, and the Shiny Demo</b> | <b>337</b>     |
| Where this fits . . . . .                                   | 337            |
| Part 1 — Testing the starter dashboard . . . . .            | 337            |
| Part 2 — Privacy-safe publishing . . . . .                  | 340            |
| Part 3 — The Shiny exemplar (demonstration only) . . . . .  | 341            |
| Where this goes next . . . . .                              | 342            |
| <br><b>In-Class Studio: Dashboard Adaptation</b>            | <br><b>343</b> |
| Where this fits . . . . .                                   | 343            |
| Before you start . . . . .                                  | 343            |
| Goal . . . . .                                              | 343            |
| The 60-minute studio sequence . . . . .                     | 344            |
| At the end of the studio . . . . .                          | 346            |
| Required studio outputs (drafts) . . . . .                  | 346            |
| Completion check . . . . .                                  | 346            |
| Where this goes next . . . . .                              | 347            |
| <br><b>Assignment 7 Walkthrough and Reference</b>           | <br><b>348</b> |
| Where this fits . . . . .                                   | 348            |
| Deadline and the decision rule . . . . .                    | 348            |
| The nine A7 deliverables, explained . . . . .               | 348            |
| Reproducibility checklist for A7 . . . . .                  | 351            |
| A pathway decision template you can reuse . . . . .         | 351            |
| Privacy-safe publishing checklist . . . . .                 | 351            |
| Paths you will use repeatedly . . . . .                     | 352            |
| Five quick knowledge checks . . . . .                       | 352            |
| Where this goes next . . . . .                              | 352            |
| <br><b>Week 9: Communication of Scientific Findings</b>     | <br><b>353</b> |
| <br><b>Week 9: Communication of Scientific Findings</b>     | <br><b>354</b> |
| Overview . . . . .                                          | 354            |
| Objectives . . . . .                                        | 355            |
| Connection . . . . .                                        | 355            |
| Dashboard Case Study: NHANES Equity . . . . .               | 356            |
| Reading Order . . . . .                                     | 356            |
| Class Plan . . . . .                                        | 357            |
| Student Output . . . . .                                    | 357            |
| Links . . . . .                                             | 357            |
| Definition of Done . . . . .                                | 357            |

|                                                |            |
|------------------------------------------------|------------|
| What Students Leave With . . . . .             | 358        |
| <b>Why Communication Is KT</b>                 | <b>359</b> |
| Where This Fits . . . . .                      | 359        |
| The Week 9 Task . . . . .                      | 359        |
| Match the Message to the Audience . . . . .    | 360        |
| What Counts as Evidence . . . . .              | 360        |
| Keep the Claim Small . . . . .                 | 361        |
| AI Use . . . . .                               | 361        |
| Check Yourself . . . . .                       | 361        |
| <b>Claim, Evidence, Limitation</b>             | <b>363</b> |
| The Core Pattern . . . . .                     | 363        |
| Build the Table First . . . . .                | 363        |
| Reuse Earlier Work . . . . .                   | 364        |
| Plain-Language Revision . . . . .              | 364        |
| What Not to Do . . . . .                       | 365        |
| Slide-Level Check . . . . .                    | 366        |
| <b>Revealjs Starter</b>                        | <b>367</b> |
| Where This Fits . . . . .                      | 367        |
| Copy, Then Render . . . . .                    | 367        |
| Copy the Starter into Your A8 Folder . . . . . | 367        |
| The Three-Slide Story . . . . .                | 368        |
| Add Evidence . . . . .                         | 368        |
| Add a Source Note . . . . .                    | 369        |
| Speaker Notes . . . . .                        | 369        |
| Before Submitting . . . . .                    | 369        |
| <b>Peer Feedback</b>                           | <b>370</b> |
| Goal . . . . .                                 | 370        |
| How and When to Exchange Feedback . . . . .    | 370        |
| Three-Slide Structure . . . . .                | 371        |
| Plain-Language Check . . . . .                 | 371        |
| Peer Feedback Form . . . . .                   | 371        |
| Citation and Accessibility Checklist . . . . . | 372        |
| Good Feedback Sounds Like . . . . .            | 372        |
| Student Output . . . . .                       | 373        |
| <b>Assignment 8 Walkthrough</b>                | <b>374</b> |
| Where This Fits . . . . .                      | 374        |
| The Six A8 Files . . . . .                     | 374        |

|                                                  |            |
|--------------------------------------------------|------------|
| Reproducibility Check . . . . .                  | 376        |
| Accessibility Check . . . . .                    | 377        |
| Where This Goes Next . . . . .                   | 377        |
| <b>Week 10: Writing &amp; Publishing Reports</b> | <b>378</b> |
| <b>Week 10: Writing &amp; Publishing Reports</b> | <b>379</b> |
| Overview . . . . .                               | 379        |
| Objectives . . . . .                             | 380        |
| Connection . . . . .                             | 380        |
| Dashboard Case Study: NHANES Equity . . . . .    | 380        |
| Reading Order . . . . .                          | 381        |
| Class Plan . . . . .                             | 381        |
| Student Output . . . . .                         | 382        |
| Links . . . . .                                  | 382        |
| Definition of Done . . . . .                     | 382        |
| What Students Leave With . . . . .               | 383        |
| <b>Report Goals</b>                              | <b>384</b> |
| Where This Fits . . . . .                        | 384        |
| Who Is the Report For? . . . . .                 | 384        |
| The Report Is Not Bigger Slides . . . . .        | 384        |
| Reuse Earlier Work . . . . .                     | 385        |
| Keep the Scope Descriptive . . . . .             | 385        |
| Report and M3 . . . . .                          | 385        |
| Check Yourself . . . . .                         | 385        |
| <b>Report Structure</b>                          | <b>387</b> |
| A Beginner-Friendly Structure . . . . .          | 387        |
| Question . . . . .                               | 387        |
| Data Source . . . . .                            | 388        |
| Methods . . . . .                                | 388        |
| Results . . . . .                                | 388        |
| Limitations . . . . .                            | 389        |
| Reproducibility and AI Use . . . . .             | 389        |
| <b>Report Template: NHANES Equity Summary</b>    | <b>390</b> |
| Standalone YAML . . . . .                        | 390        |
| Purpose . . . . .                                | 390        |
| Question . . . . .                               | 390        |
| Data Source . . . . .                            | 391        |

|                                                 |            |
|-------------------------------------------------|------------|
| Methods . . . . .                               | 391        |
| Results . . . . .                               | 392        |
| Limitations . . . . .                           | 394        |
| Plain-Language Summary . . . . .                | 394        |
| AI-Use and Audit Note . . . . .                 | 394        |
| Publishing Checklist . . . . .                  | 394        |
| Grounding Audit Prompt . . . . .                | 395        |
| <b>Citations and Grounding</b>                  | <b>396</b> |
| Why This Matters . . . . .                      | 396        |
| Citation Workflow . . . . .                     | 396        |
| Grounding Audit . . . . .                       | 397        |
| What Counts as Evidence . . . . .               | 397        |
| AI-Use Check . . . . .                          | 397        |
| Before Moving On . . . . .                      | 398        |
| <b>Private Preview</b>                          | <b>399</b> |
| How the Pieces Fit Together . . . . .           | 399        |
| Default Path . . . . .                          | 399        |
| Pre-Commit Privacy Checklist . . . . .          | 400        |
| When Public Publishing Is Safe . . . . .        | 400        |
| GitHub Pages as an Optional Extension . . . . . | 401        |
| Private Preview Checklist . . . . .             | 402        |
| Common Errors and Fixes . . . . .               | 402        |
| Final Portfolio Connection . . . . .            | 403        |
| <b>Assignment 9 and M3</b>                      | <b>404</b> |
| Where This Fits . . . . .                       | 404        |
| A9 Submission Files . . . . .                   | 404        |
| Edit These, Not Those . . . . .                 | 407        |
| M3 Submission Files . . . . .                   | 407        |
| Feedback Request . . . . .                      | 408        |
| Before Submitting . . . . .                     | 408        |
| <b>Week 11: Portfolio Surgery</b>               | <b>410</b> |
| <b>Week 11: Portfolio Surgery</b>               | <b>411</b> |
| Overview . . . . .                              | 411        |
| Objectives . . . . .                            | 412        |
| Connection . . . . .                            | 412        |
| Dashboard Case Study: NHANES Equity . . . . .   | 412        |

|                                                      |            |
|------------------------------------------------------|------------|
| Reading Order . . . . .                              | 413        |
| Class Plan . . . . .                                 | 413        |
| Student Output . . . . .                             | 414        |
| Links . . . . .                                      | 414        |
| Definition of Done . . . . .                         | 414        |
| What Students Leave With . . . . .                   | 414        |
| <b>Surgery Goals</b>                                 | <b>415</b> |
| Where This Fits . . . . .                            | 415        |
| The Core Standard . . . . .                          | 415        |
| Do Not Add New Work Unless It Reduces Risk . . . . . | 416        |
| Group Artifact Mindset . . . . .                     | 416        |
| Check Yourself . . . . .                             | 416        |
| <b>Fresh Codespace Test</b>                          | <b>417</b> |
| Goal . . . . .                                       | 417        |
| Roles . . . . .                                      | 417        |
| Runbook . . . . .                                    | 417        |
| Failure Log . . . . .                                | 418        |
| Example Failure Log . . . . .                        | 418        |
| NHANES Exemplar Checks . . . . .                     | 419        |
| Optional Python Pathway . . . . .                    | 419        |
| Final Evidence and Repair Checklist . . . . .        | 420        |
| <b>README and Repo Hygiene</b>                       | <b>421</b> |
| Why This Matters . . . . .                           | 421        |
| README Minimums . . . . .                            | 421        |
| Weak vs Strong: A README Example . . . . .           | 422        |
| Path Hygiene . . . . .                               | 422        |
| Dependency Notes . . . . .                           | 423        |
| Output Hygiene . . . . .                             | 423        |
| Git Hygiene . . . . .                                | 423        |
| Quick Repair Pattern . . . . .                       | 424        |
| <b>Peer Review and M4</b>                            | <b>425</b> |
| Where This Fits . . . . .                            | 425        |
| What to Review . . . . .                             | 425        |
| Review Sequence . . . . .                            | 425        |
| Useful Feedback . . . . .                            | 426        |
| M4 Files . . . . .                                   | 426        |
| AI Use in Peer Review . . . . .                      | 426        |



|                                                  |            |
|--------------------------------------------------|------------|
| Before Sending Feedback . . . . .                | 427        |
| <b>Optional Pathways</b>                         | <b>428</b> |
| Keep the Core First . . . . .                    | 428        |
| Optional Shiny . . . . .                         | 429        |
| Optional Static Dashboard . . . . .              | 429        |
| Optional Python . . . . .                        | 429        |
| Public Deployment . . . . .                      | 430        |
| Decision Rule . . . . .                          | 430        |
| <b>Portfolio Surgery and the Final Portfolio</b> | <b>431</b> |
| Where This Fits . . . . .                        | 431        |
| What To Produce in the Session . . . . .         | 431        |
| Final Portfolio Bridge . . . . .                 | 433        |
| Public or Private Repository . . . . .           | 434        |
| Before You Finish the Session . . . . .          | 434        |
| Before Final Portfolio . . . . .                 | 434        |
| <b>Project Presentation (Online)</b>             | <b>435</b> |
| <b>Project Presentation (Narrated Deck)</b>      | <b>436</b> |
| Overview . . . . .                               | 436        |
| What To Record . . . . .                         | 436        |
| Slide Submission . . . . .                       | 437        |
| Slide Checklist . . . . .                        | 437        |
| Portfolio Reminder . . . . .                     | 437        |
| Definition Of Done . . . . .                     | 438        |
| <b>Assignments and Milestones</b>                | <b>439</b> |
| <b>Assignment Schedule</b>                       | <b>440</b> |
| How to read this page . . . . .                  | 440        |
| Term at a glance . . . . .                       | 440        |
| Individual assignments only . . . . .            | 443        |
| Group milestones only . . . . .                  | 443        |
| For the instructor: setting up Canvas . . . . .  | 444        |
| <b>Assignments</b>                               | <b>445</b> |
| Assignment Map . . . . .                         | 445        |
| Grading Logic . . . . .                          | 447        |

|                                           |            |
|-------------------------------------------|------------|
| Common Reproducibility Contract . . . . . | 447        |
| How To Read Submission Lists . . . . .    | 447        |
| Common AI-Use Expectation . . . . .       | 448        |
| <b>Assignment 1: Modern Workflows</b>     | <b>450</b> |
| Purpose . . . . .                         | 450        |
| Due Timing . . . . .                      | 450        |
| Learning Objectives . . . . .             | 450        |
| Inputs . . . . .                          | 450        |
| Tasks . . . . .                           | 451        |
| AI-Use Expectations . . . . .             | 451        |
| Reproducibility Requirements . . . . .    | 451        |
| What To Submit . . . . .                  | 452        |
| Grading Checklist . . . . .               | 452        |
| Definition of Done . . . . .              | 453        |
| <b>Assignment 2: R With AI</b>            | <b>454</b> |
| Purpose . . . . .                         | 454        |
| Learning Objectives . . . . .             | 454        |
| Inputs . . . . .                          | 454        |
| Tasks . . . . .                           | 455        |
| AI-Use Expectations . . . . .             | 455        |
| Reproducibility Requirements . . . . .    | 455        |
| What To Submit . . . . .                  | 456        |
| Submission Route . . . . .                | 456        |
| Grading Checklist . . . . .               | 457        |
| Definition of Done . . . . .              | 457        |
| <b>Assignment 3: Git Collaboration</b>    | <b>458</b> |
| Purpose . . . . .                         | 458        |
| Learning Objectives . . . . .             | 458        |
| Inputs . . . . .                          | 458        |
| Tasks . . . . .                           | 459        |
| AI-Use Expectations . . . . .             | 459        |
| Reproducibility Requirements . . . . .    | 460        |
| What To Submit . . . . .                  | 460        |
| Submission Route . . . . .                | 460        |
| Grading Checklist . . . . .               | 460        |
| Definition of Done . . . . .              | 461        |

|                                                 |                |
|-------------------------------------------------|----------------|
| <b>Assignment 4: Polyglot Parity</b>            | <b>462</b>     |
| Purpose . . . . .                               | 462            |
| Learning Objectives . . . . .                   | 462            |
| Inputs . . . . .                                | 462            |
| Tasks . . . . .                                 | 463            |
| AI-Use Expectations . . . . .                   | 463            |
| Reproducibility Requirements . . . . .          | 463            |
| What To Submit . . . . .                        | 464            |
| Submission Route . . . . .                      | 464            |
| Grading Checklist . . . . .                     | 464            |
| Definition of Done . . . . .                    | 465            |
| <br><b>Assignment 5: Visualization Audit</b>    | <br><b>466</b> |
| Purpose . . . . .                               | 466            |
| Learning Objectives . . . . .                   | 466            |
| Inputs . . . . .                                | 466            |
| Tasks . . . . .                                 | 467            |
| AI-Use Expectations . . . . .                   | 467            |
| Reproducibility Requirements . . . . .          | 467            |
| What To Submit . . . . .                        | 468            |
| Submission Route . . . . .                      | 468            |
| Grading Checklist . . . . .                     | 469            |
| Definition of Done . . . . .                    | 469            |
| <br><b>Assignment 6: EDA and AI Audit</b>       | <br><b>470</b> |
| Purpose . . . . .                               | 470            |
| Learning Objectives . . . . .                   | 470            |
| Inputs . . . . .                                | 470            |
| Tasks . . . . .                                 | 471            |
| AI-Use Expectations . . . . .                   | 471            |
| Reproducibility Requirements . . . . .          | 471            |
| What To Submit . . . . .                        | 472            |
| Submission Route . . . . .                      | 472            |
| Grading Checklist . . . . .                     | 472            |
| Definition of Done . . . . .                    | 474            |
| <br><b>Assignment 7: Dashboard KT Prototype</b> | <br><b>475</b> |
| Purpose . . . . .                               | 475            |
| Learning Objectives . . . . .                   | 475            |
| Inputs . . . . .                                | 476            |
| Tasks . . . . .                                 | 476            |

|                                               |            |
|-----------------------------------------------|------------|
| AI-Use Expectations . . . . .                 | 477        |
| Reproducibility Requirements . . . . .        | 477        |
| What To Submit . . . . .                      | 477        |
| Submission Route . . . . .                    | 478        |
| Grading Checklist . . . . .                   | 478        |
| Definition of Done . . . . .                  | 479        |
| <b>Assignment 8: Scientific Communication</b> | <b>480</b> |
| Purpose . . . . .                             | 480        |
| Learning Objectives . . . . .                 | 480        |
| Inputs . . . . .                              | 480        |
| Tasks . . . . .                               | 481        |
| AI-Use Expectations . . . . .                 | 481        |
| Reproducibility Requirements . . . . .        | 482        |
| What To Submit . . . . .                      | 482        |
| Submission Route . . . . .                    | 483        |
| Grading Checklist . . . . .                   | 483        |
| Definition of Done . . . . .                  | 483        |
| <b>Assignment 9: Reproducible Report</b>      | <b>484</b> |
| Purpose . . . . .                             | 484        |
| Learning Objectives . . . . .                 | 484        |
| Inputs . . . . .                              | 484        |
| Tasks . . . . .                               | 485        |
| AI-Use Expectations . . . . .                 | 485        |
| Reproducibility Requirements . . . . .        | 486        |
| What To Submit . . . . .                      | 486        |
| Submission Route . . . . .                    | 487        |
| Grading Checklist . . . . .                   | 487        |
| Definition of Done . . . . .                  | 488        |
| <b>Milestones</b>                             | <b>489</b> |
| <b>Milestones</b>                             | <b>490</b> |
| Submission Contract . . . . .                 | 490        |
| Milestone Map . . . . .                       | 490        |
| Project Points and Score Ownership . . . . .  | 491        |
| Reproducibility Expectations . . . . .        | 492        |
| AI-Use Expectations . . . . .                 | 492        |
| Definition Of Done . . . . .                  | 492        |

|                                             |            |
|---------------------------------------------|------------|
| <b>M0: Group Formation</b>                  | <b>493</b> |
| Purpose . . . . .                           | 493        |
| Due Timing . . . . .                        | 493        |
| Required Deliverables . . . . .             | 494        |
| What To Submit . . . . .                    | 494        |
| File Guidance . . . . .                     | 494        |
| Reproducibility Expectations . . . . .      | 495        |
| AI-Use Expectations . . . . .               | 495        |
| Grading Checklist . . . . .                 | 496        |
| Definition Of Done . . . . .                | 496        |
| <b>M1: Project Proposal</b>                 | <b>497</b> |
| Purpose . . . . .                           | 497        |
| Due Timing . . . . .                        | 497        |
| Points and Score Ownership . . . . .        | 497        |
| Required Deliverables . . . . .             | 498        |
| What To Submit . . . . .                    | 499        |
| Proposal Guidance . . . . .                 | 499        |
| Reproducibility Expectations . . . . .      | 500        |
| AI-Use Expectations . . . . .               | 500        |
| Full-Credit Preparation Checklist . . . . . | 501        |
| Definition Of Done . . . . .                | 501        |
| <b>M2: Preliminary Analysis</b>             | <b>502</b> |
| Purpose . . . . .                           | 502        |
| Due Timing . . . . .                        | 502        |
| Points and Score Ownership . . . . .        | 502        |
| Required Deliverables . . . . .             | 503        |
| What To Submit . . . . .                    | 504        |
| Analysis Requirements . . . . .             | 504        |
| Reproducibility Expectations . . . . .      | 504        |
| AI-Use Expectations . . . . .               | 505        |
| Full-Credit Preparation Checklist . . . . . | 505        |
| Definition Of Done . . . . .                | 505        |
| <b>M3: Project Update</b>                   | <b>506</b> |
| Purpose . . . . .                           | 506        |
| Due Timing . . . . .                        | 506        |
| Points and Score Ownership . . . . .        | 506        |
| Required Deliverables . . . . .             | 507        |
| What To Submit . . . . .                    | 508        |

|                                                  |            |
|--------------------------------------------------|------------|
| Update Requirements . . . . .                    | 508        |
| Reproducibility Expectations . . . . .           | 509        |
| AI-Use Expectations . . . . .                    | 509        |
| Full-Credit Preparation Checklist . . . . .      | 509        |
| Definition Of Done . . . . .                     | 510        |
| <b>M4: Peer Review</b>                           | <b>511</b> |
| Purpose . . . . .                                | 511        |
| Due Timing . . . . .                             | 511        |
| Points and Score Ownership . . . . .             | 511        |
| Required Deliverables . . . . .                  | 512        |
| What To Submit . . . . .                         | 513        |
| Review Requirements . . . . .                    | 513        |
| Reproducibility Expectations . . . . .           | 514        |
| AI-Use Expectations . . . . .                    | 514        |
| Full-Credit Preparation Checklist . . . . .      | 515        |
| Definition Of Done . . . . .                     | 515        |
| <b>Presentation (Final Portfolio Component)</b>  | <b>516</b> |
| Purpose . . . . .                                | 516        |
| Due Timing . . . . .                             | 516        |
| Points and Score Ownership . . . . .             | 517        |
| Required Deliverables . . . . .                  | 517        |
| What To Submit . . . . .                         | 518        |
| Slide Requirements . . . . .                     | 518        |
| Source And Limitations Note . . . . .            | 518        |
| Peer Feedback Received . . . . .                 | 519        |
| Reproducibility Expectations . . . . .           | 519        |
| AI-Use Expectations . . . . .                    | 519        |
| Presentation Preparation Checklist . . . . .     | 519        |
| Definition Of Done . . . . .                     | 520        |
| <b>Final Portfolio</b>                           | <b>521</b> |
| Purpose . . . . .                                | 521        |
| Due Timing . . . . .                             | 521        |
| Points and Score Ownership . . . . .             | 521        |
| Required Deliverables . . . . .                  | 522        |
| Repository Access Note . . . . .                 | 523        |
| What To Submit . . . . .                         | 524        |
| Public And Private Repository Guidance . . . . . | 524        |
| Reproducibility Checklist . . . . .              | 525        |

|                                                   |            |
|---------------------------------------------------|------------|
| What A Strong Project README Looks Like . . . . . | 525        |
| AI-Use Expectations . . . . .                     | 526        |
| Grading Checklist . . . . .                       | 526        |
| Definition Of Done . . . . .                      | 526        |
| <b>Final Portfolio Checklist</b>                  | <b>527</b> |
| Repository Access . . . . .                       | 527        |
| Project README . . . . .                          | 527        |
| Data And Stewardship . . . . .                    | 527        |
| Reproducibility . . . . .                         | 528        |
| Communication . . . . .                           | 528        |
| AI Use . . . . .                                  | 528        |
| Definition Of Done . . . . .                      | 528        |
| <b>Repository Sharing Guidance</b>                | <b>530</b> |
| Public Repository . . . . .                       | 530        |
| Private Repository . . . . .                      | 530        |
| Safe Publishing Check . . . . .                   | 531        |
| <b>Worked Example (NHANES)</b>                    | <b>532</b> |
| <b>Worked Example (NHANES)</b>                    | <b>533</b> |
| <b>Worked Example (NHANES)</b>                    | <b>534</b> |
| The milestones . . . . .                          | 534        |
| <b>Example: M0 — Group Formation</b>              | <b>535</b> |
| <b>Example: M0 — Group Formation</b>              | <b>536</b> |
| AI-use note . . . . .                             | 536        |
| Codespace check . . . . .                         | 537        |
| Communication plan . . . . .                      | 537        |
| Group formation . . . . .                         | 537        |
| Repository access evidence . . . . .              | 538        |
| <b>Example: M1 — Project Proposal</b>             | <b>539</b> |
| <b>Example: M1 — Project Proposal</b>             | <b>540</b> |
| AI-use note . . . . .                             | 540        |
| Data intake card . . . . .                        | 541        |
| Initial visual plan . . . . .                     | 542        |

|                                                        |            |
|--------------------------------------------------------|------------|
| Project proposal . . . . .                             | 542        |
| Reproducibility plan . . . . .                         | 544        |
| <b>Example: M2 — Preliminary Analysis</b>              | <b>545</b> |
| <b>Example: M2 — Preliminary Analysis</b>              | <b>546</b> |
| AI-use note . . . . .                                  | 546        |
| Methods note . . . . .                                 | 547        |
| <b>Example: M3 — Project Update</b>                    | <b>548</b> |
| <b>Example: M3 — Project Update</b>                    | <b>549</b> |
| AI-use note . . . . .                                  | 549        |
| Dashboard preview . . . . .                            | 550        |
| Feedback request . . . . .                             | 550        |
| Project update . . . . .                               | 551        |
| Reproducibility note . . . . .                         | 551        |
| <b>Example: M4 — Peer Review</b>                       | <b>552</b> |
| <b>Example: M4 — Peer Review</b>                       | <b>553</b> |
| AI-use note . . . . .                                  | 553        |
| Auditability check . . . . .                           | 554        |
| KT feedback . . . . .                                  | 554        |
| Peer review . . . . .                                  | 555        |
| Reproducibility check . . . . .                        | 556        |
| <b>Example: Presentation — Narrated Deck</b>           | <b>558</b> |
| <b>Example: Presentation — Narrated Deck</b>           | <b>559</b> |
| AI-use note . . . . .                                  | 559        |
| Peer feedback received on the NHANES project . . . . . | 560        |
| Presentation checklist . . . . .                       | 560        |
| Source and limitations note . . . . .                  | 561        |
| <b>Example: Final Portfolio</b>                        | <b>562</b> |
| <b>Example: Final Portfolio</b>                        | <b>563</b> |
| Final project AI-use audit . . . . .                   | 563        |
| Final Portfolio Checklist . . . . .                    | 564        |
| Final risk list . . . . .                              | 566        |
| Repository access note . . . . .                       | 566        |



|                                |            |
|--------------------------------|------------|
| <b>Reference: Data Sources</b> | <b>568</b> |
| <b>NHIS</b>                    | <b>569</b> |
| <b>NHANES</b>                  | <b>577</b> |
| <b>BRFSS</b>                   | <b>586</b> |
| <b>CCS</b>                     | <b>593</b> |
| <b>CADS</b>                    | <b>599</b> |
| <b>CTNS</b>                    | <b>604</b> |
| <b>CSADS</b>                   | <b>606</b> |
| <b>PHAC Health Infobase</b>    | <b>610</b> |
| <b>CCDSS</b>                   | <b>614</b> |
| <b>GHO</b>                     | <b>619</b> |
| <b>Open Portal</b>             | <b>627</b> |
| <b>BC Data Catalogue</b>       | <b>631</b> |

# The Project

Welcome to the course notes for **Health Data Science: AI and Knowledge Translation**.

This platform bridges the gap between traditional epidemiology and modern data science. Unlike standard textbooks that focus on a single language, this resource adopts a **“Polyglot”** approach—teaching R and Python side-by-side—and integrates **Artificial Intelligence** directly into the workflow.

Whether you are a health researcher looking to modernize your skills or a data scientist seeking to understand epidemiological rigor, you are in the right place.

Here, we offer:

- **“Zero to AI Co-Pilot” Training:** Learn to use Large Language Models (LLMs) to generate, debug, and translate code while critically auditing them for bias and hallucinations.
- **Real-World Data:** Guides on navigating “messy” open data sources such as NHANES and WHO, rather than sanitized “toy” datasets.
- **Knowledge Translation:** Step-by-step walkthroughs on building a Quarto dashboard-style KT product and dynamic reports; Shiny is shown as an optional demonstration.

## What We Aim to Achieve

We are on a mission to:

- **Democratize Compute Access:** Eliminate hardware barriers by using GitHub Codespaces — a cloud-native environment that runs on any modern web browser, with no local installation required.
- **Polyglot Awareness:** R is the required core. Week 5 reads short Python/pandas examples and runs AI-assisted translation parity checks; Python is awareness, not a parallel curriculum.
- **Teach AI Audit Habits:** Shift the focus from accepting AI-generated code to **rigorous code review**, security validation, and reproducibility.
- **Enable Professional Portfolios:** Help students transform static analyses into public-facing data science portfolios.

## Dive into Our Modules

Embark on a journey through our structured modules, designed to take you from data literacy to advanced AI workflows.

### Note

The tutorials are designed with a consistent structure to provide a cohesive learning experience. Here is what you can expect in each chapter:

1. **Overview:** A concise summary of learning objectives and topics.
2. **Concepts:** Core theoretical materials (slides, video lessons).
3. **R Tutorials:** In-depth R walkthroughs with AI-assisted code drafting and auditing. Week 5 adds short Python/pandas examples for polyglot awareness only.
4. **AI Auditing:** Specific sections dedicated to verifying and “stress-testing” AI-generated code for the chapter’s topic.
5. **Knowledge Check:** Ungraded self-assessments to reinforce key concepts.

6. **Practice Exercises:** Hands-on tasks using real-world health data to build your portfolio.

## How Our Content is Presented

All our resources are hosted on an easy-to-access GitHub page. The format? Engaging text, reproducible code, clear outputs, and crisp videos that distill complex topics.

This document is created using [Quarto](#), allowing us to weave analysis, code, and interpretation into a single reproducible document.

## Open Copyright License

[CC-BY 4.0](#)



## How to Cite

The BibTeX format can be downloaded from [here](#).

## About the Course

## Course Overview

# Course Overview

**Health Data Science: AI and Knowledge Translation** is a beginner-friendly, open course that teaches you to work with real health data using R, Quarto, Git, GitHub Codespaces, and AI coding assistants — and to communicate results clearly, reproducibly, and responsibly.

## Note

This is an **open educational resource (OER)**. You can follow it independently, or an instructor may adopt it for a specific course. If you are taking this in a specific offering, that course's **Canvas (or equivalent) is the source of truth** for the schedule, deadlines, grading, and your instructor's contact details. This overview intentionally leaves those out so the material stays useful to any learner.

## What you'll learn

By working through this course you will be able to:

1. Explain key ethical, privacy, security, Indigenous Data Sovereignty, and knowledge-translation issues in health data science, including responsible use of generative AI tools.
2. Use GitHub Codespaces, Quarto, Git, and GitHub to organize, document, version, and reproduce a small health-data analysis.
3. Import, clean, summarize, and visualize health-related data using R and the tidyverse.
4. Read and compare short Python/pandas examples with AI assistance, recognize common R Python translation

traps, and explain when Python may be the more appropriate tool.

5. Use generative AI as a coding assistant while critically auditing its output for coding errors, hallucinated claims, bias, security risks, and reproducibility problems.
6. Conduct a bounded exploratory data analysis and communicate descriptive findings using clear tables, figures, narrative interpretation, and appropriate limitations.
7. Create a dashboard-style knowledge-translation product and a reproducible Quarto report for a defined audience.
8. Use a shared devcontainer and dependency documentation so analyses can be rerun in a fresh Codespace.
9. Give constructive peer feedback on the clarity, reproducibility, auditability, and knowledge-translation value of a project.

## Who it's for

**No prior experience required.** This is a “Zero to AI Co-Pilot” course: you do not need previous experience with R, Python, statistics, or programming. Comfort with a web browser, basic file management, and a willingness to troubleshoot are assumed; everything else is taught from the ground up.

## How the course works

- **Cloud-first.** All coding runs in **GitHub Codespaces** — a cloud computer you reach from any modern web browser. No local installation is required.
- **R is the core language**, with **Quarto** for reproducible documents; short **Python/pandas** examples build polyglot awareness.
- **AI as a co-pilot, always audited.** You use LLMs to draft, debug, and translate code, then check their output against your data and the documentation.
- **Reproducible by default.** Work is version-controlled with Git and designed to re-run in a fresh environment.



## What the course is built around

1. **The Full Data Lifecycle** — from ethical data access and cleaning to secure cloud computing and publication.
2. **Reproducible Science** — Git-based workflows, a shared environment, explicit dependency notes, and literate programming in Quarto.
3. **AI-Augmented Workflows** — using LLMs as coding co-pilots while auditing their output for errors, hallucinations, bias, security risks, and reproducibility failures.
4. **Knowledge Translation (KT)** — the “last mile”: turning results into dashboard-style products, visual stories, plain-language summaries, and policy-ready reports.
5. **Real-World Health Data** — hands-on work with messy, real datasets (for example, NHANES) rather than toy examples.

## Weekly topics

The material is a weekly progression; each week pairs a short concept seminar with hands-on studio work.

| Week | Topic                                                     |
|------|-----------------------------------------------------------|
| 0    | Onboarding: accounts, orientation, and setup (self-paced) |
| 1    | Health data, knowledge translation, and ethics            |
| 2    | Modern research workflows: Codespaces, Quarto, Git        |
| 3    | R with AI: data, functions, the tidyverse, debugging      |
| 4    | Git and collaboration: commits, branches, pull requests   |
| 5    | Polyglot awareness and R deepening (Python/pandas)        |
| 6    | Data visualization and AI-assisted visual audits          |
| 7    | Exploratory analysis, Table 1, and AI auditing            |
| 8    | Dashboard prototypes for knowledge translation            |
| 9    | Communicating scientific findings                         |
| 10   | Writing and publishing reproducible reports               |
| 11   | Portfolio surgery: reproducibility stress-testing         |

Use the sidebar to work through each week.

## About the developer

This open course was developed by **Dr. M. Ehsan Karim** (School of Population and Public Health, University of British Columbia), with the AI-KT team (Manyu Jain, Rainie Fu, Md. Belal Hossain). It is shared as an open educational resource for learners and adopting instructors.

## **Week 0: Asynchronous Onboarding**

# Week 0: Asynchronous Onboarding

## Purpose

Week 0 gets the account setup and orientation work out of the way before the first class. You do not need to install local software for this course. The default workflow uses a personal fork of the course template repository, GitHub Codespaces, Quarto, and Canvas.

**i** Four terms you will see all course

- **Repository (repo):** a project folder stored on GitHub. It holds all the files for your coursework — think of it as your cloud drive.
- **Codespace:** a ready-to-use computer in the cloud that opens in your browser, with R, Python, and Quarto already installed — think of it as a public computer in the library.
- **IDE (integrated development environment):** the program you write code in. This course uses VS Code, which is what a Codespace opens.
- **Quarto:** the tool that turns a document mixing text and code (a `.qmd` file) into a polished web page or report.

Week 2 walks through each of these step by step — you do not need to master them now.

## Required Tasks

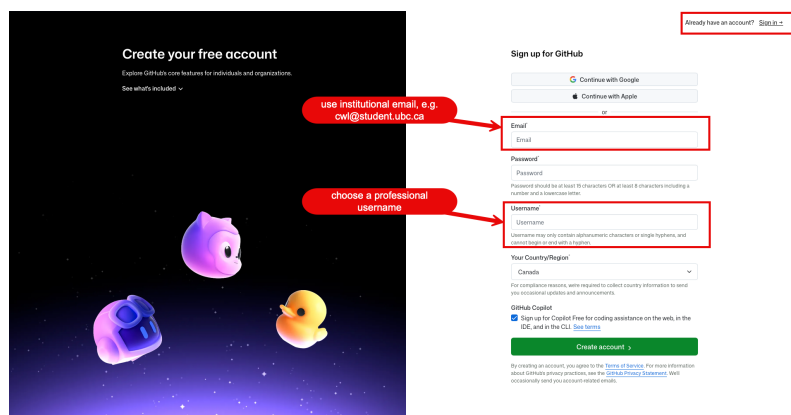
Complete these steps before Week 1:

1. Log in to Canvas and open the Week 0 module.
2. Watch the orientation video posted in Canvas.
3. Complete the pre-course survey posted in Canvas.
4. Create or confirm access to a GitHub account.
5. If eligible, apply for GitHub Education / Student Developer Pack, or confirm that you already have access.
6. Submit the Week 0 Canvas onboarding check with your GitHub username and GitHub Education status.

## GitHub Account

If you need an account, open <https://github.com/signup> and follow GitHub's account-creation prompts. If you already have an account, sign in at <https://github.com/login> and confirm that you can reach your profile.

Use a professional GitHub username that you are comfortable sharing in a course setting. If you already have a GitHub account, you may use it. You do not need to create a second account unless your current username or profile is not appropriate for coursework.



The image shows the GitHub 'Create your free account' page. On the left is a dark blue banner with the text 'Create your free account' and 'Explore GitHub's core features for individuals and organizations.' Below this is a red callout bubble pointing to the 'Email' field with the text 'use institutional email, e.g. owl@student.ubc.ca'. Another red callout bubble points to the 'Username' field with the text 'choose a professional username'. On the right is the 'Sign up for GitHub' form. It has a link 'Already have an account? Sign in.' at the top right. The form includes options to 'Continue with Google' or 'Continue with Apple', followed by 'or' and then input fields for 'Email', 'Password', and 'Username'. The 'Username' field has a note: 'Usernames may only contain alphanumeric characters or single hyphens, and cannot begin or end with a hyphen.' Below these is a 'Your Country/Region' dropdown menu set to 'Canada'. At the bottom, there is a checkbox for 'Sign up for CapScribe Free for coding assistance on the web, in the IDE, and in the CLI' with a link to 'See terms'. A green 'Create account' button is at the bottom right. Small text at the very bottom states: 'By creating an account, you agree to the Terms of Service. For more information about GitHub's privacy practices, see the GitHub Privacy Statement. We'll occasionally send you account related emails.'

Figure 1: The GitHub account creation page, where you choose your username and email

Before submitting the onboarding check:

- confirm you can log in to GitHub
- turn on two-factor authentication if GitHub prompts you
- make sure you know which email address is attached to the account
- record your GitHub username exactly as it appears in the profile URL
- use the same GitHub account and username throughout the course

## Username, Password, And Repository Privacy

A professional username usually follows a simple pattern, such as first-last or initials. For example, `maria-chen` or `j-smith-health`. Pick something you would be comfortable putting on a resume.

Protect your account:

- use a long, unique passphrase that you do not reuse on other sites
- consider a password manager so you do not have to memorize it
- turn on two-factor authentication (2FA) and keep it on; 2FA adds a second login step (a code from an app or text message) so your account stays protected even if your password is exposed

Think about repository privacy. A fork of a public repository is itself public, and GitHub does not let you switch a fork to private while it remains connected to its upstream. So if your work may include restricted, sensitive, or course-only materials, do not rely on changing a fork's visibility — instead create a separate private copy (for example with GitHub's **Import repository** option, or by using the template repository's **Use this template** button if the instructor enables it) and choose **Private**. A public repository should only be used when the instructor confirms it is safe. Making your work public is encouraged once the course privacy checklist passes, but it is never

required, and there is no penalty for staying private. Follow the workflow Canvas specifies for this offering.

The screenshot shows the GitHub 'Create a new repository' page. The 'General' section includes fields for 'Owner' (MANYAdici), 'Repository name' (test-repo), and 'Description'. The 'Configuration' section includes a 'Choose visibility' dropdown set to 'Private', a checked 'Add README' checkbox, a checked 'Add .gitignore' checkbox, and a 'No license' dropdown. A red box highlights the 'Choose visibility' dropdown, and another red box highlights the 'Private' option. A third red box highlights the 'Create repository' button at the bottom right.

Figure 2: The private-versus-public visibility setting when creating a GitHub repository

To learn more about GitHub basics or get help, see GitHub's getting-started guide: <https://docs.github.com/en/get-started>

## GitHub Education Benefits: Apply If Eligible

This course uses GitHub Codespaces every week. Codespaces usage is quota-based, so students are expected to apply for GitHub Education / Student Developer Pack if they are eligible:

<https://education.github.com/pack>

GitHub Education benefits may increase your available developer-tool allowance and provide access to additional student tools. This is helpful because the course uses Codespaces for weekly tutorials, assignments, and project work.

Approval is not required before Week 1. You can begin the course with a regular free GitHub account while your GitHub Education application is pending. Do not delay Week 0 onboarding while waiting for approval.

For the Week 0 Canvas onboarding check, report your GitHub Education status using one of the following options:

- approved
- applied and pending
- already had access
- not eligible / unable to apply
- unsure and need help

## **UBC Students**

If you are a UBC student, use your UBC student email address when applying, if available. For example, this may be your `@student.ubc.ca` email address or another UBC-issued student email connected to your student record.

Before applying:

- log in to your GitHub account
- add your UBC student email to your GitHub account, if available
- verify the UBC student email address in GitHub
- have proof of current student status ready in case GitHub asks for verification
- continue using the same GitHub username for this course

If you already have a GitHub account with a non-UBC email, you usually do not need to create a second account. Instead, add your UBC student email to the existing account and verify it, if GitHub allows. Use one consistent GitHub username for all course work.

## **Students Outside UBC**

If you are using this open course book outside UBC, follow the same general process with your own institution's student email address and proof of current enrollment. The exact verification process may differ by institution, country, and GitHub's current requirements.



## Codespaces Usage Reminder

Even with GitHub Education benefits, Codespaces usage is not unlimited. To avoid running out of usage during the term:

- stop your Codespace when you finish working
- reopen an existing Codespace instead of creating a new one each time
- use the default course machine type unless instructed otherwise
- monitor your remaining Codespaces usage in your GitHub billing or settings page
- contact the teaching team before adding personal billing information

## Existing-Account Help

If you already have a GitHub account but cannot access it:

- try GitHub account recovery before creating a new account
- use the email address most likely attached to the account
- check whether your browser is logged into a different personal account
- ask for help through the Canvas course support channel if you are blocked

If you create a new account, use that account consistently for course repositories.

## Course Repository Entry Point

Create your own copy of the course template repository on GitHub. This copy is called a fork. You will make changes in your own fork so that your work does not affect the original course repository.

The current course workflow uses a fork-based setup:

1. Open the course template repository using the link provided in Canvas. If you do not have Canvas access, use the direct GitHub repository link: <https://github.com/ehsanx/HDSx-workspace>.
2. Click **Fork**.
3. On the “Create a new fork” page, make sure the **Owner** is set to the GitHub account you submitted in the onboarding check (not another personal account or an organization), then create the fork.
4. Verify the fork landed in the correct account: the repository URL should read `https://github.com/<your-username>/HDSx-workspace`, and the page header should show **forked from ehsanx/HDSx-workspace** under your own username. If the owner is wrong, delete the fork and repeat the step.
5. Launch Codespaces from your fork.
6. Commit and sync changes to your fork.
7. Submit your fork repository link on Canvas.

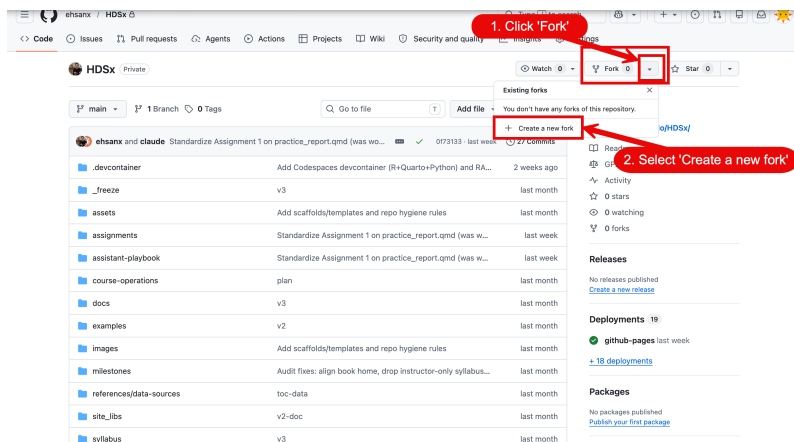


Figure 3: The Fork button on the course template repository, used to create your own copy

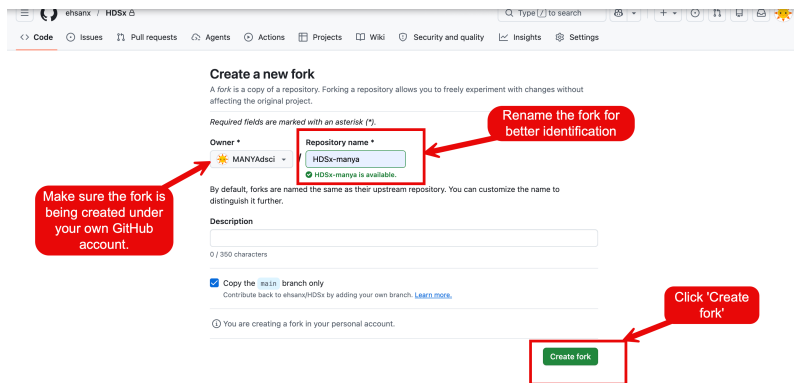


Figure 4: Choosing the correct Owner on the Create a new fork page

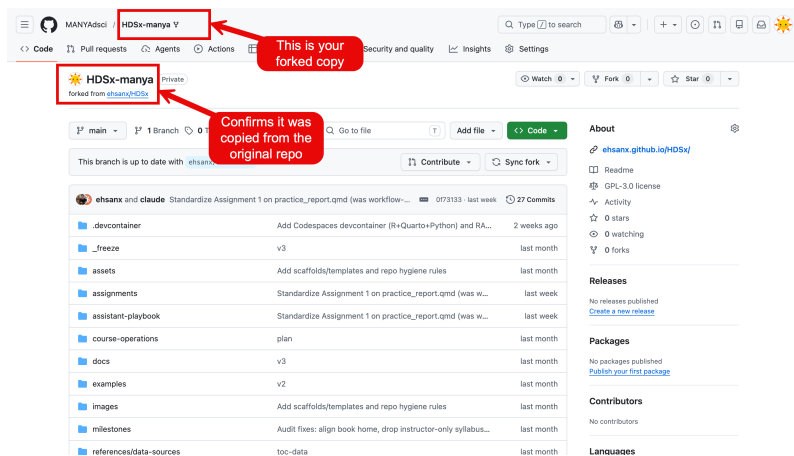


Figure 5: Confirming the fork was created under your own account

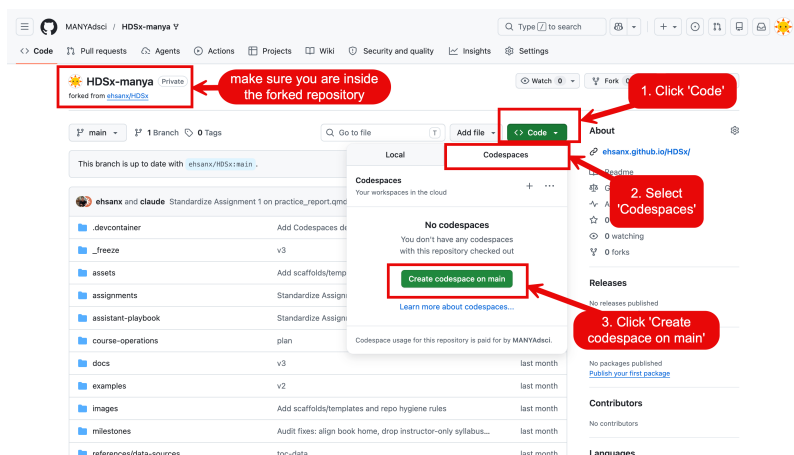


Figure 6: Launching a full-screen Codespace from your fork using the Code button

If a future offering uses an instructor-managed repository system, Canvas will say so explicitly. For this course book, the fork-based workflow is the default student route.

## Support Path

Use the Canvas course support channel for setup problems. Include:

- your GitHub username
- your GitHub Education status, if relevant
- the step where you got stuck
- the exact error message or screenshot
- whether you are using GitHub, Codespaces, or Canvas

Do not post passwords, tokens, private account recovery details, or billing information.

## What To Bring To Week 1

Bring:

- a working Canvas login
- a working GitHub login
- your GitHub username
- your GitHub Education status, if available
- a laptop or access plan for in-class work
- one question about health data, public data, AI, or reproducibility

## Definition Of Done

Before Week 1, make sure you have created a GitHub account, chosen a professional username, confirmed you can log in, watched the Canvas orientation video, completed the pre-course survey, and submitted the onboarding check. Week 0 is done when all of these are complete and you have either applied for GitHub Education if eligible or indicated your GitHub Education status in Canvas.

# **Week 1: Health Data, KT & Ethics**

# Health Data & Ethics

## Overview

Week 1 sets the ethical and communication foundation for the whole course. Health data science does not end when code runs or a p-value appears; it ends when evidence is translated responsibly for a real audience. This week introduces open health data, data provenance, stewardship, privacy, responsible AI use, Indigenous Data Sovereignty awareness, and Knowledge Translation (KT) as the “last mile” of analysis.

Health data means information related to health, health care, populations, conditions, services, or outcomes. In this course, we primarily work with tabular data and focus on where it came from, who it represents, and how it can be communicated responsibly.

This is an awareness-heavy week. There is no coding, no dashboard running, and no app setup. The NHANES case study appears only as a data/provenance artifact so you can practice evaluating a dataset before using it.

## Learning Objectives

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

- identify credible open health data portals;
- distinguish microdata from aggregated indicators;
- complete a Data Intake Card for a public health dataset;
- record basic provenance, licensing, access, and citation information;
- name privacy, security, and stewardship risks that can still apply to public data;

- explain why AI summaries must be checked against source documentation;
- recognize when Indigenous Data Sovereignty, OCAP, or CARE principles may be relevant;
- frame a dataset as the start of a KT product, not just as a file to analyze.

## Connection

This first week asks, “Should we use this data, and how should we describe it?” Next week asks, “How do we organize the files so another person can reproduce our work?” Later weeks return to the same NHANES case study through analysis, visualization, EDA, and dashboard design.

## Case Study Data Spine

The file paths and code below are shown only as examples of where the dataset lives in the course repository. In Week 1, you do not need to run this code. Your task is to understand the dataset’s source, documentation, and responsible-use considerations.

The **NHANES Health Equity data spine** is the recurring dataset thread for this course. In the early weeks, use it only as a documented public-health data artifact: provenance, ethics, file organization, and reproducible paths.

- Cached RDS: `examples/nhanes-equity/data/nhanes_equity_v6.rds`
- CSV snapshot: `examples/nhanes-equity/data/nhanes_equity_v6.csv`
- Case-study README: `examples/nhanes-equity/README.md`

The cached files are provided so early work can happen without internet access.

**What to do here:** read the code below as an example of documented data provenance — you are not required to run or modify it on this page (each week’s page states its own expectations).



```

1 library(tidyverse) # AI-EDIT(2026-06-23): tidyverse-default - consolidated core library() call
2
3 candidate_paths <- c(
4   "examples/nhanes-equity/data/nhanes_equity_v6.csv",
5   "../examples/nhanes-equity/data/nhanes_equity_v6.csv"
6 )
7
8 nhanes_path <- candidate_paths[file.exists(candidate_paths)][1]
9 if (is.na(nhanes_path)) {
10   stop("Could not find examples/nhanes-equity/data/nhanes_equity_v6.csv")
11 }
12
13 nhanes_spine <- read_csv(nhanes_path, show_col_types = FALSE)
14
15 dim(nhanes_spine)
16 #> [1] 105626      18
17 glimpse(nhanes_spine)
18 #> Rows: 105,626
19 #> Columns: 18
20 #> $ Cycle      <chr> "1999-2000", "1999-2000", "1999-2000", "1999-2000", "19~
21 #> $ BMI        <dbl> 14.90, 24.90, 17.63, NA, 29.10, 22.56, 29.39, 15.51, 18~
22 #> $ Weight     <dbl> 12.5, 75.4, 32.9, 13.3, 92.5, 59.2, 78.0, 40.7, 45.5, 1~
23 #> $ Height     <dbl> 91.6, 174.0, 136.6, NA, 178.3, 162.0, 162.9, 162.0, 156~
24 #> $ Waist      <dbl> 45.7, 98.0, 64.7, NA, 99.9, 81.6, 90.7, 64.1, 64.6, 108~
25 #> $ WeightMEC  <dbl> 10982.899, 28325.385, 46192.257, 10251.260, 99445.066, ~
26 #> $ Strata     <dbl> 5, 1, 7, 2, 8, 2, 4, 6, 9, 7, 1, 6, 13, 12, 11, 11, 5, ~
27 #> $ PSU        <dbl> 1, 3, 2, 1, 2, 2, 2, 1, 2, 1, 2, 2, 2, 1, 2, 1, 1, 1, 2~
28 #> $ Gender     <chr> "Female", "Male", "Female", "Male", "Male", "Female", "~
29 #> $ Race       <chr> "Non-Hispanic Black", "Non-Hispanic White", "Non-Hispan~
30 #> $ Education  <chr> NA, "College Grad", NA, NA, "College Grad", NA, "HS or ~
31 #> $ Marital    <chr> NA, NA, NA, NA, "Married/Partner", "Never Married", "Ma~
32 #> $ Age        <dbl> 2, 77, 10, 1, 49, 19, 59, 13, 11, 43, 15, 37, 70, 81, 3~
33 #> $ PIR        <dbl> 0.86, 5.00, 1.47, 0.57, 5.00, 1.21, NA, 0.53, NA, NA, 1~
34 #> $ EducationClean <chr> NA, "College Grad", NA, NA, "College Grad", NA, "HS or ~
35 #> $ MaritalClean <chr> NA, NA, NA, NA, "Married/Partner", "Never Married", "Ma~
36 #> $ IncomeGroup <chr> "Low Income (<1.3)", "High Income (>3.5)", "Middle Inco~
37 #> $ WHtR       <dbl> 0.4989083, 0.5632184, 0.4736457, NA, 0.5602916, 0.50370~

```

For Week 1, treat the NHANES case study as a provenance and stewardship example only. Do not run the app. Do not rebuild

the data. Do not upload row-level data to an AI tool.

## In-Class Activity

Use [Activity: Data Intake Card](#).

1. Choose one approved public health dataset or use the shared NHANES case-study example.
2. Find the dataset publisher, documentation, access page, and citation guidance.
3. Complete the Data Intake Card.
4. Write one KT framing sentence: who could use this dataset, and what decision or conversation could it support?
5. Name one ethics, privacy, security, or stewardship issue that should be checked before publishing results.

Student output: one completed Data Intake Card, including its KT framing sentence. Submit that card as the required, unweighted Complete/Needs Work Canvas checkpoint. It is a lightweight early-support check, not a percentage-weighted assignment and not an additional deliverable.

## Open Data Portals

Open data can accelerate public health learning when the source, meaning, and limits of the data are clear. Start with reputable portals that provide documentation, terms of use, and stable links.

Recommended starter portals:

- CDC/NCHS public-use survey data, such as [NHANES](#) or [NHIS](#);
- [WHO Global Health Observatory](#) for aggregated international indicators;
- [BC Data Catalogue](#) for provincial open data;
- [Government of Canada Open Government Portal](#) for Canadian datasets.

Each link above goes to that source's page in the Reference: Data Sources part of this book, which also covers other sources you may use later in the course.

When you land on a dataset page, look for:

- publisher or steward;
- population, geography, and time coverage;
- data level, such as microdata or aggregated indicators;
- documentation, codebook, or methodology notes;
- license or terms of use;
- citation recommendation;
- last updated date or access date.

## Provenance

Provenance means the traceable story of where data came from and how it reached your project. A dataset is not just a file-name. It has a steward, collection context, transformations, limitations, and documentation.

For every dataset, record:

- original source page;
- local file path if a copy is stored in the repository;
- date accessed;
- file format;
- variables you plan to use;
- known limitations or cautions;
- citation or attribution language.

The Data Intake Card is the course habit that makes provenance visible.

## Stewardship

Stewardship means using data in a way that respects the people, communities, and institutions represented by the data. Public data still requires care.

Ask:

- Could a group be stigmatized by a careless summary?
- Are small cells or rare combinations being exposed?
- Does the dataset include populations for whom special governance principles may apply?
- Are limitations visible enough for a non-technical audience?
- Does the planned KT product support understanding rather than overclaiming?

## Privacy And Security

In this course, Week 1 uses public, de-identified, or aggregated data only. Still, avoid treating “public” as the same as “risk-free.”

Course rules:

- Do not upload row-level health data to AI tools. Row-level data means individual records, where each row may represent one person, visit, survey response, or observation. Even if names are removed, individual-level records can still create privacy risks.
- Do not attempt re-identification.
- Do not publish small-cell claims.
- Do not copy data into public repositories unless the license and course instructions allow it.
- Do not use absolute local paths that reveal a personal machine or user name.

## Responsible AI

AI tools may help you draft plain-language summaries or identify questions to ask about documentation. They are not source documentation.

AI tools can also hallucinate: they can state plausible-sounding but wrong variable definitions or data facts with full confidence. They can also repeat biases from their training data. That

is why Week 1 asks you to verify every definition against the source documentation, not against an AI summary.

Acceptable Week 1 AI use:

- asking for a plain-language explanation of a term from a codebook;
- drafting a first pass of a dataset description;
- generating a checklist of questions to verify.

You can get useful AI help without uploading any data. Share the structure instead: column names, variable types, and a made-up example row. You can also share your code and your error messages. Never paste the real rows. The AI helps with code and wording; you run everything locally against the real data yourself.

Required checks:

- cite the official source, not the AI output;
- verify variable meanings in documentation;
- include an AI-use note if AI helped draft your card;
- revise AI language that overstates certainty or hides limitations.

## Indigenous Data Sovereignty

Indigenous Data Sovereignty refers to the rights and interests of Indigenous Peoples in data about their peoples, lands, resources, and knowledges. Two frameworks students should recognize are:

- OCAP: ownership, control, access, and possession, specific to First Nations contexts in Canada;
- CARE: collective benefit, authority to control, responsibility, and ethics.

Week 1 expectation: if a dataset includes Indigenous identifiers, communities, lands, or governance-relevant content, flag that on the Data Intake Card and review the source guidance before using or publishing analysis. For example, a variable indicating

First Nations, Métis, or Inuit identity would require extra care and review of source guidance.

## Formats And Metadata

Common formats:

- CSV: rectangular rows and columns;
- JSON: nested data, common for APIs;
- XLSX: spreadsheet files that may contain formatting and hidden assumptions;
- RDS: R-specific saved objects;
- GeoJSON or shapefiles: spatial data for maps.

Metadata explains what the file means. It may include variable definitions, units, missing-value codes, collection methods, survey design variables, and weighting instructions. A dataset without metadata is risky for analysis because column names alone rarely tell the full story.

## KT Framing

KT asks who needs the information, what decision or conversation it supports, and how the evidence should be communicated. A Week 1 KT framing sentence can be simple:

This dataset could help [audience](#) understand [health issue] so they can [decision, planning task, or conversation], with the limitation that [key caution].

Example:

The NHANES classroom dataset could help students understand how BMI summaries differ across income groups while practicing clear limitations about descriptive, unweighted survey summaries.

## What Students Leave With

By the end of Week 1, you should have:

- one completed Data Intake Card;
- one KT framing sentence;
- one ethics or stewardship caution;
- a clear understanding that the NHANES case study starts as a data artifact, not an app to run;
- readiness to place files into a reproducible project structure in Week 2.

# Activity: Data Intake Card

## Goal

Evaluate whether a health dataset is usable, ethical, documented, and appropriate for reproducible analysis. This activity is about judgment and documentation. No coding is required.

## Approved Portals

Choose one dataset or indicator page from one of these portals:

- CDC/NCHS public-use survey data, such as [NHANES](#) or [NHIS](#);
- [WHO Global Health Observatory](#);
- [BC Data Catalogue](#);
- [Government of Canada Open Government Portal](#).

Each link goes to that source's page in the Reference: Data Sources part of this book.

The NHANES Health Equity data spine may be used as the shared classroom example.

## Student Template

Copy this template into your notes.



```

1 dataset_name: ""
2 publisher_org: ""
3 permalink_url: ""
4 date_accessed: "YYYY-MM-DD"
5 license_terms: ""
6 geography: ""
7 time_coverage: ""
8 data_level: "" # microdata / aggregated indicator / unknown
9 file_format: "" # CSV / RDS / XLSX / JSON / other
10 metadata_or_codebook: ""
11 variables_of_interest:
12   - name: ""
13     definition: ""
14     unit: ""
15     caveat: ""
16 known_limitations:
17   - ""
18 privacy_security_notes: ""
19 stewardship_notes: ""
20 indigenous_data_relevance: "no / yes / unknown"
21 kt_framing_sentence: ""
22 how_to_cite: ""
23 ai_use_note: ""

```

Use **unknown** if you cannot confidently answer a field after checking the source documentation. Do not guess. Briefly state what you checked.

## Completed Example

```

1 dataset_name: "NHANES Health Equity classroom dataset"
2 publisher_org: "CDC / National Center for Health Statistics"
3 permalink_url: "https://wwwn.cdc.gov/nchs/nhanes/"
4 date_accessed: "YYYY-MM-DD"
5 license_terms: "Public-use NHANES files; cite CDC/NCHS documentation"
6 geography: "United States"
7 time_coverage: "Multiple NHANES cycles represented in the prepared class file"
8 data_level: "Microdata-derived classroom dataset"

```

```

9 file_format: "CSV and RDS cached in examples/nhanes-equity/data/"
10 metadata_or_codebook: "CDC/NCHS NHANES documentation and case-study README"
11 variables_of_interest:
12   - name: "BMI"
13     definition: "Body mass index from NHANES body measures data"
14     unit: "kg/m^2"
15     caveat: "Use descriptively unless survey design and weights are handled correctly"
16 known_limitations:
17   - "Prepared class dataset is for classroom analysis and should not be treated as a causal analysis"
18   - "Population inference requires attention to NHANES survey design and weights"
19 privacy_security_notes: "Use public-use data responsibly, avoid small-cell claims, and do not use for commercial purposes"
20 stewardship_notes: "State limitations plainly and avoid stigmatizing group comparisons"
21 indigenous_data_relevance: "unknown in the prepared class example; check source documentation for more information"
22 kt_framing_sentence: "This dataset can help students practice describing health equity patterns in data"
23 how_to_cite: "Cite CDC/NCHS NHANES documentation and the course case-study README"
24 ai_use_note: "AI may help draft plain-language wording, but source documentation must be checked for accuracy"

```

For `how_to_cite`, follow the course citation pattern: source organization, dataset name and cycle, publisher, URL, and access date. For example:

Centers for Disease Control and Prevention (CDC), National Center for Health Statistics. National Health and Nutrition Examination Survey (NHANES) 2017-2018 Documentation. Hyattsville, MD: U.S. Department of Health and Human Services. <https://wwwn.cdc.gov/nchs/nhanes/> (accessed 2026-06-11).

## Completion Checklist

To be marked complete, your card must include:

- stable source link or access date;
- publisher or steward;
- license or terms of use;
- geography and time coverage;
- data level and file format;
- documentation or codebook location;

- at least one documented variable definition;
- a 1-2 sentence note on how an AI tool could misrepresent or bias that variable, such as wrong category labels, outdated coding, or overgeneralization;
- one limitation or stewardship risk;
- one privacy or security note;
- Indigenous Data Sovereignty relevance marked as no, yes, or unknown;
- KT framing sentence;
- AI-use note.

## Scoring Guide

| Rating     | Evidence                                                                                                                                                                          |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Complete   | Source, publisher, terms, metadata, variable definition, limitation, stewardship note, KT framing, and AI-use note are present and grounded in documentation.                     |
| Needs work | Source documentation is missing, license terms are unclear, variable definitions are guessed, risks are absent, or AI-written text is not checked against official documentation. |

## Submission

Submit the completed card in Canvas under the Week 1 module as a **required, unweighted Complete/Needs Work checkpoint**. It carries no percentage weight and has no granular rubric. Its purpose is to identify setup, access, or workflow trouble early enough for the TA or instructor to help before later assignments build on it.

Submit only the completed card; no screenshot, reflection, or additional setup evidence is required. Starting in Week 2, keep a copy in your course repository as:

```
1 data-intake-card.md
```

## Reproducibility Check

Before submitting, check:

- the link opens or the access date is recorded;
- the publisher and license are named;
- variable definitions come from documentation, not guesses;
- file format and data level are named;
- the AI-use note says what AI helped with and how the text was checked.

## **Week 2: Modern Workflows**

# Modern Workflows

## Overview

**Summary:** This week you set up the computational environment you will use for the rest of the course. By the end, the technology should fade into the background so you can focus on health data science, not installation errors. We adopt a “**Zero to AI Co-Pilot**” approach: you do not need to install R, Python, or Git on your personal laptop. Everything runs in a cloud-native environment called GitHub Codespaces.

You can complete the course using VS Code in your web browser through GitHub Codespaces. You do not need to install the desktop version of VS Code unless you choose to.

### Key Terms This Week

- **Repository (repo):** your project folder stored on GitHub — the “cloud drive.”
- **Codespace:** a cloud computer that opens in your browser — the “public computer.”
- **IDE:** the editor you work in. Here it is VS Code, running inside the Codespace.
- **Quarto (.qmd):** a document format that mixes text and runnable code; rendering it produces a finished HTML report.

Each term gets a full section further down this page.

### Learning Objectives:

- Understand the “public computer vs cloud drive” analogy for cloud computing.

- Create a GitHub account, fork the course template repository, and work from your personal fork.
- Launch a Codespace and identify the three key areas of the VS Code interface.
- Distinguish between an IDE, a notebook, and a script, and explain why this course uses Quarto notebooks.
- Explain the concept of literate programming and why it matters for reproducible health research.
- Edit and render a Quarto (.qmd) document: YAML header, Markdown text, and code chunks.
- Maintain a reproducible project structure using the **data/** and **output/** folder convention.
- Stage, commit, and sync changes using the VS Code Source Control panel.
- Stop your Codespace to conserve your free monthly hours.

## Connection

Week 1 focused on whether a dataset is trustworthy enough to use. This week focuses on where that dataset lives in a project and how relative paths make the same work portable across machines. Next week you will start inspecting the data with R.

## Case Study Data Spine

**What to do with this code:** Read only — you do not need to run, modify, or submit it this week. The code block below shows how the case-study data is loaded with a relative path. You will start running code like this yourself in Week 3.

The **NHANES Health Equity data spine** is the recurring dataset thread for this course. In the early weeks, use it only as a documented public-health data artifact: provenance, ethics, file organization, and reproducible paths.

- Cached RDS: `examples/nhanes-equity/data/nhanes_equity_v6.rds`
- CSV snapshot: `examples/nhanes-equity/data/nhanes_equity_v6.csv`
- Case-study README: `examples/nhanes-equity/README.md`

The cached files are provided so early work can happen without internet access.

**What to do here:** read the code below as an example of documented data provenance — you are not required to run or modify it on this page (each week's page states its own expectations).

```
1 library(tidyverse) # AI-EDIT(2026-06-23): tidyverse-default - consolidated core library() cal
2
3 candidate_paths <- c(
4   "examples/nhanes-equity/data/nhanes_equity_v6.csv",
5   "../examples/nhanes-equity/data/nhanes_equity_v6.csv"
6 )
7
8 nhanes_path <- candidate_paths[file.exists(candidate_paths)][1]
9 if (is.na(nhanes_path)) {
10   stop("Could not find examples/nhanes-equity/data/nhanes_equity_v6.csv")
11 }
12
13 nhanes_spine <- read_csv(nhanes_path, show_col_types = FALSE)
14
15 dim(nhanes_spine)
16 #> [1] 105626      18
17 glimpse(nhanes_spine)
18 #> Rows: 105,626
19 #> Columns: 18
20 #> $ Cycle      <chr> "1999-2000", "1999-2000", "1999-2000", "1999-2000", "19~
21 #> $ BMI        <dbl> 14.90, 24.90, 17.63, NA, 29.10, 22.56, 29.39, 15.51, 18~
22 #> $ Weight     <dbl> 12.5, 75.4, 32.9, 13.3, 92.5, 59.2, 78.0, 40.7, 45.5, 1~
23 #> $ Height     <dbl> 91.6, 174.0, 136.6, NA, 178.3, 162.0, 162.9, 162.0, 156~
24 #> $ Waist      <dbl> 45.7, 98.0, 64.7, NA, 99.9, 81.6, 90.7, 64.1, 64.6, 108~
25 #> $ WeightMEC  <dbl> 10982.899, 28325.385, 46192.257, 10251.260, 99445.066, ~
26 #> $ Strata     <dbl> 5, 1, 7, 2, 8, 2, 4, 6, 9, 7, 1, 6, 13, 12, 11, 11, 5, ~
27 #> $ PSU        <dbl> 1, 3, 2, 1, 2, 2, 2, 1, 2, 1, 2, 2, 2, 1, 2, 1, 1, 1, 2~
28 #> $ Gender     <chr> "Female", "Male", "Female", "Male", "Male", "Female", "~
29 #> $ Race       <chr> "Non-Hispanic Black", "Non-Hispanic White", "Non-Hispan~
30 #> $ Education  <chr> NA, "College Grad", NA, NA, "College Grad", NA, "HS or ~
31 #> $ Marital    <chr> NA, NA, NA, NA, "Married/Partner", "Never Married", "Ma~
32 #> $ Age        <dbl> 2, 77, 10, 1, 49, 19, 59, 13, 11, 43, 15, 37, 70, 81, 3~
33 #> $ PIR        <dbl> 0.86, 5.00, 1.47, 0.57, 5.00, 1.21, NA, 0.53, NA, NA, 1~
```



```

34 #> $ EducationClean <chr> NA, "College Grad", NA, NA, "College Grad", NA, "HS or ~
35 #> $ MaritalClean <chr> NA, NA, NA, NA, "Married/Partner", "Never Married", "Ma~
36 #> $ IncomeGroup <chr> "Low Income (<1.3)", "High Income (>3.5)", "Middle Inco~
37 #> $ WHtR <dbl> 0.4989083, 0.5632184, 0.4736457, NA, 0.5602916, 0.50370~

```

## In-Class Activity

- Locate the case-study data, README, and snippet folders in the Explorer.
- Explain why the data example uses relative paths instead of machine-specific paths.
- Check that the weekly page includes the shared snippet rather than duplicating instructions.
- Render the book from the repository root and check that this page updates in the rendered output.

Student output: a rendered Quarto page and one commit showing a small, reproducible edit. Assignment link: [Assignment 1](#).

---

## workflows1.qmd The Analogy: Public Computer and Cloud Drive

*Deeper Dive: [The Analogy: Public Computer vs Cloud Drive](#).*

To understand how we work in this course, forget about “servers” and “clients.” Imagine you are working in a university computer lab.

### The Repository (Your Cloud Drive)

Think of **GitHub** as your personal **Google Drive** or **Dropbox**.

- It is where your files live permanently.

- If your laptop crashes, the files are still here.
- **Key concept:** This is your “Master Copy.”

## The Codespace (The Public Computer)

Think of a **Codespace** as a **public computer in the library**.

- It is a temporary machine you borrow to do your work.
- It comes pre-installed with all the software you need (R, Python, Quarto), so you don’t have to install anything on your own laptop.
- **Crucial difference:** Saved changes remain in that Codespace when it stops, but they are not in your GitHub repository until you commit and sync them. A Codespace can later be deleted, so GitHub—not the Codespace—is the safe home for finished work.

## The Workflow

1. **Log in:** Launch your Codespace (open the public computer).
2. **Work:** Edit files using the computer’s software.
3. **Save to the cloud:** Commit and push your work back to your Cloud Drive (GitHub).
4. **Log off:** Stop the Codespace to save your free hours.

This cycle — launch, work, commit, stop — is the rhythm of every week in this course.

---

## workflows2.qmd Getting Started: GitHub Account and Fork

*Deeper Dive:* [Tour of the Repository](#).

## Step 1: Check Your GitHub Account

You should already have a GitHub account from [Week 0: Asynchronous Onboarding](#), which walks through account creation step by step. If you skipped Week 0, go back and complete the account setup there first.

If login goes wrong:

- **Forgot which email you used?** Try your UBC email first — that is what Week 0 recommended.
- **Forgot your password?** Use **Forgot password?** on the GitHub sign-in page; the reset email arrives within a few minutes.
- **Account exists but you cannot access it?** Ask for help in class rather than creating a second account — two accounts cause confusion later.

### Tip

**Portfolio thinking:** Your GitHub profile will become part of your professional presence. Pick a username you would be comfortable putting on a CV (e.g., `jchen-epi` rather than `xXcoolcoder99Xx`).

## Step 2: Fork the Course Template

1. Open the course starter repository: <https://github.com/ehsanx/HDSx-workspace> (also linked from Canvas).
2. Click **Fork** in the upper-right corner of the GitHub page.
3. Create the fork under your own GitHub account.
4. Keep the default repository name unless Canvas gives a specific naming rule.
5. After the fork is created, look at the web address at the top of your browser. It should include your GitHub username. This confirms that you are working in your own copy, not the original course repository.

### Step 3: Verify Your Repository

Navigate to your forked repository. You should see:

- A file list including `practice_report.qmd`, `README.md`, and a `data/` folder.
- A formatted README below the file list explaining the project.

If you see these in your fork, you are ready to launch your Codespace.

If you do not see these, stop and ask for help before continuing. You may be in the wrong repository, the wrong branch, or the starter files may not have loaded correctly.

---

## workflows3.qmd Launching Your Codespace

*Deeper Dive:* [Opening Your First Codespace](#) and [Reopening a Codespace](#).

1. On your repository page, click the green `<>` **Code** button.
2. Select the **Codespaces** tab.
3. Click **Create codespace on main**.

The first launch takes 2–3 minutes while the environment builds. Subsequent launches are faster.

#### **i** Note

**Reopening an existing Codespace:** After you stop or close your Codespace, do **not** create a new one each time. Go to [github.com/codespaces](https://github.com/codespaces), find your existing Codespace, and click its name to restart it. Creating multiple Codespaces wastes your hours and causes confusion.

## workflows4.qmd Tour of VS Code: The “Big Three” Zones

*Deeper Dive: [VS Code Orientation](#).*

The VS Code interface can look intimidating — it is designed for professional software engineers. You only need to know three areas. Safely ignore 90% of the buttons.

### 1. The Explorer (Left Sidebar)

Your **file cabinet**. It shows folders and files in your project. You will use this to open `.qmd` files, navigate the `data/` folder, and create new files.

### 2. The Editor (Center)

Your **work area**. This is where you read and write code and text. You can have multiple files open in tabs.

### 3. The Terminal (Bottom Panel)

The **engine room** where R and Python actually run. You will not type commands here often — most of your work happens in the Editor — but this is where output and error messages appear.

#### Tip

**Lost a panel?** Use the top menu: **View** → **Explorer** or **View** → **Terminal** to bring them back. You can also press `Ctrl+`` (backtick) to toggle the Terminal.

## workflows5.qmd **Concepts: IDE, Notebook, Script**

The syllabus mentions three ways to write code. Understanding the difference helps you choose the right tool throughout the course.

### **IDE (Integrated Development Environment)**

VS Code is an **IDE** — a text editor with superpowers: file browsing, syntax highlighting, built-in terminal, extensions, and version control. It is the container that holds everything else.

### **Script**

A plain text file containing only code (e.g., `.R` or `.py`). Runs top-to-bottom. Best for reusable utilities and automation. You will write your first script in Week 5.

### **Notebook**

A document that **interleaves code and narrative**. You run code in chunks and see results inline. Examples include Jupyter notebooks (`.ipynb`, Week 5) and Quarto documents (`.qmd`, which you use starting today).

**This course primarily uses Quarto notebooks** because they combine the strengths of notebooks (interactivity, narrative) with the ability to produce publication-quality reports, slides, and websites — all from a single file.

---

## workflows6.qmd Concepts: Literate Programming

### The Big Idea

Traditional workflows separate code from writing: you run an analysis in one program, copy results into a Word document, and hope nothing changes. **Literate programming** weaves code, results, and interpretation into a single document. When the data changes, you re-render and the entire report updates automatically.

This is not just convenient — it is a **scientific obligation**. In health research, reviewers and regulators need to see exactly how you went from raw data to conclusions. A literate document is a transparent, auditable record of your reasoning.

### How Quarto Implements This

A `.qmd` file has three ingredients:

1. **YAML header** (metadata at the top, between `---` lines)
2. **Markdown text** (human-readable prose)
3. **Code chunks** (executable R or Python code in fenced blocks)

When you **Render**, Quarto runs every code chunk, captures the output (tables, figures, numbers), and weaves everything into a finished HTML, PDF, or Word document.

You will use literate programming in every subsequent week of this course: your assignments, your EDA (Week 7), your reports (Week 10), and your final portfolio are all Quarto documents.

## workflows7.qmd Project Structure: Files and Folders

*Deeper Dive:* [Files vs Folders](#).

A reproducible project needs a consistent folder structure. This course uses the following convention:

```
my-project/  
  data/          ← Raw data. Never edit files here directly.  
  output/        ← Cleaned data, figures, and tables generated by your code.  
  practice_report.qmd ← Your Quarto notebook.  
  README.md      ← Explains what this project is.  
  .gitignore     ← Tells Git which files to ignore (more in Week 4).
```

### Rules

- **data/ is read-only.** Import from it; never save into it.
- **output/ is generated.** Your code writes cleaned data and figures here. In Week 3, you will export `output/summary_table.csv`.
- **Use relative paths.** Always refer to files relative to the project root (e.g., `"data/patients.csv"`), never absolute paths (e.g., `"/home/user/Documents/patients.csv"`). This ensures your code runs on any machine.

**Activity:** In the Explorer pane, click the **New Folder** icon and create the `output/` folder if it does not already exist.

---

## workflows8.qmd Tutorial: Your First Quarto Document

*Deeper Dive:* [The Magic Moment: Render Quarto](#).



## Step 1: Open the Starter File

In the Explorer, click `practice_report.qmd`. It lives in the top level (root) of your forked workspace repository, alongside `README.md` and the `data/` folder. It opens in the Editor.

## Step 2: Understand the YAML Header

At the very top, between `---` lines, you will see something like:

```
1 ---
2 title: "My First Report"
3 author: "Your Name"
4 format:
5   html:
6     embed-resources: true
7 ---
```

Change `author:` to your actual name. The YAML header controls metadata and output format. The `embed-resources: true` line bundles every figure and style into a single self-contained HTML file, so the report still shows your plots when you download it or open it on another computer.

## Step 3: Read the Markdown

Below the YAML, you will see plain text with formatting:

- `# Heading` creates a section title.
- `**bold**` makes text **bold**.
- `- item` creates a bullet point.

This is **Markdown** — a lightweight way to format text without a word processor.

## Step 4: Read the Code Chunks

Look for grey blocks that start with ````${r}` and end with `````. These are **code chunks**. Everything inside them is R code that will execute when you render.

```
1  ```${r}
2  # This is a code chunk
3  2 + 2
4  ```
```

## Step 5: Edit a Code Chunk

Find a code chunk in the starter file. Make a small change — for example, change a number or add a comment. This is your first edit.

## Step 6: Render

**Before you render:** save the file with **Ctrl+S** (**Cmd+S** on macOS). Rendering runs the code chunks from the start and captures their results, so you do not need to run every chunk first. Running chunks interactively is still useful for finding errors before a full render.

To render, open the **Command Palette** — press **Ctrl+Shift+P** (Windows/Linux) or **Cmd+Shift+P** (macOS), or use **View → Command Palette** — and run **Quarto: Render Document**. If you are asked which output format to produce, choose **HTML**. The detailed steps and an always-works terminal fallback are below.

### 💡 How to render: the course default

1. Open the `.qmd` file you want to render, then open the **Command Palette** (**View → Command Palette**, or press **Ctrl+Shift+P** on Windows/Linux / **Cmd+Shift+P** on Mac) and run **Quarto: Render Document**. The Quarto extension renders the file and opens a preview pane.

2. **Always-works fallback (terminal):** type the render command with the file's path as shown in the Explorer, for example:

```
1 quarto render practice_report.qmd
```

Always name the file — a bare `quarto render` rebuilds the *whole* project and takes much longer.

3. If you see a **Render** (or **Preview**) button in the editor toolbar, it does the same thing as step 1.

**Where did the output go?** Watch the render log for the **Output created:** line — it names the exact `.html` file created. By default the file appears next to your `.qmd`; in projects that set an output directory (like this book's `docs/` folder), it appears there instead. If no preview opened automatically, find that `.html` file in the Explorer, right-click it, and choose **Download** to open it in your browser (or **Open Preview** if available).

A preview pane appears showing your formatted report with code output embedded.

#### **i** Where is my HTML file?

In your personal workspace, `practice_report.html` is saved beside `practice_report.qmd` at the repository root. Watch the render log for the **Output created:** line, then right-click the HTML file and choose **Open Preview** or **Download**. The course book uses a separate `docs/` output directory; your personal workspace does not.

#### **i** Seeing a TeX warning?

If you see a warning that no TeX installation was found, you can ignore it as long as your HTML file was created successfully. TinyTeX is only needed for PDF output — do not install anything.



Tip

**What just happened:** Quarto ran every code chunk, captured the results, combined them with your Markdown text, and produced an HTML document. This is literate programming in action.

---

## workflows9.qmd Tutorial: Stage, Commit, and Sync

*Deeper Dive:* [Commit Your Work](#).

Your work currently exists only on the public computer (Codespace). To save it permanently to your Cloud Drive (GitHub), you need to **commit**.

Think of this as a three-step save process:

1. **Stage** = choose which changed files to include.
2. **Commit** = save a snapshot with a message.
3. **Sync** = upload that snapshot to GitHub so it is safely stored online.

### Step 1: Open Source Control

Click the **Source Control** icon on the left sidebar (it looks like a branching graph). You will see a list of files that have changed since your last commit.

### Step 2: Stage Your Changes

Hover over a changed file name. Click the **+** icon that appears. This **stages** the file — it tells Git “I want to include this change in my next snapshot.”

 Note

**Staging vs committing:** Staging selects *which* changes to include. Committing takes the snapshot. Think of staging as putting items on the conveyor belt; committing is pressing the “scan” button at checkout.

### Step 3: Write a Commit Message

In the text box at the top of the Source Control panel, type a short, descriptive message:

Updated author name and edited first code chunk  
in practice report

A good commit message answers: *what did I change and why?*

### Step 4: Commit

Click the **Commit** button (checkmark). Your changes are now saved as a snapshot in your local history.

### Step 5: Sync with GitHub

Click **Sync Changes** (in the Source Control panel or the status bar at the bottom). This pushes your commit to GitHub — your Cloud Drive.

**Verify:** Go to your repository on [github.com](https://github.com) and check that your changes appear. You should see your commit message next to the updated file.

 Warning

**If you skip the Sync step**, your commit exists only on the Codespace. If the Codespace is deleted, the commit goes with it. Always Sync.

## workflows10.qmd Stopping Your Codespace (Saving Credits)

*Deeper Dive:* [Stop Your Codespace](#).

GitHub Free personal accounts currently include 120 core-hours per month. On the course's default 2-core machine, that is about 60 hours of active runtime. **If you leave the browser tab open, the meter keeps running** until the Codespace reaches its idle timeout, even if you are not actively working.

### The Right Way to Stop

1. Go to [github.com/codespaces](https://github.com/codespaces).
2. Find your active Codespace.
3. Click the **three dots (...)** → **Stop codespace**.

### What Happens If You Just Close the Tab?

The Codespace runs for another 30 minutes before it times out automatically. That is 30 minutes of wasted credits every time.

#### Warning

**Policy:** It is your responsibility to manage your compute hours. If you run out before the month resets, you will need to wait or pay out of pocket. Get in the habit of stopping your Codespace at the end of every session.

---

## workflows11.qmd Troubleshooting

*Deeper Dive:* [Troubleshooting: Rebuild Container](#).

Sometimes things break. Here is a graduated approach — try the simplest fix first.

## Level 1: Reload the Window

Press **F1** (or **Ctrl+Shift+P** / **Cmd+Shift+P**) to open the **Command Palette**. Type **Reload Window** and select it. This restarts VS Code without rebuilding the entire environment.

## Level 2: Restart the R or Python Session

If R or Python is frozen, open the Command Palette → type **R: Restart R Session** (or the equivalent for Python). This clears the session without touching your files.

## Level 3: Rebuild the Container (Nuclear Option)

If nothing else works:

1. Open the **Command Palette**.
2. Type **Rebuild Container**.
3. Select **Codespaces: Rebuild Container**.

This shuts down and rebuilds the entire environment from scratch. It solves nearly all technical glitches but takes a few minutes.

### Warning

**Before rebuilding:** Make sure you have committed and synced your work. Uncommitted changes may survive a rebuild, but there is no guarantee.

---

## workflows12.qmd Knowledge Check

1. In the analogy, what is the “Cloud Drive” and what is the “Public Computer”?
2. Why don't you need to install R or Python on your laptop for this course?

3. Name the “Big Three” zones of the VS Code interface and what each is for.
  4. What is the difference between an IDE, a notebook, and a script?
  5. What is literate programming, and why does it matter for health research?
  6. What are the three ingredients of a `.qmd` file?
  7. What is the difference between **staging** and **committing**?
  8. Why must you click **Sync Changes** after committing?
  9. How do you properly stop a Codespace to save your free hours?
- 

## workflows13.qmd Assignment 1: Your First Quarto Report

**Due Monday after the Week 2 class at 4 PM.** Submit via Canvas.

### Task

Edit and render the supplied root-level `practice_report.qmd`, demonstrating that you can navigate your personal workspace Codespace, edit Quarto, and commit your work. This is the same authoritative workflow as the [Assignment 1 brief](#); there is no separate nested A1 submission.

### Step-by-Step Workflow

1. **Launch** your Codespace from your forked repository.
2. **Edit the YAML:** Change `author:` to your name.
3. **Add a Markdown section:** Below the existing content, add a new heading (`## My Notes`) and write 2–3 sentences about what you learned this week. Use at least one Markdown feature (bold, italics, or a bullet list).



4. **Edit a code chunk:** Modify an existing code chunk or add a new one. For example:

```
1 ```{r}
2 # My first R calculation
3 2 + 2
4 ```
```

5. **Render** the document. Check that the output looks correct.
6. **Commit and Sync:**
- Open Source Control.
  - Stage your changed files (+).
  - Write a commit message: e.g., "Complete Week 2 assignment: edit YAML, add notes, render report".
  - Click **Commit**, then **Sync Changes**.
7. **Stop** your Codespace.
8. **Verify** on github.com that your commit appears in the repository.
9. **Submit** your personal workspace repository link on Canvas. The synced commit must include the root-level `practice_report.qmd` and `practice_report.html`; no extra reflection or commit-evidence file is required.

---

## workflows14.qmd Quick Reference

| Action           | How                                                                                                |
|------------------|----------------------------------------------------------------------------------------------------|
| Launch Codespace | Repository → green <><br><b>Code</b> → Codespaces →<br><b>Create codespace on main</b>             |
| Reopen Codespace | <a href="https://github.com/codespaces">github.com/codespaces</a> →<br>click Codespace name        |
| Stop Codespace   | <a href="https://github.com/codespaces">github.com/codespaces</a> → ...<br>→ <b>Stop codespace</b> |

| Action            | How                                                                                                                 |
|-------------------|---------------------------------------------------------------------------------------------------------------------|
| Open Explorer     | <b>View</b> → <b>Explorer</b> or click file icon (left sidebar)                                                     |
| Open Terminal     | <b>View</b> → <b>Terminal</b> or <b>Ctrl+`</b>                                                                      |
| Render Quarto     | Command Palette (F1) → <b>Quarto: Render Document</b> , or terminal: <code>quarto render practice_report.qmd</code> |
| Stage a file      | Source Control → hover file → click +                                                                               |
| Commit            | Type message → click                                                                                                |
| Sync to GitHub    | Click <b>Sync Changes</b>                                                                                           |
| Reload window     | Command Palette (F1) → <b>Reload Window</b>                                                                         |
| Rebuild container | Command Palette (F1) → <b>Codespaces: Rebuild Container</b>                                                         |

# The Analogy: Public Computer vs Cloud Drive

To understand the workflow, imagine working in a university computer lab.

## Repository (Cloud Drive)

**Important Distinction:** While GitHub is your “Cloud Drive,” it does **NOT** work like Google Drive.

- **Google Drive:** Auto-saves every keystroke.
- **GitHub:** Works like **Email**. You can write a draft (save in Codespace), but the cloud never sees it until you hit “Send” (Commit & Push).
- Stores your files permanently

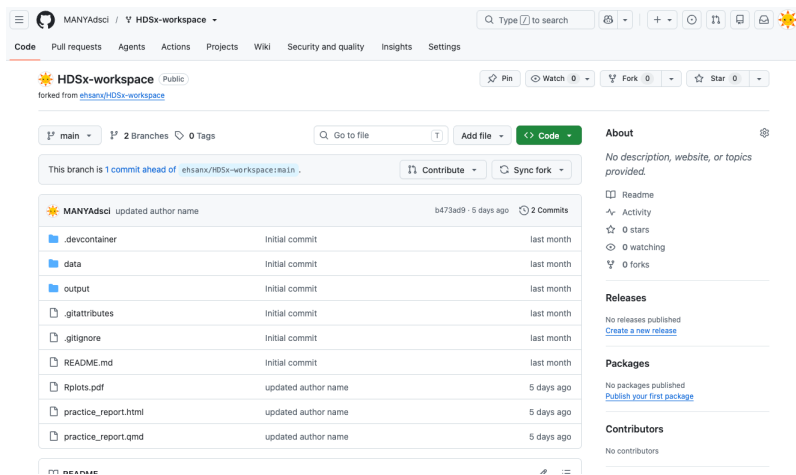


Figure 1: The code will be stored and accessible on GitHub as long as the company exists. Given that it's owned by a large corporation like Microsoft, it will likely last for a long time.

- Safe if your computer crashes
- This is your **master copy**

## Codespace (Public Computer)

Codespaces is like a library computer:

- **Temporary machine** It is a disposable environment. If it breaks, we just delete it and get a new one.

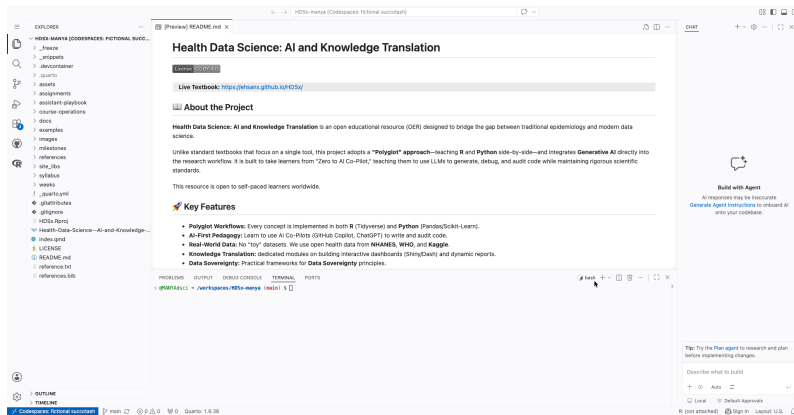


Figure 2: Can be accessed by clicking the <> Code button, navigating to the Codespaces tab, and then starting up a Codespace which contains all the file in the Repo.

- **Pre-installed software** It comes with R, Python, and Quarto installed, so you don't have to install anything on your own laptop.
- **The “Two-Save” Rule (Crucial)** Because this is a temporary computer, there are two steps to saving:
  1. **Ctrl+S (Save):** This saves the file inside the current Codespace. The change is still there when that Codespace stops and restarts, but it is not yet in your GitHub repository.
  2. **Commit & Sync:** This sends your work back to the Repository (Cloud Drive). It protects the work if the Codespace is later deleted.

**Warning:** Closing the tab does not erase saved changes, but it also does not sync them to GitHub. Reopen the same Codespace to recover unsynced saved work, then commit and sync it as soon as possible.

## Check for understanding

Answer:

- Where are your permanent files stored?

- What happens if you press `Ctrl+S` but never commit?

# Tour of the Repository

## Step-by-step

1. Open your forked course repository on GitHub

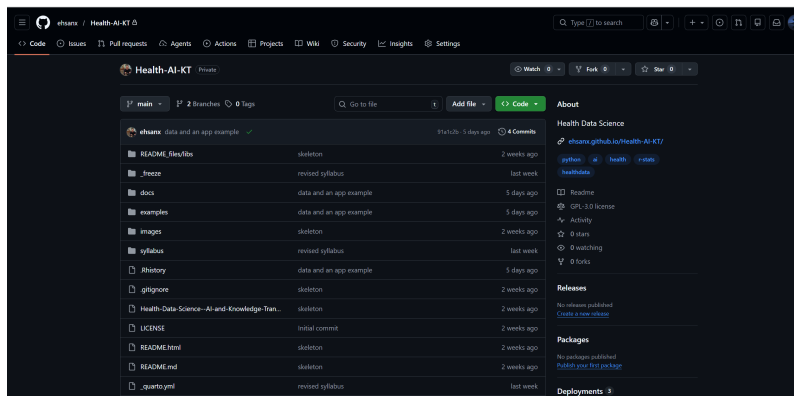
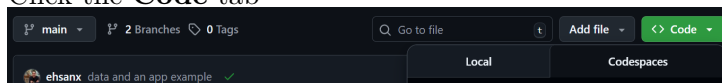


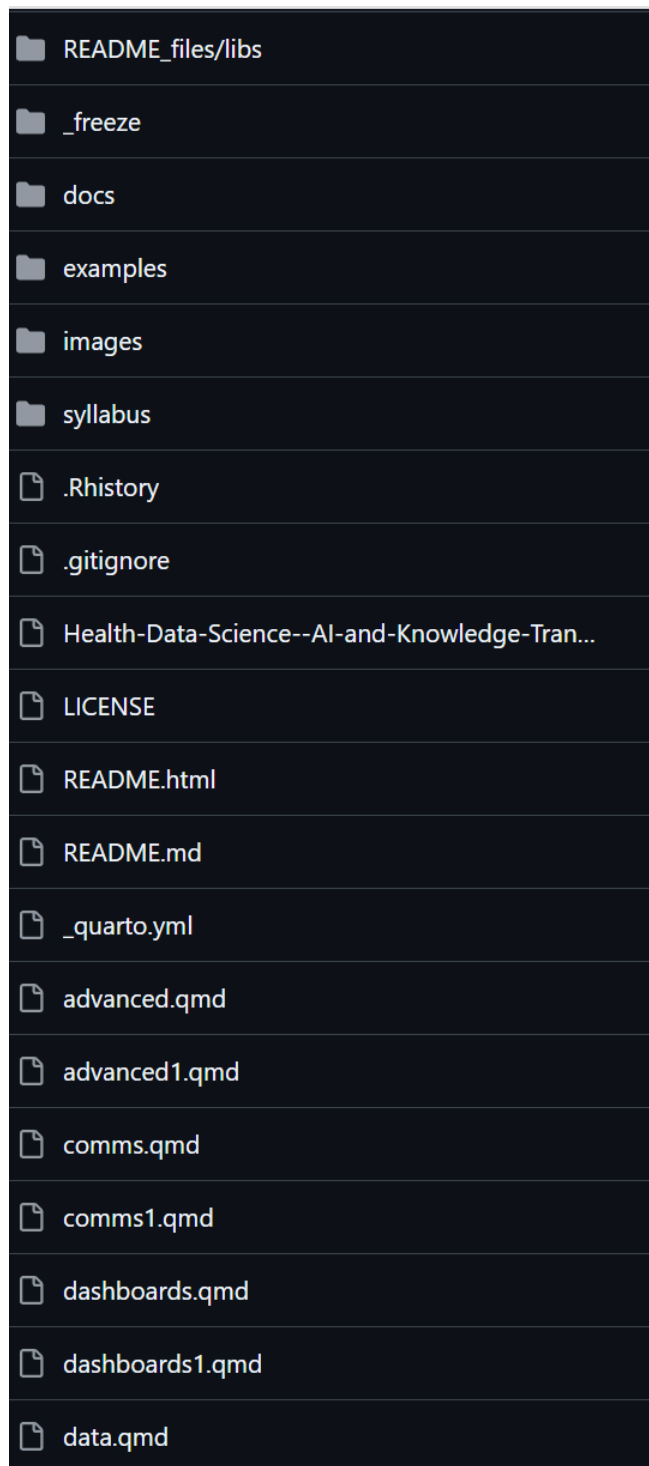
Figure 1: Main github page

2. Click the **Code** tab



- **Why?** GitHub has many tabs (Issues, Actions, Settings). The **Code** tab is your “Reset” button. If you ever get lost in a menu, click this to go back to your files.

3. Look at the file list





- This shows the **current state** of the project in your fork.
- **Crucial:** If a file is listed here, it is safe. If it is *not* here, it only exists on your temporary “Desk” (Codespace) and is at risk of being lost.

#### 4. Scroll to read the README

- The `README.md` file is displayed automatically at the bottom. Think of it as the “Cover Sheet” or “Instruction Manual” for your project.

## Task

Find this file:

`practice_report.qmd`

**Verification:** If you can see `practice_report.qmd` in this list, it means the file is safely stored in the cloud (The Vault).

If you can see it in your fork, it is safely stored in the cloud.

You should see `practice_report.qmd`, `README.md`, and a `data/` folder. If you do not see these, stop and ask for help before continuing. You may be in the wrong repository, the wrong branch, or the starter files may not have loaded correctly.

# Opening Your First Codespace

This process is like walking into the library and turning on a computer. Start from your forked repository, then create a fresh, temporary environment in the cloud.

## Steps

1. Click the green **Code** button
  - This button is usually for downloading files, but we are using it to launch a computer.

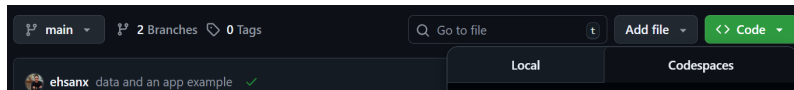


Figure 1: Code button

2. Click the **Codespaces** tab
  - **Crucial Choice:** You will see two tabs: “Local” and “Codespaces.”
  - **Local:** Means downloading to your own messy laptop (Hard mode).
  - **Codespaces:** Means using the pre-installed cloud computer (Easy mode). Always choose this.

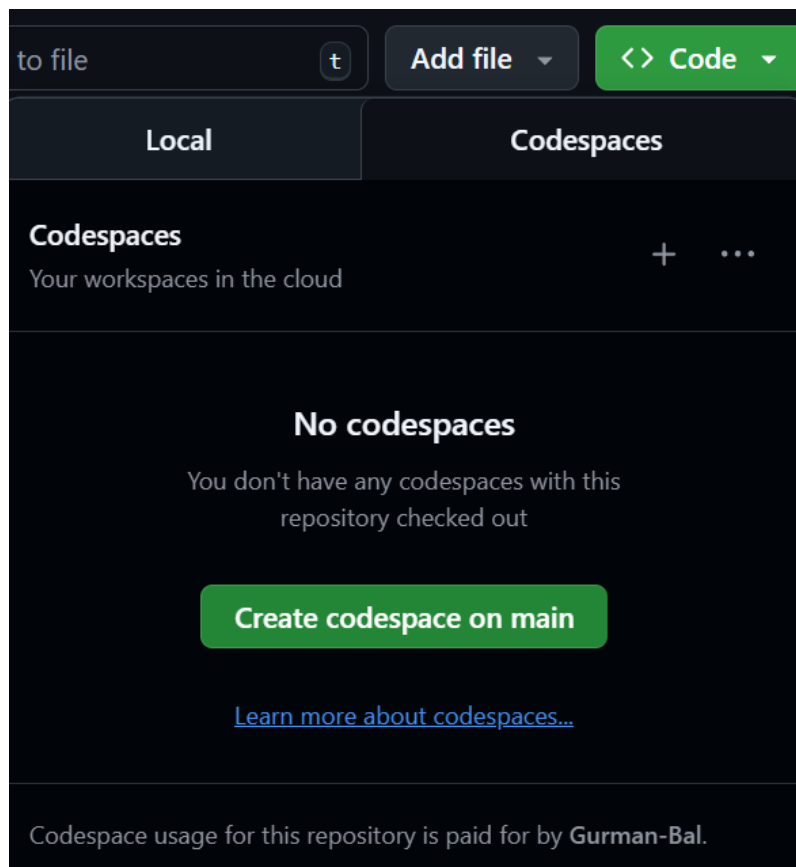


Figure 2: Codespaces tab

3. Click **Create Codespace on main**

- This tells GitHub to build a new computer for you from scratch.

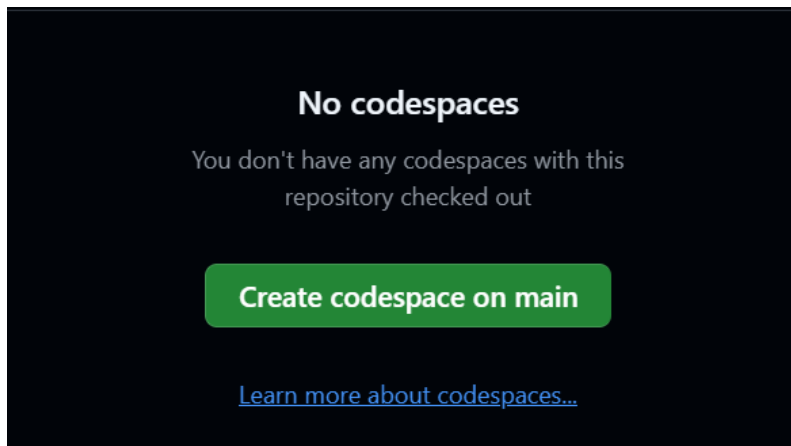


Figure 3: Create Codespace

## The “Boot Up” Phase

Wait 2–3 minutes.

- **What is happening?** GitHub is silently installing R, Python, Quarto, and all the course libraries for you.
- **The Analogy:** Think of this as the “ventilation system” starting up in a lab. It takes a moment to get the environment ready.

**Success:** You should see **VS Code** (a dark-themed code editor) appear in your browser window.

### **i** The TOOLCHAIN SELF-TEST

The setup runs **automatically** — you do not need to start anything. Near the end of the build, a block titled **TOOLCHAIN SELF-TEST** prints in the terminal. It lists version numbers for R, Quarto, and Python, then runs `quarto check`.

- **If you do not see it (or scrolled past it):** open the Command Palette (F1) and run **Codespaces: View Creation Log** to see the full build output.
- **If the self-test shows errors:** do not try to install

software yourself. Rebuild the environment instead — see [Troubleshooting: Rebuild Container](#) — and ask for help if it still fails.

# Reopening a Codespace

**The Rule:** Do not create a new Codespace each time you log in.

- **Why?** If you click the green “Create” button again, GitHub builds a *second* computer.
- **The Risk:** If you left uncommitted work (Drafts) on “Computer A” and you start “Computer B,” your work will appear lost. It isn’t lost: it’s just on the other computer.

To resume exactly where you left off:

## Steps

1. Go to the **Codespace Dashboard**: <https://github.com/codespaces>
  - Think of this as the “Front Desk” where all your active computers are listed.
2. Find your Codespace
  - Look for the box labeled with your course repository name.
  - **Note:** GitHub gives each Codespace a funny random name (e.g., *literate-computing-machine* or *glorious-couscous*). This is the name of your rented desk.

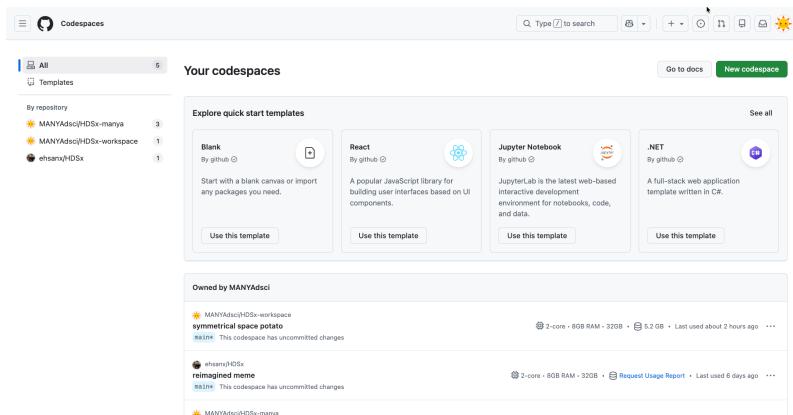


Figure 1: GitHub Codespaces

### 3. Click the **Name** of the Codespace

- Do **not** click the “New codespace” button in the corner. Click the text link of the funny name.

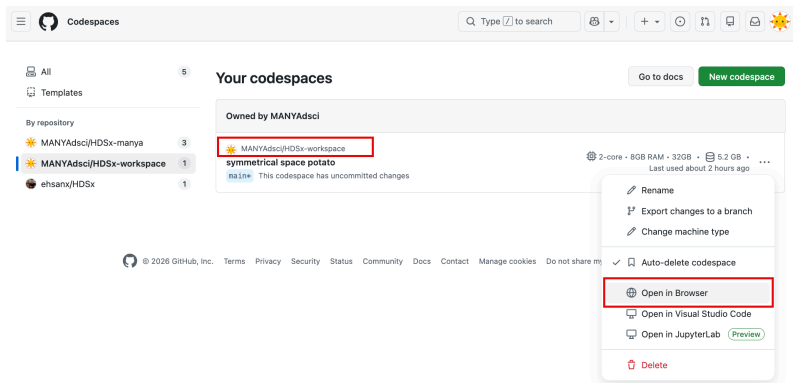
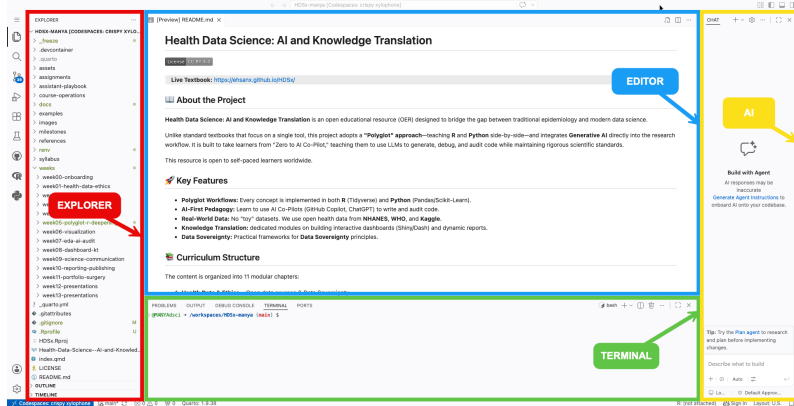


Figure 2: Your Codespace

### 4. Wait for it to resume

- **Benefit:** Resuming an existing Codespace is much faster than creating a fresh one because the software is already installed.

# VS Code Orientation



When you open a Codespace, you are using a version of **VS Code** in your browser. It has the same interface as the desktop version used by professional software engineers at Google and Microsoft.

You can complete the course using VS Code in your web browser through GitHub Codespaces. You do not need to install the desktop version of VS Code unless you choose to.

**Don't Panic:** The interface looks complex because it has hundreds of features. **You only need to know three areas.** You can safely ignore 90% of the buttons you see.

## Explorer (left)

The “File Cabinet”



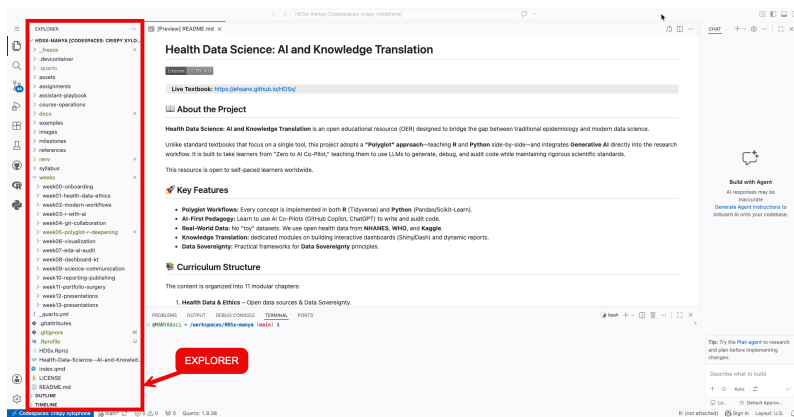


Figure 1: Shows files and folders.

- This shows your folders (directories) and files.
- Think of this as your navigation pane. Double-click a file here to open it in the Editor.

## Editor (center)

### The “Notebook”

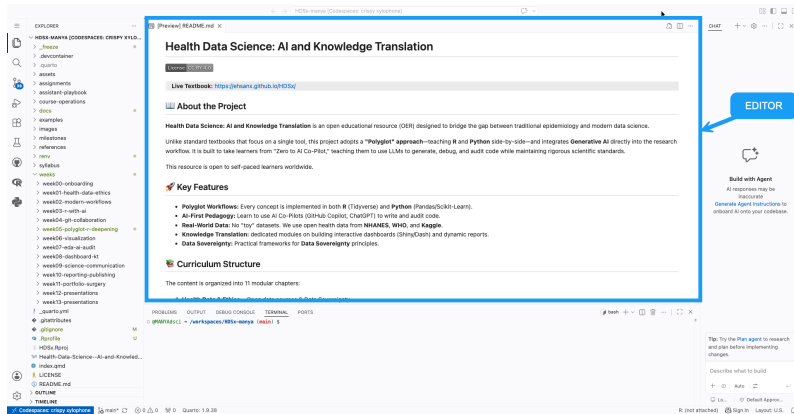


Figure 2: Where you type.

- This is where you do your actual work: typing code and writing text.
- In this course, you will mostly be editing .qmd (Quarto) files here.

## Terminal (bottom)

### The “Engine Room”

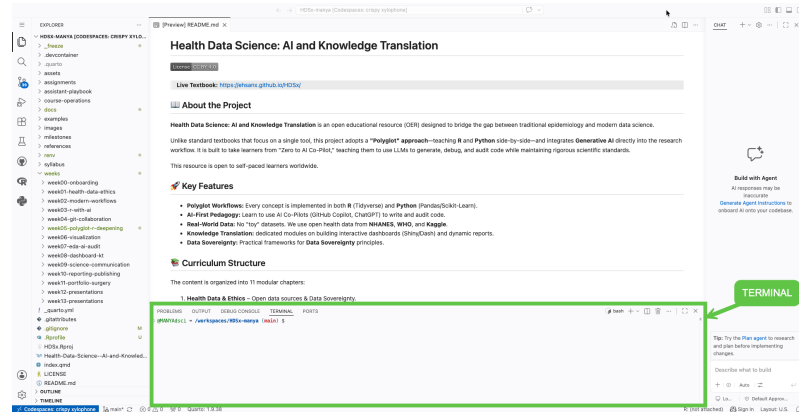


Figure 3: Terminal window

- This is where R and Python actually run.
- You will see text scrolling here when you “Render” a document. It looks like “The Matrix,” but it is usually just the computer telling you what it is working on.

If the terminal disappears (or you accidentally close it):

**View → Terminal**

### Mini Task

Open and close a few files in the Explorer to get comfortable with the layout.

# Files vs Folders

We use a consistent structure to ensure reproducibility. In Data Science, being “organized” isn’t just about being neat—it is a requirement for your code to work on someone else’s machine.

## The Golden Rule of Health Data

We treat data like a **Crime Scene**—never touch the evidence.

- **Start:** `data/` (The Crime Scene). **READ ONLY.**
  - *Rule:* Never open a dataset in Excel and “fix” a typo. That is tampering with evidence.
  - *Correct Way:* Load the “dirty” data into R/Python and fix the typo using code. This leaves a trail of exactly what you changed.
- **End:** `output/` (The Evidence). All graphs and cleaned tables go here.
- **The Detective:** Your code (`.qmd`) moves information from Start to End.

## Files vs. Folders (The Mental Model)

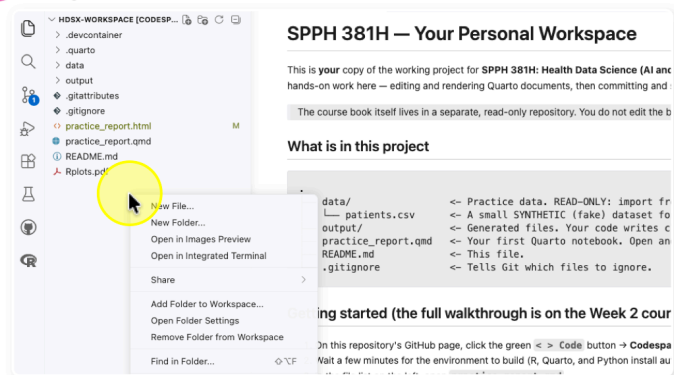
- **Files (e.g., `report.qmd`):** These are **sheets of paper**. You write on them.
- **Folders (e.g., `data/`):** These are **labeled boxes**. You store things in them.

## Create a folder

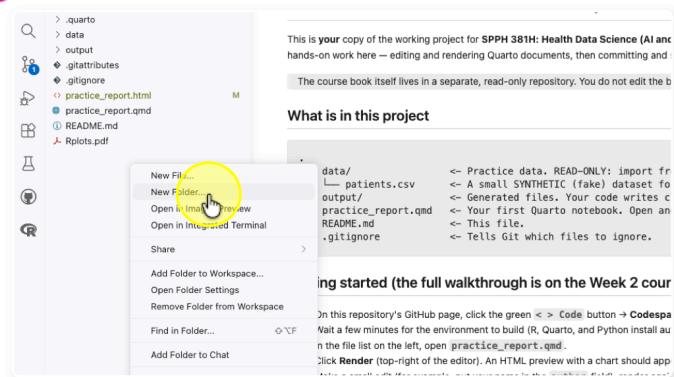
In Explorer:

1. Right-click anywhere in the blank space of the Explorer

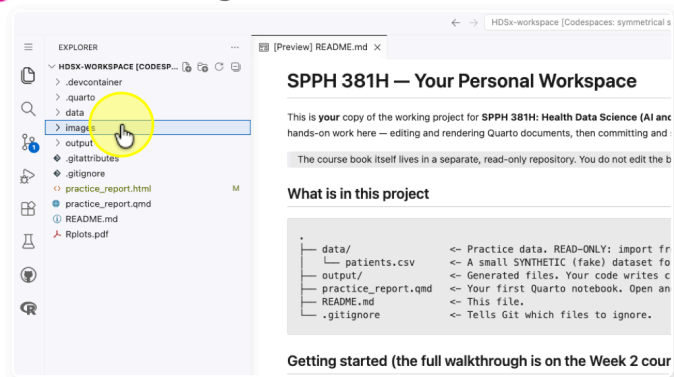
## 1 Right-click here



## 2 Click on "New Folder..."



## 3 Name it: 'images'



pane.

2. Select **New Folder**

3. Name it:

images

**Pro Tip:** Always use **lowercase** and **no spaces** for folder names (e.g., `my_data`, not `My Data`). Computers struggle with spaces in file paths.

## Typical structure

Your project should always look like this:

|                      |                                |
|----------------------|--------------------------------|
| <code>data/</code>   | <code>raw data</code>          |
| <code>output/</code> | <code>generated results</code> |
| <code>images/</code> | <code>screenshots</code>       |
| <code>.qmd</code>    | <code>reports</code>           |

**Remember:** The `data` folder is for *input*. The `output` folder is for *results*. They should never mix.

# The Magic Moment: Render Quarto

## Why do we “Render”?

In the old days, you would run code, copy a graph, save it as a JPG, and paste it into Word. If the data changed, you had to redo every step manually.

**The Better Way:** Think of your code (.qmd file) as a **Recipe** and the final report (HTML/PDF) as the **Meal**. \* **Editing:** You don’t change the meal; you change the recipe. \* **Rendering:** This is the “cooking” process. When you click **Render**, the computer follows your instructions from scratch to create a fresh meal.

This is the secret to reproducibility: if you give your recipe to a friend, they can cook the exact same meal.

## Step 1 — Open the file

1. Go to the **Explorer** pane (Left side).
2. Locate and click on the file:

```
practice_report.qmd
```

This will open the “Source Code” in the center Editor window. You will see a mix of normal text and code chunks.

## Step 2 — Edit name (The YAML Header)

Look at the very top of the file. You will see a block of text sandwiched between three dashes (`---`). This is called the **YAML Header**. It controls the document’s metadata.

Find the line that says `author:`

```
author: "Your Name"
```

**Task:** Delete “Your Name” and type your actual name inside the quotes. Do not remove the quotes!

## Step 3 — Render

💡 How to render: the course default

1. Open the `.qmd` file you want to render, then open the **Command Palette** (**View** → **Command Palette**, or press **Ctrl+Shift+P** on Windows/Linux / **Cmd+Shift+P** on Mac) and run **Quarto: Render Document**. The Quarto extension renders the file and opens a preview pane.
2. **Always-works fallback (terminal):** type the render command with the file’s path as shown in the Explorer, for example:

```
1 quarto render practice_report.qmd
```

Always name the file — a bare `quarto render` rebuilds the *whole* project and takes much longer.

3. If you see a **Render** (or **Preview**) button in the editor toolbar, it does the same thing as step 1.

**Where did the output go?** Watch the render log for the **Output created:** line — it names the exact `.html` file created. By default the file appears next to your `.qmd`; in projects that set an output directory (like this book’s



docs/ folder), it appears there instead. If no preview opened automatically, find that `.html` file in the Explorer, right-click it, and choose **Download** to open it in your browser (or **Open Preview** if available).

After you start the render, **observe**:

- **The Terminal (Bottom):** You will see text scrolling quickly. This is your code chunks executing.
- **The Preview (Right):** A new pane should appear showing a beautiful, formatted report with your name on it.

#### **i** Where is my HTML file?

For your `practice_report.qmd`, the rendered file (`practice_report.html`) is saved **right next to the .qmd file** in the Explorer panel. Watch the render log for the **Output created:** line — it names the exact file. To view it, right-click the `.html` file in the Explorer and choose **Download** (or **Open Preview** if available); typing the file name in the terminal will not open it. (In some larger projects an output directory is configured so the file lands there instead, but your practice project does not set one.)

**Congratulations!** You just performed “Knowledge Translation”—turning raw code into a human-readable document.

## If it fails (Troubleshooting)

Rendering involves many moving parts, so it sometimes gets stuck.

- **The “Wait” Rule:** If nothing happens for 20 seconds, the background process might be initializing. Give it a moment.
- **The “Retry” Rule:** Sometimes the connection hiccups. Run **Quarto: Render Document** one more time.

- **The “No Button” Rule:** The Render button sometimes does not appear in a fresh or stale Codespace. You do not need it — the Command Palette and terminal methods above always work.
- **The “TeX Warning” Rule:** If you see a warning that no TeX installation was found, you can ignore it as long as your HTML file was created successfully. Only run `quarto install tinytex` if you need to produce PDF output.
- **The “Red Text” Rule:** Look at the **Terminal** tab at the bottom. If you see red text, it is an error message. Read it—it usually tells you exactly what went wrong (e.g., “File not found” or “Syntax error”).

Still stuck after trying these rules? See [Troubleshooting: Rebuild Container](#) for the step-by-step fix with screenshots.

# Commit Your Work

Your work is currently only on the “Rented Desk” (Codespace). It is a **Draft**. To save it permanently to the “Vault” (GitHub), you must **Commit** and **Push** (Send).

**The Analogy:** Think of this like shipping a package. You have packed the box (saved the file), but now you need to put a label on it and hand it to the courier.

Think of this as a three-step save process:

1. **Stage** = choose which changed files to include.
2. **Commit** = save a snapshot with a message.
3. **Sync** = upload that snapshot to GitHub so it is safely stored online.

## Steps

1. Click the **Source Control** icon
  - It looks like a **blue bubble** or a **connect-the-dots graph** on the far left sidebar.
  - This opens the “Shipping Department.” You will see a list of every file you changed.
2. Stage the two Assignment 1 files
  - Hover over `practice_report.qmd` and `practice_report.html`, then click the **+** beside each file.
  - Check the **Staged Changes** list before continuing. Do not stage unrelated files.
3. Type a **Commit Message**

- **The Label:** You cannot ship a package without a label.
- **What to write:** Be descriptive but brief. Imagine you are leaving a note for your future self about *what* you did.
- *Example:* “Updated author name in practice report”.
- *Bad Example:* “stuff” or “fix”.

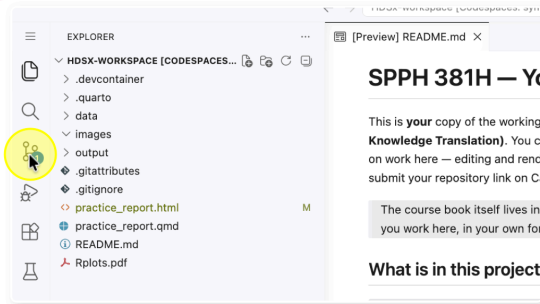
4. Click **Commit**

- This records the staged files as one snapshot in the Codespace.

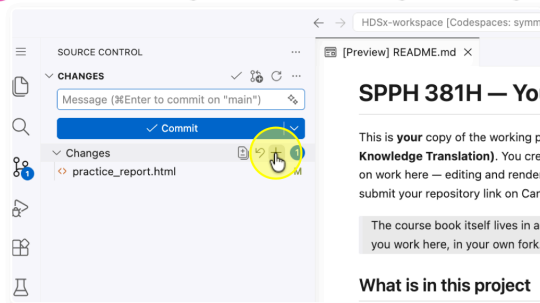
5. Click **Sync Changes** (or **Push**)

- This sends the commit to your GitHub repository.
- *Note:* You might be asked to click “Always” or “Yes” to approve the sync.

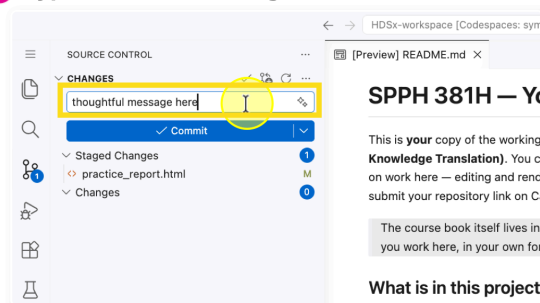
1 Click the Source Control icon



2 Click '+' to stage files, or '+ Changes' to stage all.



3 Type a commit message



4 Select 'Commit and Sync' from dropdown

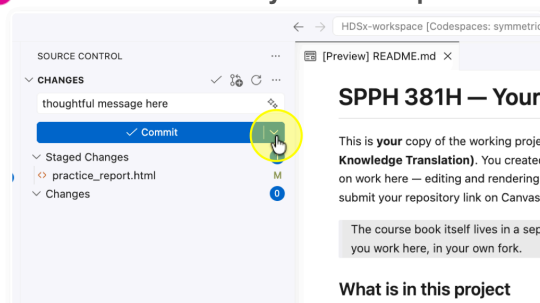


Figure 1: This will push your changes to the remote branch

## Verify (Trust but Verify)

Did it actually work?

1. Go back to your **GitHub Repository** tab (The Vault).
2. **Refresh** the page.
3. Look at the file list. You should see your “Commit Message” displayed next to the file you just edited.

# Troubleshooting: Rebuild Container

Sometimes the cloud computer gets “stuck.” Maybe the terminal froze, or R crashed.

**The Fix:** We don’t fix broken computers; we get new ones. “Rebuilding the Container” is the cloud equivalent of “**Have you tried turning it off and on again?**”—but better, because it gives you a fresh machine while keeping your files safe.

## When to use this?

If things break:

- **Terminal frozen:** You type, but nothing appears.
- **Render fails:** You get weird errors that make no sense.
- **Environment issues:** R or Python can’t find a package that was there yesterday.

One warning that does **not** need a rebuild: “no TeX installation was found.” That one is harmless as long as your HTML file was created — see the TeX Warning rule in [Render with Quarto](#).

Before you rebuild, commit and sync your saved work first — see [Commit Your Work](#) for the steps with screenshots.

## Steps

1. **Open the Command Palette**

- Press **F1** (or **Ctrl+Shift+P** on Windows / **Cmd+Shift+P** on Mac).
- A search bar will appear at the very top of the screen. This is the “God Mode” search bar for VS Code.

## 2. Search

- Type: **Rebuild Container**

## 3. Execute

- Click on the option that says: **Codespaces: Rebuild Container**

## 4. Wait (~2-5 minutes)

- The screen will reload.
- **Don’t Panic:** You might see a black screen with scrolling text logs. This is normal: it is reinstalling the software.
- When your files reappear in the Explorer, you are ready to go.

## Did I lose my work?

- **Saved files:** No. Anything you pressed **Ctrl+S** on is safe.
- **Unsaved typing:** Yes.
- **R Memory:** Yes. You will need to re-run your code chunks from the top.



# Stop Your Codespace

GitHub Free personal accounts currently include 120 core-hours per month. On the default 2-core course machine, that is about 60 hours of active runtime.

**The Analogy:** Think of a Codespace like a **Taxi**.

- **Working:** When you are coding, the meter is running.
- **The Trap:** If you just close your browser tab, it is like getting out of the taxi but asking the driver to “wait here.” **The meter keeps running.**
- **The Fix:** You must explicitly “end the ride” (Stop the Codespace).

## Correct shutdown

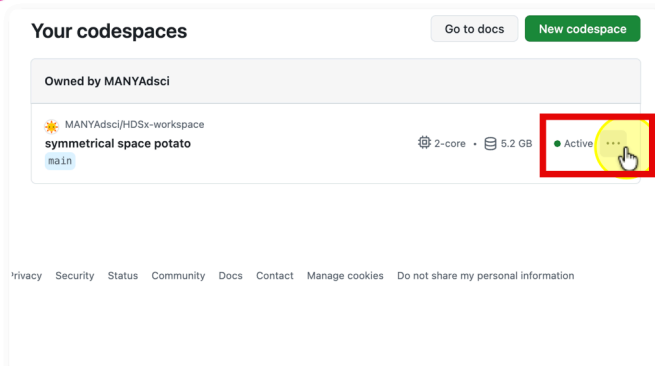
1. Go to the **Codespace Dashboard**: <https://github.com/codespaces>
  - This is your command center.
2. Find your active Codespace
  - Look for the green text that says “Active”.
3. Click ... (The “Meatball” Menu)
  - It is located on the far right side of the Codespace box.
4. Click **Stop Codespace**
  - **Stop:** This stops compute usage while keeping saved files and the installed environment available for the next restart. Running programs and in-memory R or Python objects stop. **(Recommended)**

- **Delete:** This destroys the computer. You will have to wait 2-3 minutes for the “Boot Up” phase next time you start.

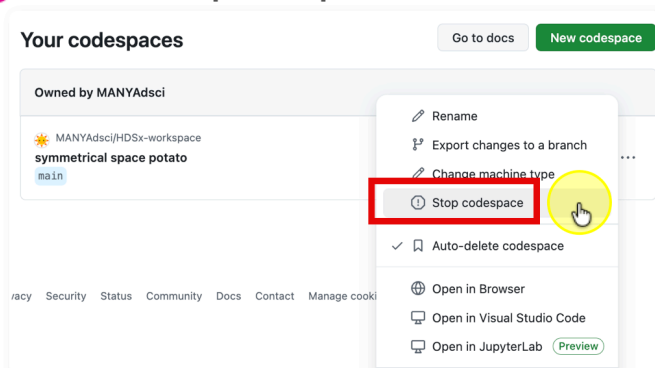
**Critical Warning:** Closing the browser tab is **NOT** enough.

- By default, GitHub keeps the machine running until **30 minutes of inactivity** have passed after you close the tab. Students can change this idle-timeout setting, so stop the Codespace explicitly rather than waiting.
- **The Cost:** If you just close the tab every day, you will waste ~15 hours of your monthly quota on empty time.

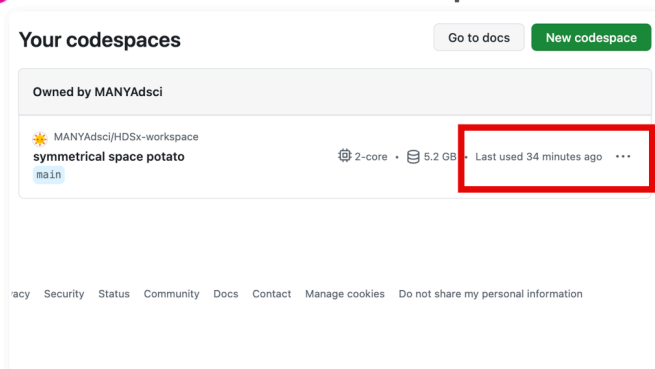
1 Click ‘...’



2 Click on "Stop codespace"



3 Reload to check that the codespace is not active



## Final Checklist

- ☐ I understand repository vs Codespace
- ☐ I created or located my forked course repository
- ☐ I rendered a `.qmd` file
- ☐ I committed changes
- ☐ I know how to rebuild container
- ☐ I know how to stop Codespace

## Reflection

Write 2–3 sentences:

- What was easiest?
- What was confusing?
- What questions do you still have?

## **Week 3: R with AI**

# Week 3: R with AI

## Overview

**Summary:** In Week 2 you learned how to use your Codespace, edit a Quarto document, and make your first commit. Now you learn how to speak R—not by memorizing syntax, but by directing an AI co-pilot and *auditing* what it writes. Your job is to be the pilot: you decide what the analysis should do, the AI drafts the code, and you verify every line before it flies.

**What is R?** R is the main programming language we use in this course for cleaning, summarizing, and visualizing health data. GitHub and Codespaces provide the workspace; R is the tool we use inside that workspace to work with data.

### Learning Objectives:

- Understand that R is a calculator that needs specific, case-sensitive instructions.
- Load packages with `library()` and explain why this is needed each session.
- Explain how the course devcontainer and dependency notes make package use visible.
- Use core constructs: variables (assignment), functions (with arguments), and the pipe (`%>%` / `|>`).
- Perform basic data transformations using `dplyr` verbs: `filter()`, `select()`, `mutate()`, `summarise()`, `group_by()`.
- Inspect a dataset on first contact: structure, dimensions, missingness.
- Identify control flow (`if/else`, loops) in AI-generated code and audit whether it is correct.
- Write and adapt a simple custom function.
- Apply an AI safety checklist to catch hallucinations, wrong column names, and unsafe operations.

- Document your analysis in Quarto by weaving code chunks with Markdown narrative.
- Commit incrementally as you work (reinforcing Week 2 Git basics).

## Connection

Week 2 made the project structure reproducible. This week you use that structure to load, inspect, summarize, and audit R code with AI support. Next week you will put those edits under version control.

Before you start, make sure your tools are ready: [The Setup \(The Cockpit\)](#).

**i** How to use the code blocks on this page

For each code block below, follow the label above it: **Read** means inspect only, **Run** means execute it as written, **Modify** means make the requested change, and **Copy** means place it in the assignment file named in the instructions.

## Case Study Data Analysis

**What to do with this code (the case-study block below):** Run — it loads the cached NHANES CSV and prints two summaries. You do not need to modify it this week.

The **NHANES Health Equity data spine** now supports code-reading, EDA, visualization, and reproducibility checks. Use the cached CSV/RDS for routine class work; a CDC retrieval script exists for advanced users.

- Cached RDS: [examples/nhanes-equity/data/nhanes\\_equity\\_v6.rds](#)
- CSV snapshot: [examples/nhanes-equity/data/nhanes\\_equity\\_v6.csv](#)
- Case-study README: [examples/nhanes-equity/README.md](#)

The default classroom path is to use the cached data so the analysis work is reproducible without internet access.

**What to do here:** you may run the code below — it loads the cached CSV and prints two small summaries; reading it carefully is enough unless your week's page asks for more.

```
1 library(tidyverse) # AI-EDIT(2026-06-23): tidyverse-default - consolidated core library() call
2
3 candidate_paths <- c(
4   "examples/nhanes-equity/data/nhanes_equity_v6.csv",
5   "../examples/nhanes-equity/data/nhanes_equity_v6.csv"
6 )
7
8 nhanes_path <- candidate_paths[file.exists(candidate_paths)][1]
9 if (is.na(nhanes_path)) {
10   stop("Could not find examples/nhanes-equity/data/nhanes_equity_v6.csv")
11 }
12
13 nhanes_analysis <- read_csv(nhanes_path, show_col_types = FALSE)
14
15 nhanes_analysis |>
16   summarise(
17     rows = n(),
18     missing_bmi = sum(is.na(BMI)),
19     missing_income = sum(is.na(IncomeGroup))
20   )
21 #> # A tibble: 1 x 3
22 #>   rows missing_bmi missing_income
23 #>   <int>      <int>      <int>
24 #> 1 105626      9356      9415
25
26 nhanes_analysis |>
27   filter(!is.na(BMI), !is.na(IncomeGroup), Age >= 20, Age <= 80) |>
28   group_by(IncomeGroup) |>
29   summarise(
30     n = n(),
31     mean_bmi = round(mean(BMI), 1),
32     .groups = "drop"
33   )
34 #> # A tibble: 3 x 3
```



```

35 #>   IncomeGroup           n mean_bmi
36 #>   <chr>           <int>   <dbl>
37 #> 1 High Income (>3.5) 16330    28.6
38 #> 2 Low Income (<1.3) 15293    29.4
39 #> 3 Middle Income     19228    29.3

```

## In-Class Activity

- Load the CSV snapshot and inspect column names, row count, and missingness.
- Summarize one numeric variable and one grouping variable with clear NA handling.
- Ask an AI assistant to comment on your code, then audit whether the comments match the actual operations.
- Note one place where an AI assistant could hallucinate a variable name or overstate what the descriptive summary means.

Student output: a short Quarto note with one data inspection, one summary, and one AI-audit comment. Assignment link: [Assignment 2](#).

---

## Packages: Opening the Toolbox

Deeper dive: [Packages \(The Toolbox\)](#).

### Install vs Load

- `install.packages("tidyverse")` — **buying** the toolbox. Done once per environment (already done in your course Codespace).
- `library(tidyverse)` — **opening** the toolbox. Required every time you restart R or reopen your Codespace.

**What to do with this code:** Run — execute it as written every time you reopen your Codespace.

**Where to run this:** First start R so you are *inside* an R session. Open a terminal in VS Code (**Terminal** → **New Terminal**), type R, and press Enter. You should now see the R prompt, which looks like >. Then type the line below at that > prompt (you can also run it from the R Console or a code chunk in your .qmd).

```
1 library(tidyverse)
```

#### Note

If you see an error like `there is no package called 'tidyverse'`, the package is not installed in your Codespace. Use the Canvas support channel or office hours and include the full error message.

## Course Environment and Dependency Note

The personal workspace devcontainer is the supported course environment. It installs the R and Python packages used by the required pathway, so you do not need to initialize `renv`, create a lockfile, or install packages for Assignment 2.

For every submitted analysis:

1. Load each R package explicitly with `library(...)` in the Quarto source.
2. Name the packages you used in `dependency-note.md`.
3. State whether you used only the course defaults.
4. If you think you need another package, ask the instructor or TA before adding it. Record an approved addition in the workspace devcontainer setup and in your dependency note so a fresh Codespace can reproduce it.

For Assignment 2, a complete dependency note can be as short as:

```
1 Packages used: tidyverse. No packages were added beyond the course Codespace baseline.
```

The test is practical: your document must render in a fresh course Codespace.

---

## Core Constructs: Boxes, Toasters, and the Assembly Line

Deeper dives: [Basics: Boxes & Toasters](#); the pipe is also covered in [Control Flow](#).

### Variables: The Labeled Box

R has no memory unless you store results in a named box using `<-`.

**What to do with this code:** Run — type it in the Console, then check the Environment pane to see your new “boxes”.

```
1 patient_age <- 55
2 clinic_name <- "Vancouver General"
```

#### Warning

**Case sensitivity:** Age, age, and AGE are three different boxes. This is one of the most common bugs in AI-generated code. Always check that column names match your data exactly.

## Functions: The Toaster

A function takes **input** and returns **output**. Arguments (inside the parentheses) control its behaviour.

**What to do with this code:** Run — execute it as written.

```
1 mean(c(72, 85, 90), na.rm = TRUE)
```

- `c(72, 85, 90)` is the input (a vector of numbers).
- `na.rm = TRUE` tells R to ignore missing values. **Always look for this** when auditing AI code that works with health data — missingness is the norm, not the exception.

## The Pipe: The Assembly Line

The pipe passes the result of one step into the next. Think of it as “**and then...**”

**What to do with this code:** Read — data here is a placeholder, so this block will not run as-is.

```
1 data %>%  
2   filter(age > 65) %>%  
3   summarise(mean_bmi = mean(bmi, na.rm = TRUE))
```

Read this as: “*Take the data, and then keep only patients over 65, and then calculate the average BMI.*”

### Tip

You will see two pipe symbols in the wild: `%>%` (from the `magrittr` / `tidyverse` packages) and `|>` (built into R 4.1+). They do the same thing for our purposes. If AI generates the other one, don’t panic — both work in your Codespace.

## Data Import and First Contact

Deeper dive: [The Loading Dock \(Data Import\)](#).

### Loading Data

Use relative paths so your code runs on any machine (including your grader's Codespace).

**What to do with this code:** Modify — replace the path with the relative path to your own file, then run it.

```
1 my_data <- read_csv("data/patients.csv")
```

**How to get the path right:** Right-click the file in the VS Code Explorer → **Copy Relative Path** → paste it into your code.

#### AI Prompt:

Read the CSV file at “data/patients.csv” and save it as `my_data`. Use the `readr` package.

### First Contact: Inspect Before You Analyze

Never start analysis without looking at your data. Run these four checks:

**What to do with this code:** Run — execute it as written once your data is loaded.

Run these in your assignment `.qmd` (as a code chunk) or at the R > prompt, using the course workspace repo. `glimpse()` comes from `tidyverse`, so if you get `could not find function "glimpse"`, run `library(tidyverse)` first.

```
1 # How big is it?
2 dim(my_data)           # rows x columns
3
4 # What are the column names and types?
5 glimpse(my_data)
6
7 # Quick summary statistics
8 summary(my_data)
9
10 # How much is missing?
11 colSums(is.na(my_data))
```

Open the **Environment** pane (top-right in RStudio or via the R extension) to see your variables and click a data frame to view it in a grid.

#### Warning

**Health data reality:** Real datasets like NHANES have substantial missingness. Knowing *where* and *how much* is missing is not optional — it shapes every downstream decision.

A concrete example: in NHANES, income questions are skipped far more often than age questions. If you silently drop rows with missing income, your “average” quietly excludes many of the very people a health-equity analysis is trying to describe. Handling missingness carefully is part of responsible data stewardship, not just a coding chore.

---

## Data Transformation: The Five dplyr Verbs

These five verbs are the core of data wrangling in R. You will use them in nearly every assignment and in your term project.

**What to do with the code in this section:** Modify — swap in your own dataset and column names, then run each block.

### **filter() — Keep rows that match a condition**

```
1 # Patients aged 65 and older
2 seniors <- my_data %>% filter(age >= 65)
```

### **select() — Keep (or drop) columns**

```
1 # Keep only ID, age, and BMI
2 slim <- my_data %>% select(id, age, bmi)
```

### **mutate() — Create or modify a column**

```
1 # Add a BMI category column
2 my_data <- my_data %>%
3   mutate(bmi_category = case_when(
4     bmi < 18.5 ~ "Underweight",
5     bmi < 25   ~ "Normal",
6     bmi < 30   ~ "Overweight",
7     TRUE      ~ "Obese"
8   ))
```

### **summarise() — Collapse rows into a summary**

```
1 my_data %>% summarise(mean_age = mean(age, na.rm = TRUE))
```

### **group\_by() + summarise() — Summaries by group**

```
1 my_data %>%
2   group_by(sex) %>%
3   summarise(
4     n = n(),
5     mean_bmi = mean(bmi, na.rm = TRUE)
6   )
```

### AI Prompt:

Using `dplyr`, filter my dataset to patients over 65, group by sex, and calculate the mean BMI for each group. Handle missing values.



#### Tip

**Audit the AI output:** After running the AI-generated code, check that (a) the column names match your actual data, (b) `na.rm = TRUE` is present where needed, and (c) the row counts make sense (e.g., the filtered data should be smaller than the original).

**Git checkpoint:** You have now loaded, inspected, and transformed data. Stage your `.qmd`, write a commit message describing what you did, and commit. Don't wait until the end.

---

## Control Flow: Auditing AI Decisions

Deeper dive: [Control Flow](#).

You will not write many `if/else` blocks or loops by hand. But AI frequently generates them, so you need to **read** them.

**What to do with the code in this section:** Read — these are patterns to recognize in AI output; you do not need to run them.

### If/Else: The Fork in the Road

```
1 if (mean_age > 50) {  
2   print("Older cohort")  
3 } else {  
4   print("Younger cohort")  
5 }
```



**Audit question:** Is the threshold (50) appropriate for my research question, or did the AI pick an arbitrary number?

## Loops: The Treadmill

```
1 for (col in c("age", "bmi", "sbp")) {  
2   print(summary(my_data[[col]]))  
3 }
```

**Audit questions:** Does the loop have a clear end condition? Could this be replaced with a simpler vectorized operation (e.g., `summarise(across(...))`)?

### Note

In R, loops are rarely the best tool. If AI gives you a loop, ask: “*Can this be done without a loop using dplyr or purrr?*” The answer is usually yes.

---

## Writing a Simple Function

The syllabus asks you to write at least one custom function. A function packages a repeated task so you don’t copy-paste code.

### Example: A Reusable Summary

**What to do with this code:** Run, then Modify — run it as written on your data, then change one thing as described in “Your Task” below.

```

1 summarise_column <- function(data, column_name) {
2   data %>%
3     summarise(
4       n_total    = n(),
5       n_missing  = sum(is.na(.data[[column_name]])),
6       mean_val   = mean(.data[[column_name]], na.rm = TRUE),
7       sd_val     = sd(.data[[column_name]], na.rm = TRUE)
8     )
9   }
10
11 # Use it
12 summarise_column(my_data, "bmi")
13 summarise_column(my_data, "age")

```

## Your Task

1. Ask your AI co-pilot to write a function that takes a data frame and a column name and returns summary statistics.
2. **Audit the code:** Does it handle NA? Does it work on a numeric column? What happens if you pass a categorical column?
3. **Adapt it:** Change one thing (e.g., add a median, or add a `group_by` argument). This is the difference between accepting code blindly and understanding it.

## AI Prompt:

Write an R function called `summarise_column` that takes a data frame and a column name (as a string) and returns the count, number of missing values, mean, and standard deviation. Then show me how to call it on the “bmi” column of `my_data`.

---

## Debugging: Reading the Red Text

Deeper dive: [The Detective \(Debugging\)](#).

Errors are normal. AI makes mistakes, you make typos, and data surprises you.

## The “Fix It” Loop

1. **Read** the error message. Look for keywords: `not found`, `unexpected`, `non-numeric`.
2. **Copy** the full error text.
3. **Paste** it into your AI chat and ask:

Here is an R error from my Codespace: [paste]. Explain what went wrong in one sentence and give corrected code.

## Common Errors and What They Mean

| Error message                           | Likely cause                                                      |
|-----------------------------------------|-------------------------------------------------------------------|
| object 'X' not found                    | Typo in variable name, or you forgot to run an earlier chunk      |
| could not find function "X"             | Forgot <code>library(tidyverse)</code> or misspelled the function |
| non-numeric argument to binary operator | Trying to do math on a text/factor column                         |
| unexpected symbol                       | Missing comma, parenthesis, or pipe                               |

### Warning

**AI debugging guardrail:** When AI suggests a fix, re-read the corrected code before running it. Does the fix address the *actual* error, or did AI rewrite the whole chunk and change your logic?

When asking AI for help, explicitly say: *“Do not change the logic or intended result of my code. Explain any suggested change before rewriting it.”*

## AI Safety: The Audit Checklist

Deeper dive: [AI Safety: The Audit Checklist](#).

Every time you accept AI-generated code, run through this checklist:

1. **Column names:** Are they spelled exactly as they appear in `glimpse(my_data)`?
2. **Data types:** Is the code doing math only on numeric columns?
3. **Missingness:** Is `na.rm = TRUE` present where needed?
4. **Privacy:** Does the code avoid printing individual patient rows that could be identifiable?
5. **Logic:** Does the code do what *you* intended, or what the AI *assumed* you meant?

## The Paper Trail Rule

Deeper dive: [The Paper Trail \(Comments\)](#).

Every time you use AI to write a code chunk, add a `#` comment above it explaining what you asked for and what you verified:

**What to do with this code:** Read — then imitate this comment pattern above your own AI-assisted chunks.

```
1 # AI prompt: "Calculate mean BMI by sex, handling NAs"
2 # Verified: column names match, na.rm = TRUE present, output has 2 rows (M/F)
3 my_data %>%
4   group_by(sex) %>%
5   summarise(mean_bmi = mean(bmi, na.rm = TRUE))
```

This comment trail is your audit log. It demonstrates accountability (required by the course AI policy) and helps you debug later.

## Exporting Clean Data

After cleaning and transforming, save your work so future weeks can pick up where you left off.

**What to do with this code:** Run — it creates `output/clean_data.csv`; create the `output/` folder first if it does not exist.

```
1 # Save a clean version for Week 4 and beyond
2 write_csv(my_data, "output/clean_data.csv")
```



Tip

**Reproducibility habit:** Your Quarto document should contain *every step* from raw data to clean export. Anyone (including your grader) should be able to render it from scratch and get the same `clean_data.csv`.

**Git checkpoint:** Your analysis is complete. Stage, commit with a clear message (e.g., "Complete Week 3: load, inspect, transform, and export patient data"), and Sync.

---

## Literate Analysis in Quarto

Your `.qmd` file is not just code — it is a **document**. Weave narrative around your code chunks to explain what you are doing and why.

### Pattern

**What to do with this code:** Read — a pattern to imitate in your own document, not text to paste as-is.

```

1  ## Data Inspection
2
3  We begin by examining the structure of the dataset to understand
4  variable types and missingness before any transformations.
5
6  ```{r, eval=FALSE}
7  glimpse(my_data)
8  colSums(is.na(my_data))
9  ```
10
11
12  The dataset contains `nrow(my_data)` observations across
13  `ncol(my_data)` variables. Notably, BMI is missing for
14  approximately 12% of records, which we handle with `na.rm = TRUE`
15  in downstream summaries.

```

This interleaving of code and prose is **literate programming** — the methodology that underpins reproducible research in this course. Practice it now; you will need it for your term project report (Week 10) and final portfolio.

---

## Knowledge Check

1. What is the difference between `install.packages()` and `library()`?
2. Why is `na.rm = TRUE` important when working with health data?
3. What does `my_data %>% filter(age > 65) %>% summarise(n = n())` do? Read it aloud using “and then.”
4. Name the five core `dplyr` verbs and what each does in one sentence.
5. You see the error object `'BMI' not found` but your column is called `bmi`. What went wrong?
6. What is the difference between `%>%` and `|>`? Does it matter for this course?

7. Why should you write `#` comments above AI-generated code?
  8. What four checks should you run immediately after loading a new dataset?
- 

## Assignment 2: R With AI

Full instructions: [Assignment 2 README](#). Short version: [Assignment 2 walkthrough](#).

**Task:** Build a short Quarto report that loads, inspects, transforms, and summarizes a health dataset — using your AI co-pilot for code generation and your own judgment for verification.

### Step-by-Step Workflow

1. Open the [Assignment 2 README](#) and follow its setup steps: copy the starter file (`assignments/assignment02-r-with-ai/starter-analysis.qmd`) into your submission folder as `analysis-note.qmd`. The dataset is `examples/nhanes-equity/data/nhanes_equity_v6.csv` ([view on GitHub](#)).
2. **Load packages and data:** Use AI to write the `library()` and `read_csv()` calls. Verify the data loaded correctly with `dim()` and `glimpse()`.
3. **Inspect missingness:** Run `colSums(is.na(...))`. Add a Markdown paragraph interpreting what you see.
4. **Transform:** Use `dplyr` verbs to:
  - Filter to a meaningful subgroup (e.g., `age > 18`).
  - Create at least one new column with `mutate()`.
  - Produce a grouped summary with `group_by()` + `summarise()`.
5. **Write a function:** Create (or adapt from AI) a reusable summary function. Call it on at least two different columns.
6. **Export:** Save your summary table to `output/summary_table.csv`.

7. **Document:** Every code chunk should have:
  - A `#` comment stating what you asked the AI and what you verified.
  - A Markdown paragraph below interpreting the result.
8. **Render** the document to verify it runs cleanly from top to bottom.
9. **Git:** Stage, write a descriptive commit message, Commit, and Sync.

## Deliverable

A rendered `.qmd` that a reader can follow as a narrative: what data you started with, what you did to it, what you found, and why your decisions were justified.

## Quick Reference

See also: [Final Summary Checklist](#).

| Concept           | R code                                 | Notes                                       |
|-------------------|----------------------------------------|---------------------------------------------|
| Load packages     | <code>library(tidyverse)</code>        | Every session                               |
| Read CSV          | <code>read_csv("data/file.csv")</code> | Use relative paths                          |
| Assignment        | <code>x &lt;- 5</code>                 | The <code>&lt;-</code> arrow stores a value |
| Inspect structure | <code>glimpse(df)</code>               | Column names, types, preview                |
| Dimensions        | <code>dim(df)</code>                   | Rows $\times$ columns                       |
| Summary stats     | <code>summary(df)</code>               | Min, median, mean, max, NAs                 |
| Check missingness | <code>colSums(is.na(df))</code>        | Count NAs per column                        |
| Filter rows       | <code>filter(df, age &gt; 65)</code>   | Keep matching rows                          |



| Concept           | R code                                                | Notes               |
|-------------------|-------------------------------------------------------|---------------------|
| Select columns    | <code>select(df, id, age, bmi)</code>                 | Keep named columns  |
| New column        | <code>mutate(df, bmi_cat = ...)</code>                | Create or modify    |
| Summarize         | <code>summarise(df, m = mean(x, na.rm = TRUE))</code> | Collapse to one row |
| Group + summarize | <code>group_by(df, sex) %&gt;% summarise(...)</code>  | Per-group summaries |
| Export            | <code>write_csv(df, "output/clean.csv")</code>        | Save for later      |
| Pipe              | <code>%&gt;%</code> or <code> &gt;</code>             | “And then...”       |

# The Setup (The Cockpit)

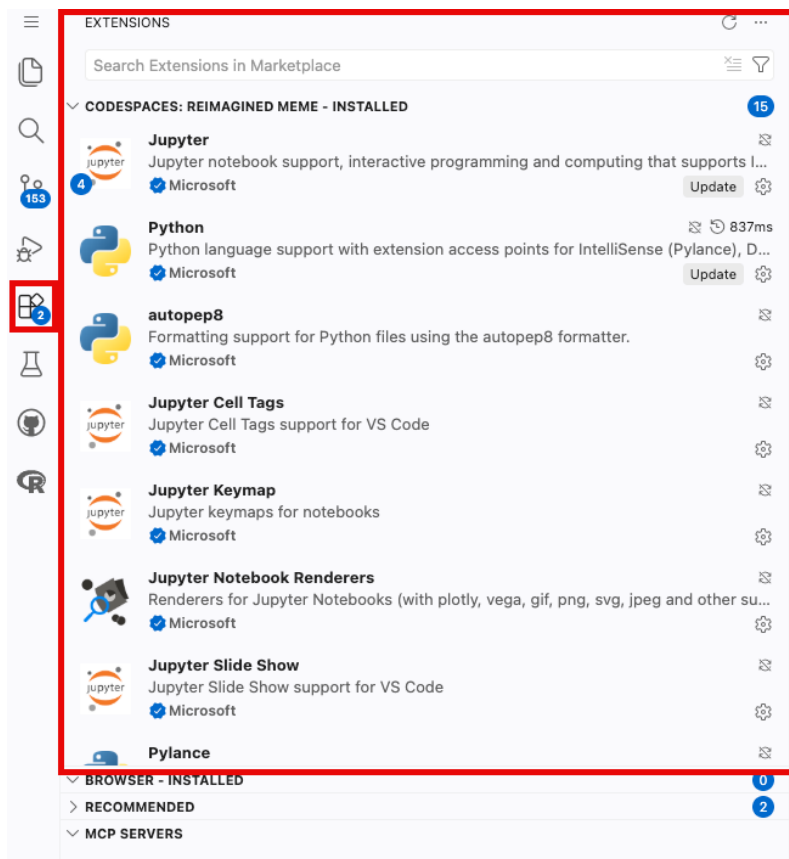
Before we start flying, we need to check our instruments in **VS Code**. If your environment isn't set up correctly, your code will not run.

A quick orientation: GitHub and Codespaces are the *workspace* — they store your files and give you a computer in the browser. R is the *tool* you will use inside that workspace to clean, summarize, and visualize health data.

The Codespace should open ready to use. If R, Quarto, required packages, or Copilot Chat are missing, do not manually change the environment. Instead, rebuild the container by following [Troubleshooting: Rebuild Container](#). If it still fails after a rebuild, ask your TA or instructor for help.

## Verify Your AI Extensions

1. Click the **Extensions** icon on the far-left sidebar (it looks like four squares).
2. In the search bar, ensure both **GitHub Copilot Chat** and **R** are installed and active.



## Pick a Chat Model

1. Open the Copilot Chat sidebar: go to **View** → **Command Palette** and run **Chat: Open Chat**, or click the chat icon at the top of the VS Code window. *Optional tip:* a keyboard shortcut (such as **Ctrl+Alt+I** on Windows) may also work, but shortcuts vary and can conflict with your browser, so use the menu path if nothing happens.
2. Look for the model selector dropdown at the bottom of the chat window.
3. **Quota check:** If you see a model marked as *included* or with a “0x” **multiplier**, pick it — these models do not use up your free student quota. If your picker only shows models marked “1x” or higher, that is okay: pick any

available chat model and continue. Exact model names and multipliers change over time, so your list may not match a classmate's screen.

- *Troubleshooting:* If the Chat asks you to “Sign In” or “Subscribe,” your Student Pack is not active. Use your institutional login at `copilot.microsoft.com` or authenticate via your university email at `https://m365.cloud.microsoft/chat/`.

## Check the Course Package Environment

The personal workspace devcontainer installs the packages required for the course pathway. Do not initialize `renv`. In each analysis file, load the packages you use explicitly, and list them in the assignment's dependency note. If a required package is missing after rebuilding the container, copy the error message and ask your TA or instructor rather than installing packages at random.

## Interface Review: Where does R actually live?

To learn R, we are going to start in the “sandbox.” Here is how your screen is laid out:

- **The Console (Bottom):** Your **Engine Room**. *Start here!* This is where you can type R commands directly next to the `>` prompt and press Enter to see immediate results. If you make a mistake, it will show up here in red text.
- **The Environment (Top Right):** Your **Work Bench**. Whenever you save a piece of data (like a patient's age), it will appear in a list here so you can verify R actually “remembers” it.
- **The Explorer (Left Sidebar):** Your **File Cabinet**. This is where you find your `data/` folder containing your CSV files.

- **The Editor (Top Center):** Your **Laboratory Notebook**. Typing in the Console is great for testing, but it forgets everything when you close the window. Later in the lesson, we will use the Editor to write and save our permanent code.

## Interface Review: Working with .qmd and .Rmd files

When you open a .qmd (Quarto) or .Rmd (RMarkdown) file, you must know where to look. Here is how your coding workflow maps to your screen:

- **The Explorer (Left Sidebar):** Your **File Cabinet**. This is where you find your reports and your `data/` folder. You cannot run code here; you only click files to open them.
- **The Editor (Top Center):** Your **Laboratory Notebook**. This is where you write your text narrative and your R code.
  - *Note:* R code must be placed inside grey **Code Chunks** (starting with ````\r{}` and ending with `````). To run a chunk, click the small green “Play” triangle in the top right corner of that specific grey box.
- **The Console (Bottom):** Your **Engine Room**. When you click the green “Play” button in the Editor, the code is sent down here to be executed. If there is an error, it will show up here in red.
  - *Pro Tip:* You can type quick math directly into the console (like `100 / 4`), but it will not be saved in your final report.
- **The Environment (Top Right):** Your **Work Bench**. Whenever you load data or save a calculation, it will appear in a list here so you can verify R actually “remembers” it.

# Packages (The Toolbox)

In R, tools come in “packages.” Think of `install.packages` as buying a physical toolbox, and `library()` as opening the lid.

- **To open your tools:** Type this in your code and run it:

```
library(tidyverse)
```

**Note:** You must run `library(tidyverse)` every single time you restart your Codespace.

# Basics: Boxes & Toasters

R is a system of storage (Boxes) and transformation (Toasters).

## Variables (The Box)

Use `<-` to save data into a name.

Code snippet

```
Save the number 55 into a variable called patient_age patient_age
```

## Functions (The Toaster)

Functions take input, perform an action, and return output.  
Look at the snippet below

Code snippet

```
calculate_rectangle_area <- function(length, width) {  
  area <- length * width  
  return(area)  
}
```

```
# Example usage:  
calculate_rectangle_area(10, 5) # Returns 50
```

# Control Flow

## The Pipe %>% (The Assembly Line)

The symbol %>% means “**and then...**”.

Both of these lines are identical, which would you prefer to read? (Note |> is another way to write %>%)

```
print(round(sqrt(10), 2))
```

```
10 |> sqrt() |> round(2) |> print()
```

For a real world example of how one may use the pipe

```
library(dplyr)
```

```
mtcars %>%  
  filter(wt > 3) %>%          # 1. Filter for cars heavier than 3,000 lbs  
  group_by(cyl) %>%          # 2. Group by number of cylinders  
  summarize(avg_mpg = mean(mpg)) # 3. Calculate the average MPG
```

## If / Else (The Fork in the Road)

An **if/else** statement lets the computer make a decision.

The program checks a condition:

- If it is **TRUE**, it runs one block of code.
- If it is **FALSE**, it runs another.



```
temperature <- 5
if (temperature > 10) {
  print("You can wear a t-shirt")
} else {
  print("Wear a jacket")
}
```

## What happens

1. The computer checks: `temperature > 10`
2. `5 > 10` is **FALSE**
3. So the **else** part runs.
4. Output:

```
"Wear a jacket"
```

Think of it like a **fork in the road**:

- One path if the condition is true
- Another path if it's false.

## Loops (The Treadmill)

A loop repeats code multiple times.

Instead of writing the same instruction again and again, you let the computer do it.

## Example

```
for (i in 1:5) {
  print(i)
}
```

## What happens

The computer runs the code **5 times**.

Steps:

1. `i = 1` → print 1
2. `i = 2` → print 2
3. `i = 3` → print 3
4. `i = 4` → print 4
5. `i = 5` → print 5

Output:

1 2 3 4 5

## Audit Check (Important)

When using loops, ask:

### 1. Does the loop end?

Bad example (infinite loop):

```
while (TRUE) {  print("This never stops") }
```

### 2. Could this be simpler?

Sometimes R has built-in functions that replace loops.

Instead of:

```
total <- 0 for (i in 1:5) {  total <- total + i }
```

You can write:

```
sum(1:5)
```

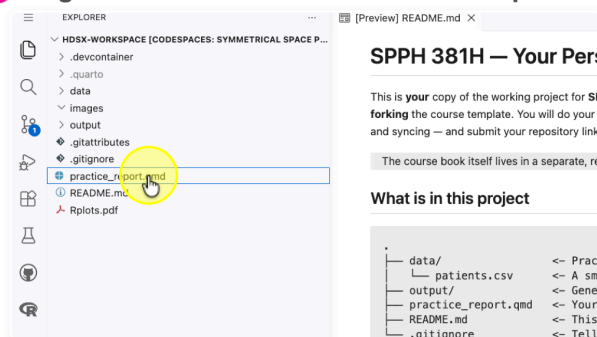
This is **shorter, safer, and faster**.

# The Loading Dock (Data Import)

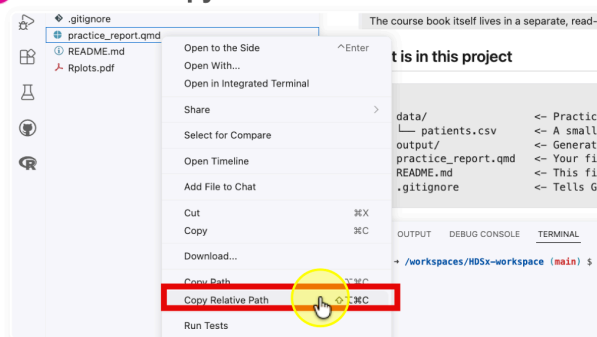
Don't type file paths from memory; you will likely make a typo.

1. Find a data file in the **Explorer** (left sidebar).
2. **Right-click** the file → Select **Copy Relative Path**.

1 Right-click the desired file from the Explorer



2 Click on "Copy Relative Path"



3. Ask AI: “Read the CSV at [Paste Path Here] and save it as my\_data.”

## Mini-Examples: Read In, Write Out

**What to do with this code:** Modify — swap in your own file names, then run it.

```
1 library(tidyverse)
2
3 # Read input: a relative path starts from the project folder,
4 # so it works in anyone's Codespace.
5 my_data <- read_csv("data/patients.csv")
6
7 # Write output: save your results into the project's output/ folder.
8 write_csv(my_data, "output/clean_data.csv")
```

If R says the file does not exist, re-copy the relative path from the Explorer (step 2 above). For `write_csv()`, create the `output/` folder first if it does not exist.

# The Detective (Debugging)

AI makes mistakes. When you see **Red Text** (errors) in the Console:

1. **Copy** the Red Text.
2. **Paste** it into Copilot Chat.
3. **Ask:** *“Explain this error and fix the code below: [Paste Code]”*

Try running this line and ask AI why it doesn't work:

```
"apple" + 5
```

# AI Safety: The Audit Checklist

Use this checklist before accepting any AI code:

- **Spelling:** Are column names spelled exactly as they appear in the data?
- **Case Sensitivity:** Is it `Age` or `age`? (R sees these as different things).
- **Missingness:** Did the AI remember `na.rm = TRUE` for math?
- **Privacy:** Did the AI try to print the whole dataset (potentially exposing patient IDs)?

## Prompt-engineering tip

When asking AI for help, explicitly say: *“Do not change the logic or intended result of my code. Explain any suggested change before rewriting it.”* This keeps the AI in a reviewing role instead of letting it quietly rewrite your analysis.

# The Paper Trail (Comments)

**The Rule:** Every time you use AI to write code, write a # comment in plain English above it explaining what you told the AI to do.

Code snippet

```
# I asked AI to filter for patients with BMI over 30 and count them # [AI CODE BELOW]
```

Below is an example of non readable code

```
# What is x? What is 1.05? Why are we subsetting?
f <- function(x, y) {
  z <- x[x$a > y, ]
  r <- sum(z$b) * 1.05
  return(r)
}
```

```
res <- f(df, 100)
```

And here is an example of readable code

```
library(dplyr)

# Calculate the total revenue for high-value orders including a 5% tax
calculate_total_revenue <- function(order_data, minimum_price_threshold) {

  sales_tax_rate <- 1.05

  total_revenue <- order_data |>
    filter(price > minimum_price_threshold) |>
    summarize(total = sum(price)) |>
    pull(total)
```

```
    return(total_revenue * sales_tax_rate)
}
```

# Usage:

```
final_report_value <- calculate_total_revenue(current_orders, minimum_price_threshold = 100)
```

Just by reading through each line a reader should understand what is going on. One does not need to comment every single line, but rather comment whenever something complex is happening on the code line below it. And as always source work which is not yours.



## Assignment 2: R With AI

1. Open the [Assignment 2 README](#) for the full instructions, then copy the starter file (`assignments/assignment02-r-with-ai/starter-analysis.qmd`) into your submission folder as `analysis-note.qmd`.
2. In your personal workspace, import the NHANES dataset at `examples/nhanes-equity/data/nhanes_equity_v6.csv`.
3. Use the **Pipe** to filter and summarize one variable.
4. **Audit:** Add a `#` comment explaining your prompt.
5. **Render** the file to HTML.
6. **Commit and Push** to GitHub.

## Final Summary Checklist

- ☐ Extensions are active (Copilot + R).
- ☐ Picked a chat model in Copilot Chat (an included “0x” model if available; any available model is fine).
- ☐ Loaded `library(tidyverse)`.
- ☐ Data imported via **Relative Path**.
- ☐ All code chunks have an **Audit Comment (#)**.
- ☐ Handled **NA** values correctly.
- ☐ Document **Rendered** and **Pushed**.

## **Week 4: Git & Collaboration**

# Git & Collaboration

## Overview

**Summary:** You already know how to save files and make a basic commit in your Codespace. Now, learn to use Git as a *time machine* and *collaboration engine* for your analysis—tracking every change, safely undoing mistakes, working with teammates through branches and Pull Requests, and keeping a professional audit trail of your research workflow.

### Learning Objectives:

- Distinguish between saving a file and taking a version-control “snapshot” (commit).
- Read code differences (diffs) to see exactly what changed.
- Use the Codespaces Source Control UI to revert to a previous working version.
- Create and switch branches to isolate experimental work.
- Open a Pull Request on GitHub and understand its role in peer review.
- Configure `.gitignore` and use relative paths for repo hygiene.
- Use AI to translate diffs into clear, professional audit logs.
- Set up a group repository and agree on collaboration norms for the term project.

## Connection

Week 3 focused on writing and auditing small R summaries. This week treats each edit as a reviewable change with a clear history. Next week you will compare equivalent summaries across R and Python.

## Case Study Data Analysis

The **NHANES Health Equity data spine** now supports code-reading, EDA, visualization, and reproducibility checks. Use the cached CSV/RDS for routine class work; a CDC retrieval script exists for advanced users.

- Cached RDS: [examples/nhanes-equity/data/nhanes\\_equity\\_v6.rds](#)
- CSV snapshot: [examples/nhanes-equity/data/nhanes\\_equity\\_v6.csv](#)
- Case-study README: [examples/nhanes-equity/README.md](#)

The default classroom path is to use the cached data so the analysis work is reproducible without internet access.

**What to do here:** you may run the code below — it loads the cached CSV and prints two small summaries; reading it carefully is enough unless your week's page asks for more.

```
1 library(tidyverse) # AI-EDIT(2026-06-23): tidyverse-default - consolidated core library() call
2
3 candidate_paths <- c(
4   "examples/nhanes-equity/data/nhanes_equity_v6.csv",
5   "../..examples/nhanes-equity/data/nhanes_equity_v6.csv"
6 )
7
8 nhanes_path <- candidate_paths[file.exists(candidate_paths)][1]
9 if (is.na(nhanes_path)) {
10   stop("Could not find examples/nhanes-equity/data/nhanes_equity_v6.csv")
11 }
12
13 nhanes_analysis <- read_csv(nhanes_path, show_col_types = FALSE)
14
15 nhanes_analysis |>
16   summarise(
17     rows = n(),
18     missing_bmi = sum(is.na(BMI)),
19     missing_income = sum(is.na(IncomeGroup))
20   )
21 #> # A tibble: 1 x 3
22 #>   rows missing_bmi missing_income
23 #>   <int>      <int>      <int>
```

```

24 #> 1 105626          9356          9415
25
26 nhanes_analysis |>
27   filter(!is.na(BMI), !is.na(IncomeGroup), Age >= 20, Age <= 80) |>
28   group_by(IncomeGroup) |>
29   summarise(
30     n = n(),
31     mean_bmi = round(mean(BMI), 1),
32     .groups = "drop"
33   )
34 #> # A tibble: 3 x 3
35 #>   IncomeGroup      n mean_bmi
36 #>   <chr>          <int>   <dbl>
37 #> 1 High Income (>3.5) 16330    28.6
38 #> 2 Low Income (<1.3)  15293    29.4
39 #> 3 Middle Income     19228    29.3

```

**What to do with the code above:** read it — you do not need to run or modify it this week. This week’s focus is tracking *changes* to files like this one, not the analysis itself.

## In-Class Activity

- Create a branch for a tiny dashboard documentation change.
- Edit only a label, caption, or README sentence; do not change analytic logic.
- Open a pull request and use the diff to verify the change is limited.
- In the pull request description, state how you verified the change is documentation-only.

Student output: a documentation-only pull request with a short audit note. Assignment link: [Assignment 3](#).

## git1.qmd Concepts: The “Save Game” of Reproducible Science

Deeper Dive: [Git Basics](#)

### Why Git Matters in Health Research

- **The audit trail:** Why `analysis_final_v3.qmd` is unsafe in health research; Git gives you a permanent, searchable history.
- **The timeline:** Your repository is a sequence of safe checkpoints (commits). Each commit records *who* changed *what* and *why*.
- **Reading the diff:** Red = removed. Green = added. This is the visual language of reproducibility.

### Quick Recap (from Week 2)

You already know how to stage, commit, and sync in your Codespace. This week, we go deeper: branching, collaboration, and building habits that will carry through your term project.

---

## git2.qmd Tutorial: Time Travel in Codespaces

Deeper Dive: [Time Travel](#)

*Scenario:* We will intentionally break last week’s R script and use Git to rescue it.

## Step 1: Verify Your Baseline

Open your `.qmd` from last week. Make a small, meaningful edit (e.g., improve a comment). Stage (+), write a descriptive commit message, and commit ( ). This is your safety net.

## Step 2: The Disaster

Delete a crucial line of R code in a chunk (e.g., the line that loads the data). **Save** the file (Ctrl/Cmd + S).

## Step 3: Assess the Damage

Open **Source Control** → click the modified file → view the side-by-side **diff** to see exactly what changed. Practice reading the red (removed) and green (added) lines.

## Step 4: The Undo Button

Hover over the file name in the Source Control panel and click **Discard Changes** (the curved arrow) to restore the file to the **last committed** version.



Tip

**Key takeaway:** Saving writes to disk. Committing writes to *history*. Only committed work can be recovered by Git.

---

## git3.qmd Repo Hygiene: `.gitignore` and Relative Paths

Deeper Dive: [Repo Hygiene](#)



## Why This Matters

As your projects grow, you will generate files that should *not* be tracked: temporary Quarto support folders, local caches, `.DS_Store`, and API keys. Rendered files explicitly required by an assessment are different: they are part of the evidence and must be committed.

### `.gitignore` Walkthrough

1. Open (or create) the `.gitignore` file in the root of your repository.
2. Add common patterns:

```
# Quarto temporary/support files
/.quarto/
**/.quarto/
*_files/

# OS junk
.DS_Store
Thumbs.db

# Sensitive
.env
```

**What to do with this block:** copy these patterns into your own `.gitignore` file. This is plain text, not R code — there is nothing to run.

3. Stage and commit with a message such as "Add project `gitignore` rules".

#### Required renders are committed

Do **not** add global `*.html` or `*.pdf` patterns to the course workspace `.gitignore`. If an assignment or milestone explicitly lists a rendered HTML or PDF file, commit it with the source. For practice renders that are not requested, use Source Control to stage only the files named in the

brief.

## Relative Paths

Always use **relative paths** in your code (e.g., `read.csv("data/nhanes.csv")`), never absolute paths (e.g., `read.csv("/home/user/Documents/nhanes.csv")`). This ensures your analysis runs on *any* machine—including your grader's Codespace.



### Warning

**Reproducibility rule:** Your term project must run from start to finish in a *fresh* Codespace. Absolute paths guarantee failure.

---

## git4.qmd Branching: Safe Experimentation

Deeper Dive: [Branching](#)

### Concept

A **branch** is a parallel copy of your project. You can experiment freely on a branch without affecting the stable **main** version. When your experiment works, you **merge** it back.

Think of it like a lab notebook: **main** is your clean, published record. A branch is your scratch pad.

### Tutorial: Create and Merge a Branch in VS Code

#### Step 1: Create a Branch

- Click the branch name in the bottom-left corner of VS Code (it likely says **main**).

- Select **Create new branch...** and name it something descriptive: `add-age-filter`.

### Step 2: Make Changes on the Branch

- Edit your `.qmd` (e.g., add a line of code that filters data by age).
- Stage and commit on this branch.

### Step 3: Switch Back to `main`

- Click the branch name again → select `main`. Notice your recent edits *disappear* from the file. They are safely stored on the other branch.

---

## git5.qmd Pull Requests: The Peer Review Gateway

Deeper Dive: [Pull Requests](#)

### What Is a Pull Request?

A **Pull Request (PR)** is a GitHub feature that says: *“I have changes on a branch. Please review them before they go into `main`.”* PRs are how professional teams and researchers collaborate. You will use them for peer review later in this course.

### Tutorial: Open Your First PR

1. After committing changes to a branch (see `git4.qmd`), click **Sync Changes** to push the branch to GitHub.
2. Open your repository on **github.com**. You should see a banner: *“add-age-filter had recent pushes.”* Click **Compare & pull request**.

3. Write a title and short description explaining what you changed and why.
4. Click **Create pull request**.
5. For now, review the diff on GitHub, then click **Merge pull request** and complete the merge yourself — practice PRs do not need to wait for review. You will practice reviewing each other's pull requests later, in Milestone 4.



Tip

**Collaboration norm:** In your term project, agree with your group that changes go through PRs rather than directly to `main`. This creates a built-in review step.

---

## git6.qmd AI Auditing for Version Control

Deeper Dive: [AI Auditing](#)

Treat the LLM as a **Git Translator** that explains diffs and drafts audit-quality messages.

### git6a.qmd Scenario A: Diff Translation (Pre-Commit Audit)

#### Prompt:

“I am about to commit these changes. Explain what I changed in plain English. Did I delete anything important?”

Use this to spot accidental deletions or unintended edits before you commit.

## git6b.qmd Scenario B: Auto-Scribe Commit Messages

### Prompt:

“Based on this diff, write a one-sentence, professional commit message for my Quarto document.”

Check that the message captures the **intent** (e.g., "Filter patient cohort to age 65 per protocol"), not just the action ("changed line 42").

## git6c.qmd Scenario C: Panic Translator (Error Handling)

### Prompt:

“VS Code in my GitHub Codespace showed this Git error: [paste]. What does it mean in simple terms, and which **buttons in the VS Code UI** should I click to fix it?”

#### Warning

**Strict Guardrail:** Never run random terminal commands suggested by an AI (especially `--force` or `reset --hard`). Prefer the VS Code visual interface, which is safer. If the AI suggests a terminal command, ask it for the GUI equivalent first.

---

## git7.qmd Setting Up Your Group Repository

Deeper Dive: [Setting Up Your Group Repository](#)

This week you form your term project groups (**Milestone 0**). Use this session to establish your collaboration workflow.

## Steps

1. **One member** opens the group project template repository linked from Canvas and creates a fork for the group.
2. That member adds teammates as collaborators on the group fork.
3. All other members accept the collaborator invitation and launch a Codespace from the group fork.
4. As a group, agree on norms:
  - All changes go through **branches** → **PRs** (no direct pushes to **main**).
  - Commit messages should be descriptive (use the AI auto-scribe technique).
  - Keep the repo clean: use **.gitignore**, relative paths, and a clear folder structure.
5. Each member makes a small “hello” commit (e.g., add your name to the **README.md**) and pushes it to verify access.

### Note

**Milestone 0 due Monday (Week 4 deadline posted in Canvas):** Record your group formation on Canvas and ensure everyone can commit to the shared repository.

---

## git8.qmd Knowledge Check

Deeper Dive: [Knowledge Check](#)

1. What is the difference between **saving** a file and **committing** it?
2. What does a red line indicate in a diff?
3. Which button restores the last **committed** version of a file?
4. Why should you use a **.gitignore** file?
5. What is the purpose of a **branch**?

6. How does a **Pull Request** support peer review?
7. Why are relative paths essential for reproducibility?

---

## git9.qmd Practice Exercise: The Full Workflow Drill

Deeper Dive: [Practice Exercise: Full Workflow](#)

1. Open your Codespace from last week.
2. Create a `.gitignore` file with sensible defaults for your project. **Commit**.
3. Create a new branch called `practice-edit`.
4. On the branch, make a small improvement to your R script (e.g., add a clarifying comment or improve a variable name).
5. Use an AI model to draft a clear commit message from your diff. **Commit** on the branch.
6. Push the branch and open a **Pull Request** on GitHub.
7. Review the PR diff on GitHub, then merge it.
8. Back on `main`, intentionally introduce a breaking typo and **Save**.
9. Use **Discard Changes** to recover the last working version.

---

## git10.qmd Quick Reference: Common Git Actions in VS Code

Deeper Dive: [Quick Reference](#)

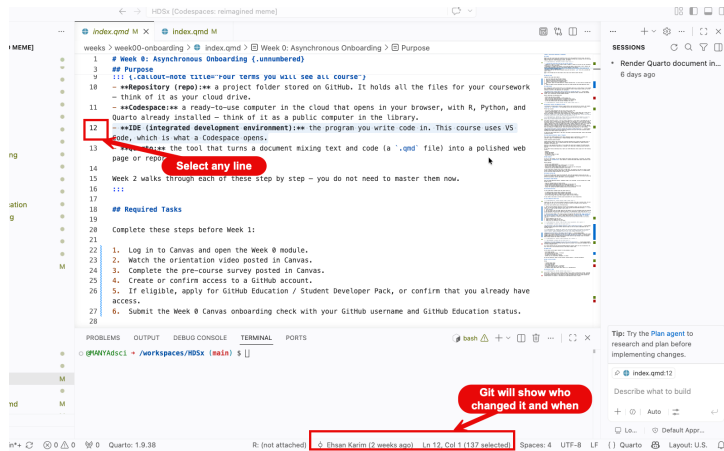
| Action        | How                                   |
|---------------|---------------------------------------|
| Stage changes | Source Control → hover file → click + |

| Action                           | How                                                                |
|----------------------------------|--------------------------------------------------------------------|
| <b>Commit</b>                    | Type message → click                                               |
| <b>View diff</b>                 | Click file name in Source Control                                  |
| <b>Discard uncommitted edits</b> | Hover file → click <b>curved arrow</b> (Discard Changes)           |
| <b>Create a branch</b>           | Click branch name (bottom-left) → <b>Create new branch...</b>      |
| <b>Switch branches</b>           | Click branch name → select target branch                           |
| <b>Merge a branch</b>            | Command Palette → <b>Git: Merge Branch...</b>                      |
| <b>Push / Sync</b>               | Click <b>Sync Changes</b> in status bar                            |
| <b>Open a Pull Request</b>       | Push branch → go to github.com → <b>Compare &amp; pull request</b> |
| <b>Add to .gitignore</b>         | Edit <b>.gitignore</b> in repo root → stage + commit               |



# Git Basics

- **The Audit Trail:** Git records *who* changed *what*.



Selecting any line will show who changed that line and when they did so.

- **The Timeline:** A sequence of safe checkpoints (commits). This shows the reason behind each change and what was change. These checkpoints can always be re-verted back to if the need be.

1



2

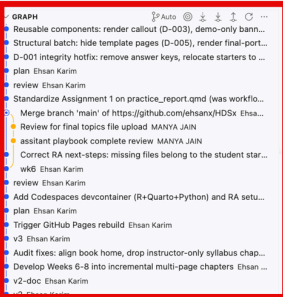


Figure 1: This tree shows every commit (checkpoints)

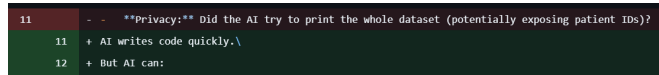
Where do you find this Timeline? In your Codespace, open any file, then look at the bottom of the **Explorer** panel (the left sidebar) for the **Timeline** section. Click any entry to see what the file looked like at that checkpoint. To undo edits you have not committed yet, use **Discard Changes** in Source Control — the [Time Travel tutorial](#) walks through that rescue step. If you ever need

to go back to an older commit, ask the teaching team for help rather than experimenting with reset commands.

- **The Language of Diffs:**

A diff shows what changed between two versions of a file.

- **Red (-):** Removed lines.
- **Green (+):** Added lines.



```
11 - **Privacy:** Did the AI try to print the whole dataset (potentially exposing patient IDs)?
11 + AI writes code quickly.
12 + But AI can:
```

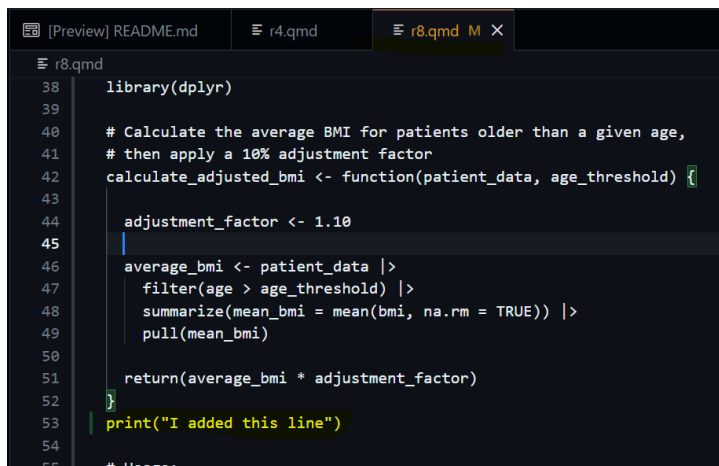
Observing this image, we see that line 11 was replaced, the old data on line 11 is colored red, while the new changes are shown in green.

**Key Rule:** Saving writes to disk on your computer. Committing writes to *history*. Commit saves a checkpoint inside your local Codespace. Push or Sync Changes sends that checkpoint back to GitHub so it is visible in the online repository.

# Time Travel

Follow these steps

1. **Edit:** Make a change to your code, any file in the repo will do, and save the file.



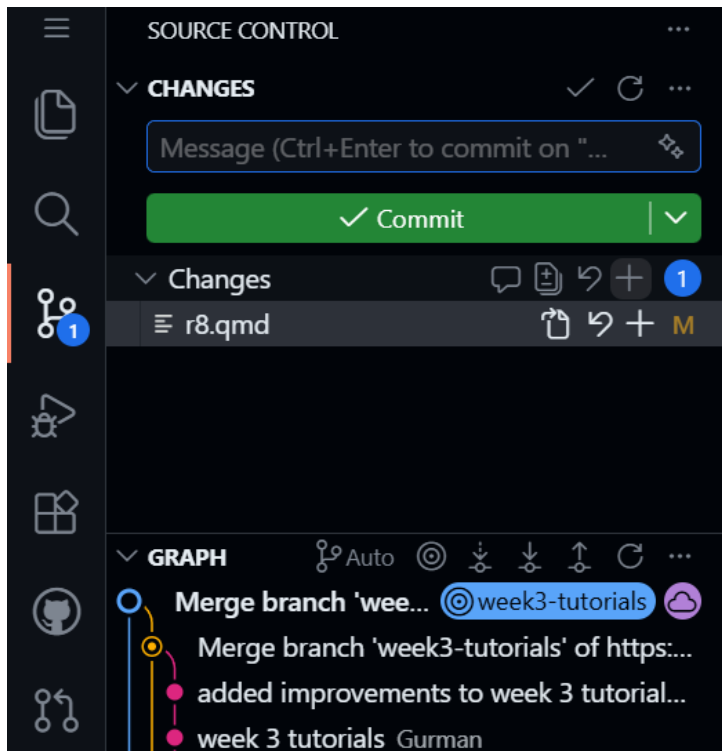
```
38 library(dplyr)
39
40 # Calculate the average BMI for patients older than a given age,
41 # then apply a 10% adjustment factor
42 calculate_adjusted_bmi <- function(patient_data, age_threshold) {
43
44   adjustment_factor <- 1.10
45
46   average_bmi <- patient_data |>
47     filter(age > age_threshold) |>
48     summarize(mean_bmi = mean(bmi, na.rm = TRUE)) |>
49     pull(mean_bmi)
50
51   return(average_bmi * adjustment_factor)
52 }
53 print("I added this line")
54
55 # Usage:
```

Key things to note are that the open file tab has changed colors, this shows that github has acknowledged that the file has been changed.

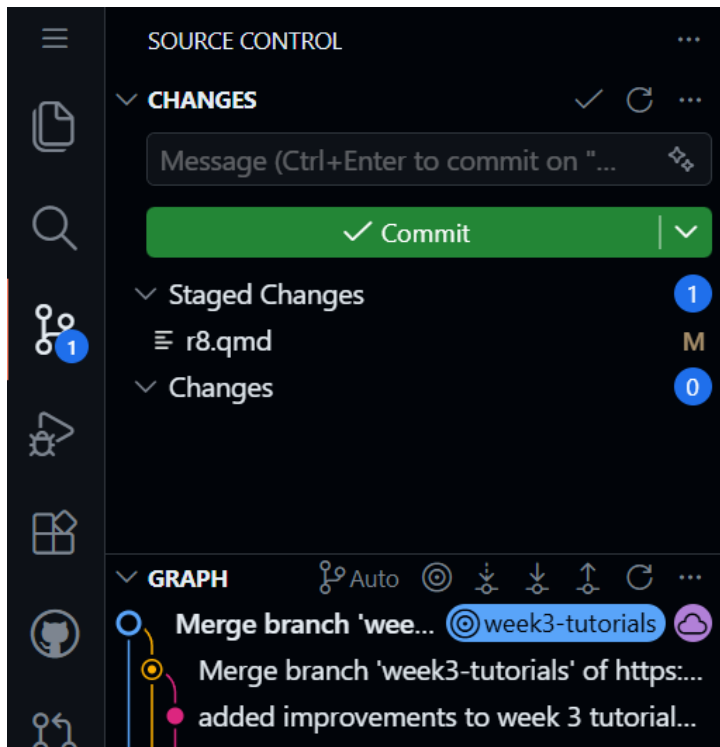
2. **Commit:** Stage and Commit in the Source Control tab.

As we have seen from the previous tutorial, open up the git tab as indicated by the orange-peach highlight. When a file is changed you will be presented with the screen below.

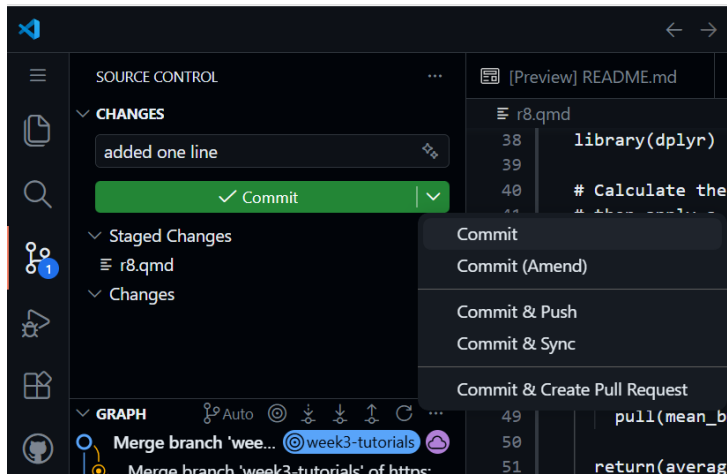
To stage a change, press the + icon.



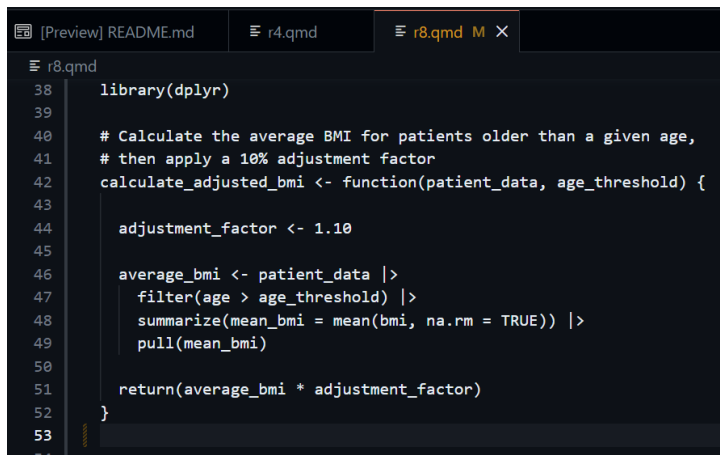
Your menu should look like this now;



Then when you are ready to commit, you can click the big green button! You can also click the down arrow on the big green button to see more options, as well as the commit option. We will discuss those other options later.

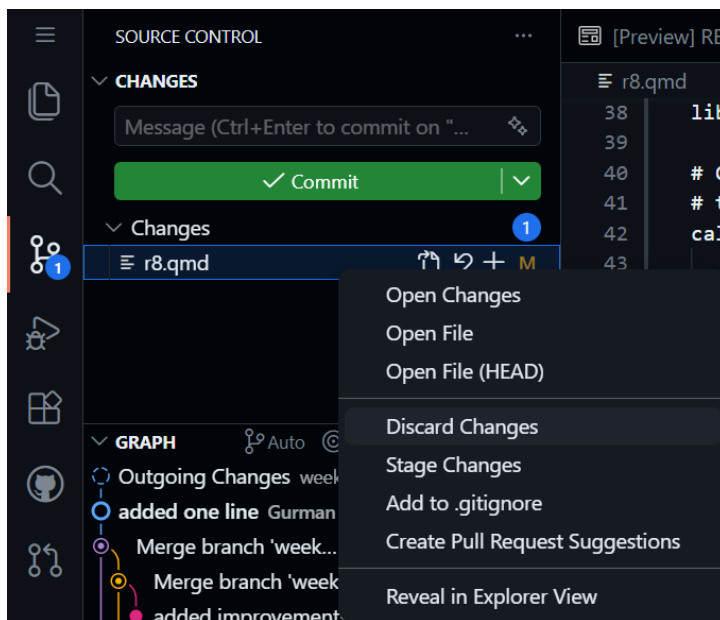


3. **Break:** Delete the code you just added and save.



```
38 library(dplyr)
39
40 # Calculate the average BMI for patients older than a given age,
41 # then apply a 10% adjustment factor
42 calculate_adjusted_bmi <- function(patient_data, age_threshold) {
43
44   adjustment_factor <- 1.10
45
46   average_bmi <- patient_data |>
47     filter(age > age_threshold) |>
48     summarize(mean_bmi = mean(bmi, na.rm = TRUE)) |>
49     pull(mean_bmi)
50
51   return(average_bmi * adjustment_factor)
52 }
53
```

4. **Rescue:** Right-click the file in Source Control again, then click the **Discard Changes** option.



This will revert the file back to what it was before. This can recover data that your computer potentially cannot, it's a very powerful in git and showcases one aspect of how git operates. It is a record of all your work and saved chunks.

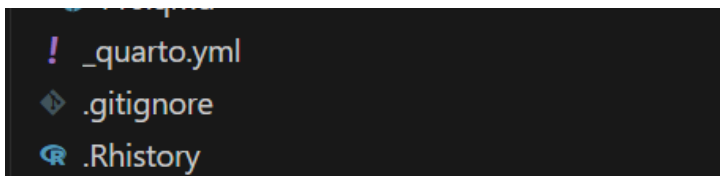
After using the rescue step, go back to the file and confirm that the line you deleted is still present.

# Repo Hygiene

## .gitignore

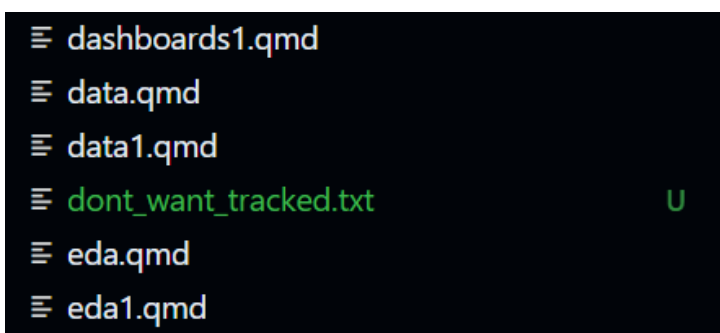
Sometimes there are files you do not want to be tracked by github. This could be anything from files specific to your computer to items like patient data which should not be stored on the cloud unless allowed so by policy.

1. Open the `.gitignore` file in your root directory. If your repository does not already have one, create a file named `.gitignore`: It has to be this exact spelling for git to recognize it.



You will know its correct when it has its own icon as seen above

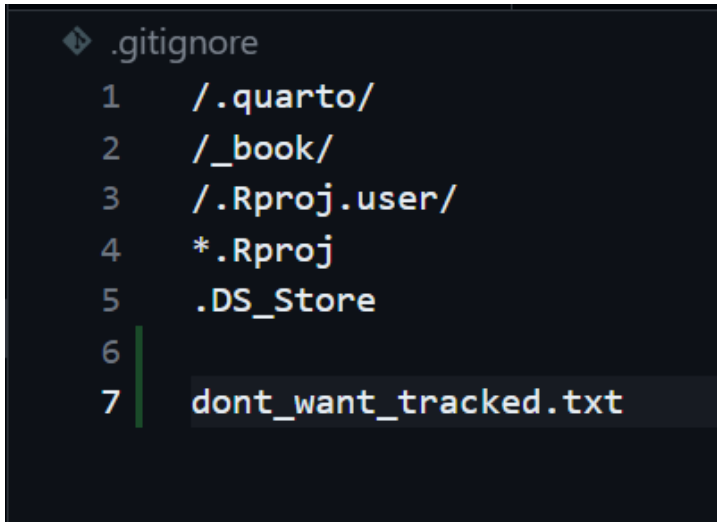
2. Then create a new file, any type will do but for this lesson I will create a text file called `dont_want_tracked`



Notice that the file is green in the Explorer. Git colors new, untracked files green.

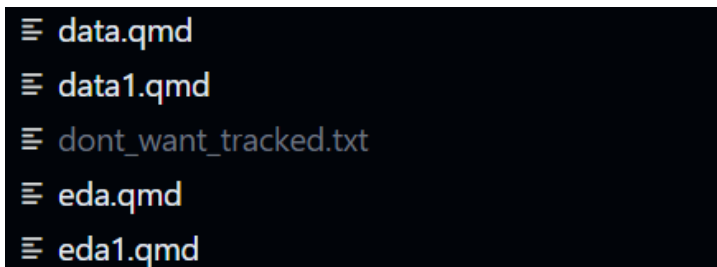


3. Then open the gitignore file and add the name of the file you dont want tracked into it.



```
.gitignore
1  /.quarto/
2  /_book/
3  /.Rproj.user/
4  *.Rproj
5  .DS_Store
6
7  dont_want_tracked.txt
```

4. Now you will see the new file, which was green is now grayed out. This shows that github is no longer tracking it and coloring it green (which it does for new files).



```
≡ data.qmd
≡ data1.qmd
≡ dont_want_tracked.txt
≡ eda.qmd
≡ eda1.qmd
```

This is simple enough right? Whenever in doubt, ask AI how to stop a file from being tracked by git and it will offer some advice. Now stage and commit this like we have done in the previous tutorial.

For the course workspace, ignore machine-specific files, secrets, `.quarto/`, and temporary `*_files/` support directories. Do not globally ignore `*.html` or `*.pdf`: when an assessment explicitly requires a rendered artifact, that file must be committed with its source.

## Relative Paths

A relative path starts from the project folder and works in anyone’s Codespace. An absolute path starts from one person’s computer, such as `/Users/name/...`, and should not be used in submitted work.

For example, if your repository has a folder called `data` containing `nhanes.csv`, the relative path is `data/nhanes.csv` — and it is exactly the same in your Codespace, your teammate’s Codespace, and your grader’s Codespace.

Always use relative paths in your code (e.g., `read.csv("data/nhanes.csv")`), never absolute paths (e.g., `read.csv("/home/user/Documents/nhanes.csv")`). This ensures your analysis runs on any machine—including your grader’s Codespace.

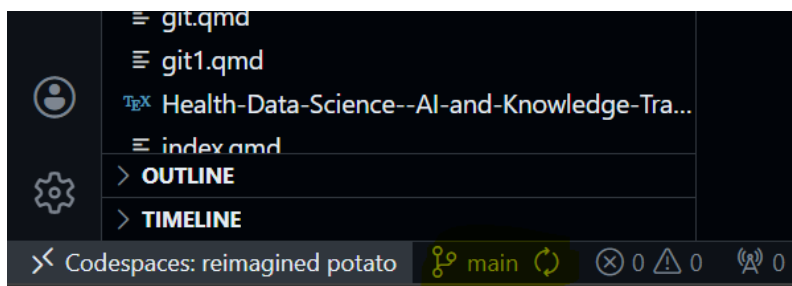
Reproducibility rule: Your term project must run from start to finish in a fresh Codespace. Absolute paths guarantee failure.

# Branching

A branch is a safe parallel version of your code. Think of it as a safe parallel universe for your project: you can experiment freely there without touching the stable `main` version. Before creating a new branch, confirm you are currently on `main` unless the instructions tell you otherwise.

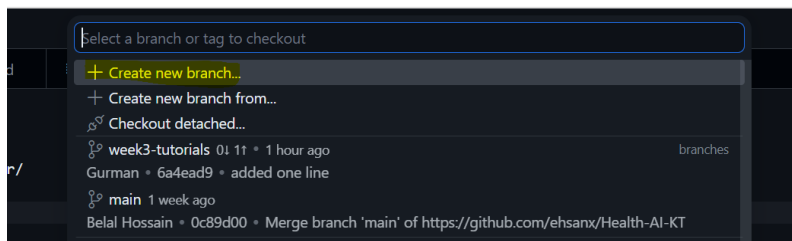
## 1. Create:

Click branch name (bottom-left).

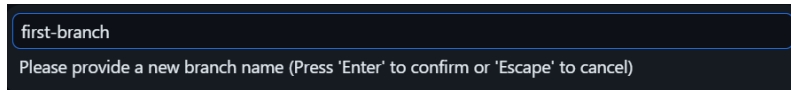


**Safety check:** before you go further, read the name shown in that bottom-left corner. It should say `main`. If it shows something else, click it and select `main` first, so your new branch starts from the stable version.

This will pop up a menu where you have the option to **Create new branch**.

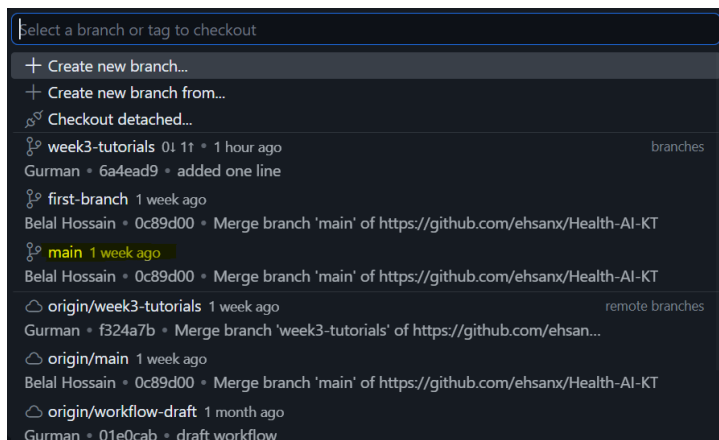


Select this and call it first-branch. Note branch names may not have any spaces and limited special characters, the system will let you know if a name is invalid.



**Safety check:** look at the bottom-left corner again. It should now show **first-branch**. If it still says **main**, the branch was not created — repeat the step before making any edits.

2. **Work:** Make changes, like those in the previous tutorial and commit your changes on your branch.
3. **Switch:** Click the branch button on the bottom left again and once this menu pops up, select **main**. Note that your changes vanish (they are safely in the other branch). If you decide to switch back to your branch, your changes will reappear. This is amazing for multitasking on different features and keeping code clean.

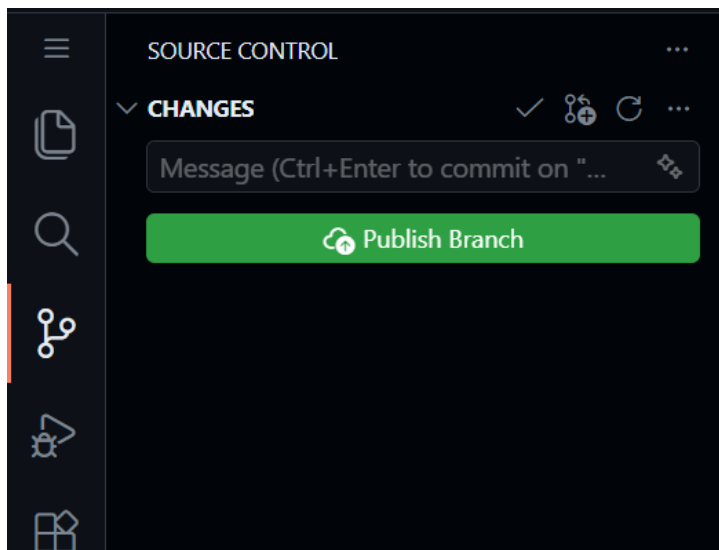


# Pull Requests

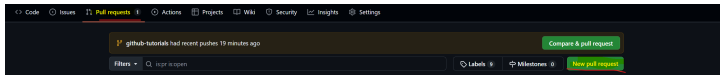
A **Pull Request (PR)** is a GitHub feature that says: *“I have changes on a branch. Please review them before they go into **main**.”* PRs are how professional teams and researchers collaborate. You will use them for peer review later in this course. This is also the best way to merge your own changes into **main**: it forces you to see what has changed, so you can check what you are submitting to the **main** branch — the backbone, or “trunk,” of the tree.

1. **Sync:** Click “Sync Changes” to push your branch to GitHub. This can be blue or green or say publish branch or not. Whichever the case, clicking this button will make everything on your computer update to the online github repo.

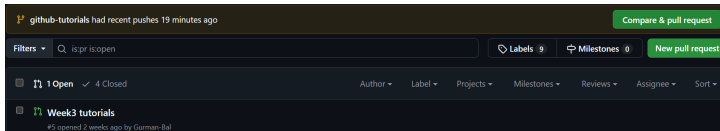
Commit saves a checkpoint inside your local Codespace. Push or Sync Changes sends that checkpoint back to GitHub so it is visible in the online repository.



2. **PR:** Go to your repo on github online and click “Compare & pull request.”

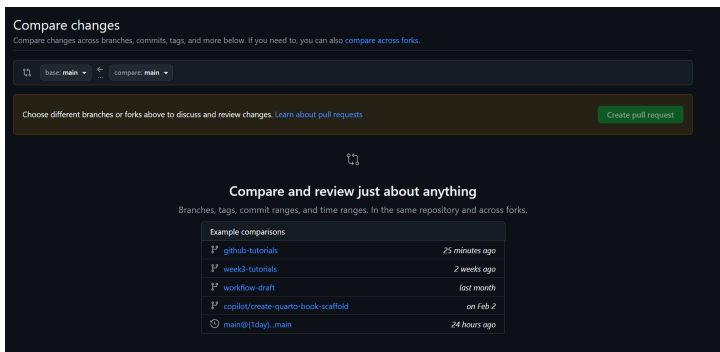


We are then presented with the screen below. Press the big green button “New Pull Request”



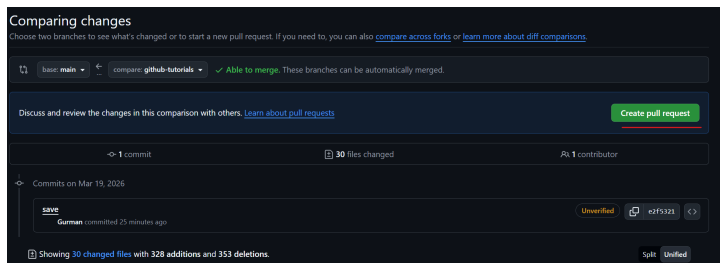
Note: GitHub does not always show this screen. Sometimes clicking **Compare & pull request** takes you straight to the comparison/create screen. If that happens, nothing is wrong — skip ahead to choosing your branches and creating the pull request.

Then you are presented this menu. This shows all the different branches that can be merged into main. Select on the one you want to merge into main.



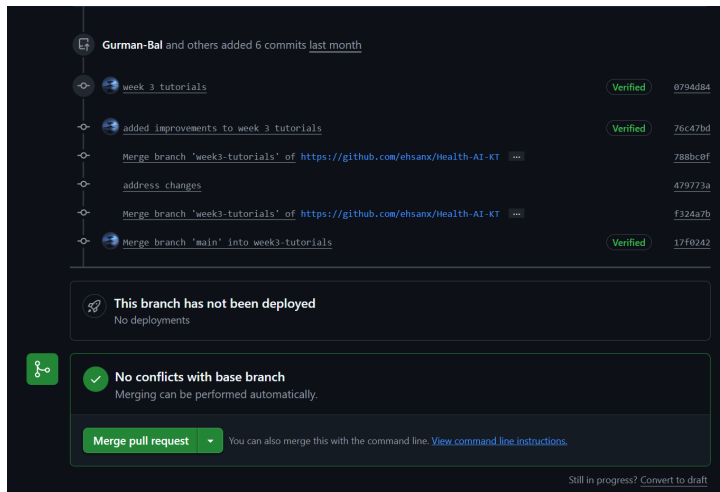
Before creating the Pull Request, check the base repository and base branch carefully. Do not open a PR into the original course repository unless the instructor explicitly asks you to. The **base** is where your changes will land (it should be **main** in *your own* repository); the **compare** (also called *head*) is the branch holding your changes.

Once you select the one, you can make a pull request! Select this.



3. **Merge:** Review the diff, then click “Merge pull request.”  
If there are merge conflicts, ask AI the best way to resolve them.

For this week’s practice pull requests, merge your own PR right away — you do not need to wait for a teammate or TA to review it. You will practice reviewing each other’s pull requests later, in Milestone 4.



# AI Auditing

Use these prompts with your AI assistant:

- **Audit:** “I am about to commit these changes. Explain what I changed in plain English. Did I delete anything important?”

Use this to spot accidental deletions or unintended edits before you commit.

- **Scribe:** “Based on this diff, write a one-sentence, professional commit message for my Quarto document.”

Check that the message captures the **intent** (e.g., “Filter patient cohort to age 65 per protocol”), not just the action (“changed line 42”).

- **Panic:** “VS Code shows this error: [PASTE ERROR]. What does it mean in simple terms and what buttons should I click?”

Never run random terminal commands suggested by an AI (especially `--force` or `reset --hard`). Prefer the VS Code visual interface, which is safer. If the AI suggests a terminal command, ask it for the GUI equivalent first.

## From Prompts to Submission Files

The three prompts above produce text you will reuse as evidence in [Assignment 3](#). After using AI to explain your diff or draft a commit message, copy the useful part into `git-audit-note.md`. For `pull-request-evidence.md`, paste the PR link or screenshot filename and one sentence explaining what the PR shows.

For example, `git-audit-note.md` could look like:



```
1  ## Diff summary (AI-assisted, verified by me)
2  Changed one sentence in README.md to clarify where the data file lives.
3  No code lines were added or removed.
4
5  ## .gitignore check
6  Confirmed secrets and temporary files are excluded. Required HTML remains committable.
7
8  ## AI-use sentence
9  I used Copilot Chat to summarize the diff; I re-read the diff myself
10 and confirmed the summary is accurate.
11
12 ## MO readiness
13 All group members can open the shared repo and have committed at least once.
```

And pull-request-evidence.md could look like:

```
1  PR link: https://github.com/your-username/your-repo/pull/1
2  Branch: docs/git-audit-note
3  Final commit hash: a1b2c3d
4  What the PR shows: a documentation-only edit to one README sentence.
```

Both files go in assignments/assignment03-git/submission/.

# Setting Up Your Group Repository

This page supports **Milestone 0 (group formation)** — see the [M0 README](#) for the full deliverables. Your group does not create a blank repository from scratch: you start from the group project template.

1. **Fork the template:** One member opens the group project template repository (linked from Canvas) and creates a fork for the group. A fork is the group's own copy of the template. If you cannot access Canvas, ask the teaching team for the template link.
2. **Invite:** That member adds the other teammates as collaborators on the group fork; others accept the invite.
3. **Clone:** Everyone launches their own Codespace from the group fork.
4. **Norms:** Agree: “No direct pushes to `main`” — all changes go through branches and pull requests.
5. **Test:** Everyone makes a small `README.md` update (for example, add your name) to verify access.

# Knowledge Check

1. **Saving vs. Committing:** Saving = disk; Committing = history.
2. **Red line:** Deletion.
3. **Restore:** Discard Changes button.
4. **.gitignore:** Prevents tracking junk/secrets.
5. **Branch:** Parallel experimental workspace.
6. **PR:** A peer review workflow for merging code.
7. **Relative Paths:** Ensures the project runs on any machine.

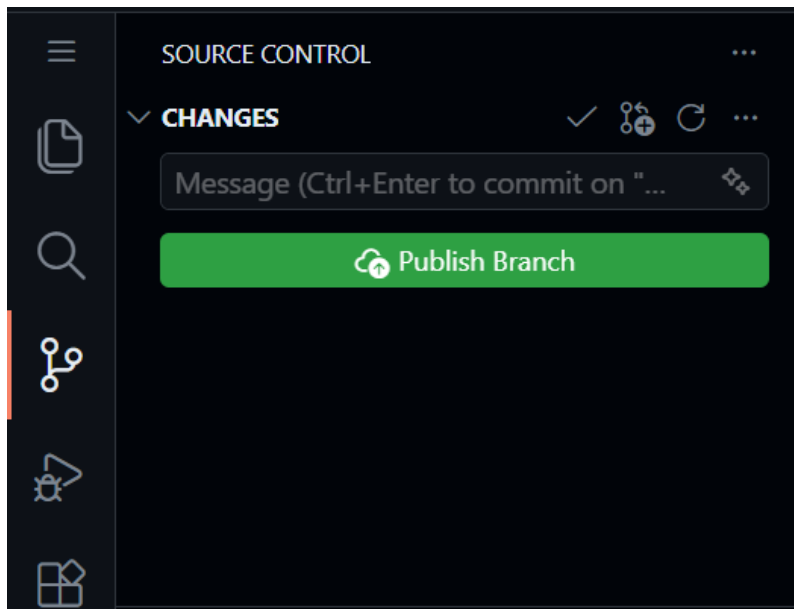
## Practice Exercise: Full Workflow

| Step | Task           | Done                     |
|------|----------------|--------------------------|
| 1    | Edit Code      | <input type="checkbox"/> |
| 2    | Create Branch  | <input type="checkbox"/> |
| 3    | AI Draft Msg   | <input type="checkbox"/> |
| 4    | Stage & Commit | <input type="checkbox"/> |
| 5    | Push & PR      | <input type="checkbox"/> |
| 6    | Merge          | <input type="checkbox"/> |

## Quick Reference

| Action  | How-To                                                                     |
|---------|----------------------------------------------------------------------------|
| Stage   | Source Control (+)                                                         |
| Commit  | Type msg +                                                                 |
| View    | Click file in Source Control                                               |
| Diff    |                                                                            |
| Discard | Right-click file in Source Control → <b>Discard Changes</b>                |
| Branch  | Bottom-left status bar                                                     |
| Sync    | <b>Sync Changes</b> button in Source Control<br>(appears after you commit) |

**Where is Sync Changes?** After you commit, the **Sync Changes** button appears in the Source Control panel (it may be blue or green, or say *Publish Branch*). Clicking it uploads your commits to GitHub so they are visible in the online repository.



# **Week 5: Polyglot Awareness & R Deepening**

# Polyglot Awareness & R Deepening

## Overview

Week 5 is about becoming a careful bilingual reader of data analysis code. R remains the core language for required course work. Python is introduced so you can read short **pandas** examples, translate small tasks with AI assistance, and verify that a translated result still answers the same question.

Polyglot awareness means being able to recognize and compare similar data steps in more than one programming language. R remains the required language; Python is used here for comparison and translation practice.

The week has two linked goals:

- deepen R fluency with **dplyr**, functions, and clear summaries;
- compare a small R result with a Python/pandas translation using explicit parity checks.

## Learning Objectives

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

- explain when R is the better default and when Python may be useful;
- write a clear R **dplyr** pipeline for a descriptive health-data summary;
- refactor repeated R code into a small function;
- create and run a Python notebook in Codespaces;
- load and inspect a CSV with **pandas**;



- identify common R-to-Python translation traps;
- use AI one step at a time while auditing each translated output;
- document approved Python dependencies in the workspace-root `.devcontainer/requirements.txt`;
- use Git to commit a clean translation workflow.

## Connection

Week 4 made small changes reviewable through Git. This week uses that review mindset inside the analysis itself: first build a trusted R result, then translate one step at a time and compare. Next week you will turn those checked summaries into clearer visual explanations.

## Case Study Data Analysis

**What to do with this code:** read it. The code block below runs automatically when this page is built — you do not need to run or change it here. You will write your own version in the studio.

The **NHANES Health Equity data spine** now supports code-reading, EDA, visualization, and reproducibility checks. Use the cached CSV/RDS for routine class work; a CDC retrieval script exists for advanced users.

- Cached RDS: [examples/nhanes-equity/data/nhanes\\_equity\\_v6.rds](#)
- CSV snapshot: [examples/nhanes-equity/data/nhanes\\_equity\\_v6.csv](#)
- Case-study README: [examples/nhanes-equity/README.md](#)

The default classroom path is to use the cached data so the analysis work is reproducible without internet access.

**What to do here:** you may run the code below — it loads the cached CSV and prints two small summaries; reading it carefully is enough unless your week's page asks for more.

```

1 library(tidyverse) # AI-EDIT(2026-06-23): tidyverse-default - consolidated core library() call
2
3 candidate_paths <- c(
4   "examples/nhanes-equity/data/nhanes_equity_v6.csv",
5   "../examples/nhanes-equity/data/nhanes_equity_v6.csv"
6 )
7
8 nhanes_path <- candidate_paths[file.exists(candidate_paths)][1]
9 if (is.na(nhanes_path)) {
10   stop("Could not find examples/nhanes-equity/data/nhanes_equity_v6.csv")
11 }
12
13 nhanes_analysis <- read_csv(nhanes_path, show_col_types = FALSE)
14
15 nhanes_analysis |>
16   summarise(
17     rows = n(),
18     missing_bmi = sum(is.na(BMI)),
19     missing_income = sum(is.na(IncomeGroup))
20   )
21 #> # A tibble: 1 x 3
22 #>   rows missing_bmi missing_income
23 #>   <int>      <int>      <int>
24 #> 1 105626      9356      9415
25
26 nhanes_analysis |>
27   filter(!is.na(BMI), !is.na(IncomeGroup), Age >= 20, Age <= 80) |>
28   group_by(IncomeGroup) |>
29   summarise(
30     n = n(),
31     mean_bmi = round(mean(BMI), 1),
32     .groups = "drop"
33   )
34 #> # A tibble: 3 x 3
35 #>   IncomeGroup      n mean_bmi
36 #>   <chr>      <int>   <dbl>
37 #> 1 High Income (>3.5) 16330    28.6
38 #> 2 Low Income (<1.3) 15293    29.4
39 #> 3 Middle Income    19228    29.3

```

## R Deepening Studio

Start in R. Use the NHANES case-study CSV:

```
1 examples/nhanes-equity/data/nhanes_equity_v6.csv
```

## Where To Write Your R Code

Work directly in your personal workspace. Create `assignments/assignment04-polyglot/submission/polyglot-` and put the R studio code in R chunks there. This avoids creating a second `week5.R` file that would only need to be copied later.

## R Pipeline and R Function

The studio code — a `dplyr` pipeline and a small reusable function — lives in one place so there is a single copy to keep correct: work through Parts A and B of [R Deepening and Parity Check](#). Type or paste each block into R chunks in `polyglot-parity.qmd` and run it.

Before translating, compare the pipeline output and the function output. They should have the same grouping labels and the same number of rows.

## In-Class Activity: R First, Python Second

Work in pairs. One partner reads the R output; the other reads the Python output.

A parity check means confirming that the R result and Python result match on the important parts: row counts, column counts, grouping labels, missing values, and rounded summary values.

R steps happen in `polyglot-parity.qmd`. Python steps happen in `translation.ipynb` in the same A4 submission folder (see [Python Environment](#)). `pandas` comes preinstalled in the course Codespace.

1. Load the NHANES CSV in R and Python.
2. Record raw row count, column count, and missing BMI count in both languages.
3. Produce mean and SD of BMI by `IncomeGroup` and `Gender` in R.
4. Translate the same task to Python/pandas one step at a time.
5. Fill in a parity table comparing row counts, grouping labels, and rounded summary values.
6. If something does not match, write the likely reason before changing code.

If the R and Python results do not match, follow the 15-minute rule: spend up to 15 minutes debugging on your own — recheck the [translation traps](#) and write down what you tried and your best guess at the cause. After 15 minutes, ask your partner, a TA, or the instructor for help, and show your notes.

Student output: a parity-check note with the R pipeline, R function, Python translation, parity table, and a short AI-use note.

## Page Map

Use the child pages in this order:

1. [Python vs. R](#)
2. [Python Environment](#)
3. [Packages and Reproducibility](#)
4. [Translation Traps](#)
5. [Load and Inspect Data](#)
6. [Python Tracebacks](#)
7. [One Step, One Cell](#)
8. [Mixing R and Python](#)
9. [Reusable Python Scripts](#)
10. [Git Progress](#)
11. [Proposal Readiness](#)
12. [Knowledge Check](#)
13. [Bilingual Translation](#)
14. [R To Python Cheatsheet](#)

For the explicit R mastery exercises — the worked pipeline, the reusable function, and the extra `dplyr` and function practice — use [R Deepening and Parity Check](#).

## Assignment 4

Complete [Assignment 4: Polyglot Parity](#).

Submit a repository link on Canvas after committing:

- `assignments/assignment04-polyglot/submission/polyglot-parity.qmd;`
- `assignments/assignment04-polyglot/submission/translation.ipynb;`
- `assignments/assignment04-polyglot/submission/helpers.py.`

## Knowledge Check

Before leaving Week 5, answer these:

1. What is one analysis task you would keep in R?
2. What is one task where Python may be useful?
3. What are three parity checks you ran?
4. What did AI help translate, and how did you verify it?
5. What changed in your Git history from the beginning to the end of the week?

# R Deepening and Parity Check

Week 5 is polyglot-aware, not Python-first. Your strongest analysis path remains R and the tidyverse. Python enters as a comparison language: it helps you read code from other sources, ask better AI translation questions, and verify that a translated result still means the same thing.

A parity check means confirming that the R result and Python result match on the important parts: row counts, column counts, grouping labels, missing values, and rounded summary values.

## Learning Goals

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

- write a clear R `dplyr` pipeline for a descriptive health-data summary;
- refactor a repeated summary into a small R function;
- translate the same task to Python/pandas one step at a time;
- compare outputs across languages using row counts, grouping levels, and summary values;
- document any mismatch before deciding which code to change.

## Dataset

Use the NHANES case-study snapshot:

```
1 examples/nhanes-equity/data/nhanes_equity_v6.csv
```

## Part A: R Pipeline

Write Parts A and B as R chunks in `assignments/assignment04-polyglot/submission/polyglot-parity.qmd` in your personal workspace. Work in the assessed file from the start rather than creating a second script to move later.

Start with R. The goal is a descriptive, unweighted summary of BMI by income group and gender.

```
1 library(tidyverse)
2
3 nhanes <- read_csv("examples/nhanes-equity/data/nhanes_equity_v6.csv")
4
5 r_summary <- nhanes |>
6   filter(!is.na(BMI), !is.na(IncomeGroup), !is.na(Gender)) |>
7   group_by(IncomeGroup, Gender) |>
8   summarise(
9     n = n(),
10    mean_bmi = round(mean(BMI), 2),
11    sd_bmi = round(sd(BMI), 2),
12    .groups = "drop"
13  ) |>
14  arrange(IncomeGroup, Gender)
15
16 r_summary
```

## Part B: R Function

Refactor the summary so the value and grouping variables are easy to change later.

```
1 summarise_mean_sd <- function(data, value, group1, group2) {
2   data |>
3     filter(!is.na({{ value }}), !is.na({{ group1 }}), !is.na({{ group2 }})) |>
4     group_by({{ group1 }}, {{ group2 }}) |>
5     summarise(
6       n = n(),
7       mean_value = round(mean({{ value }}), 2),
8       sd_value = round(sd({{ value }}), 2),
```

```

9       .groups = "drop"
10     ) |>
11     arrange({{ group1 }}, {{ group2 }})
12   }
13
14   r_function_summary <- summarise_mean_sd(nhanes, BMI, IncomeGroup, Gender)
15   r_function_summary

```

Before moving to Python, check that the function output has the same number of rows and the same grouping labels as the original pipeline.

## More R Mastery Exercises

The pipeline and function above are the worked example. Complete at least two of these short exercises in R chunks in `assignments/assignment04-polyglot/submission/polyglot-parity.qmd` before moving to Python:

1. **Swap the value column.** Run `summarise_mean_sd(nhanes, Waist, IncomeGroup, Gender)`. Check that the grouping labels match the BMI summary. Did the row count change? Why might it?
2. **Swap a grouping column.** Run `summarise_mean_sd(nhanes, BMI, Race, Gender)`. How many grouped rows do you get now, and why?
3. **Extend the pipeline.** Add `mean_age = round(mean(Age), 1)` inside `summarise()` in the Part A pipeline, then rerun it. Confirm the number of rows did not change — adding a summary column should never change the grouping.

## AI Audit Prompt

Use this prompt only after you have a working R result:

Translate this R `dplyr` pipeline into Python/pandas one step at a time. For each step, explain the equivalent Python operation, the output I should inspect,



and the parity check I should run against my R result. Do not combine steps or add new analysis choices.

## Part C: Python Translation

Create the Python translation at `assignments/assignment04-polyglot/submission/translation.ipynb` in your personal workspace — the same notebook you set up in [Python Environment](#) and use in [Bilingual Translation](#). Attempt each step yourself first with the [cheat sheet](#); use AI to check or fix one step at a time.

Translate one step at a time. Do not ask AI for a full script until you have verified each cell.

```
1 import pandas as pd
2
3 nhanes = pd.read_csv("../../examples/nhanes-equity/data/nhanes_equity_v6.csv")
4
5 py_summary = (
6     nhanes
7     .dropna(subset=["BMI", "IncomeGroup", "Gender"])
8     .groupby(["IncomeGroup", "Gender"], as_index=False)
9     .agg(
10         n=("BMI", "size"),
11         mean_bmi=("BMI", "mean"),
12         sd_bmi=("BMI", "std")
13     )
14     .sort_values(["IncomeGroup", "Gender"])
15 )
16
17 py_summary["mean_bmi"] = py_summary["mean_bmi"].round(2)
18 py_summary["sd_bmi"] = py_summary["sd_bmi"].round(2)
19
20 py_summary
```

## Part D: Parity Audit

Record the following checks in a short Quarto note:

| Check                                  | R command                                                                           | Python command                                                               | What to record              |
|----------------------------------------|-------------------------------------------------------------------------------------|------------------------------------------------------------------------------|-----------------------------|
| Raw row count                          | <code>nrow(nhanes)</code>                                                           | <code>len(nhanes)</code>                                                     | Same number of rows         |
| Column names                           | <code>names(nhanes)</code>                                                          | <code>nhanes.columns.tolist()</code>                                         | Same as expected variables  |
| Analysis rows after missingness filter | <code>nrow(filter(nhanes, !is.na(BMI), !is.na(IncomeGroup), !is.na(Gender)))</code> | <code>nhanes.dropna(subset=["BMI", "IncomeGroup", "Gender"]).shape[0]</code> | Same row count              |
| Group count                            | <code>nrow(r_summary)</code>                                                        | <code>len(py_summary)</code>                                                 | Same number of grouped rows |
| Summary values                         | <code>inspect(mean_bmi and sd_bmi)</code>                                           | <code>inspect(mean_bmi and sd_bmi)</code>                                    | Values match after rounding |

### **i** When results do not match: the 15-minute rule

Debug on your own for up to 15 minutes first: recheck the [translation traps](#), rerun the checks above, and write down what you tried and your best guess at the cause. After 15 minutes, ask for help — and bring your notes. Documented debugging attempts are part of the expected workflow, not a sign of failure.

## Student Output

Submit a short note with:

- the R pipeline;
- the R function;
- the Python translation;
- a parity table showing the checks above;
- two sentences explaining any mismatch or confirming that the outputs agree.

These are the Assignment 4 submission files; do not also save copies under a **weeks/** folder in your personal workspace.

# Python vs. R

## The Course Position

R remains the core language for required analysis in this course. Python is introduced so you can read short Python examples, translate small workflows with AI assistance, and recognize when Python may be useful in a health data project.

You are not choosing one language forever. You are learning how to move carefully between them.

## Strengths At A Glance

Think in terms of your goal, not package names:

| If your goal is...                                                | Better default | Why                                                                                                               |
|-------------------------------------------------------------------|----------------|-------------------------------------------------------------------------------------------------------------------|
| Summarise a health dataset and write it up (like our NHANES work) | R              | The <code>dplyr</code> steps you already know plus Quarto take you from raw CSV to a finished report in one place |
| Make clear charts for a report or slides                          | R              | <code>ggplot2</code> charts drop straight into your Quarto documents                                              |

| If your goal is...                                                   | Better default | Why                                                                                                   |
|----------------------------------------------------------------------|----------------|-------------------------------------------------------------------------------------------------------|
| Read example code you found online or got from another team          | Either         | A lot of shared data code is written in Python; this week teaches you to read it without abandoning R |
| Write a small reusable helper function                               | Either         | Both languages do this well — you practice both in Week 5                                             |
| Build machine-learning models or software that runs behind a website | Python         | Most of that tooling is built Python-first; it is beyond the scope of this course                     |

When in doubt in this course, use R. Reach for Python only when an instruction, or your project, clearly calls for it.

## Notebooks vs. Scripts

- A notebook (`.ipynb`) is useful for exploration, teaching, and step-by-step AI translation checks.
- A script (`.py`) is useful for reusable functions and automation.
- A Quarto document (`.qmd`) is best when your analysis needs narrative, code, figures, citations, and a rendered final product.

## Check Your Understanding

Write two sentences:

1. One task where you would stay in R.
2. One task where Python might be worth considering.

# Python Environment

## Create A Notebook

In your GitHub Codespace:

Do this in your personal `HDSx-workspace` fork — the same Codespace you use for assignments, not the read-only course-book repository or a blank repo.

1. In the Explorer, select `assignments/assignment04-polyglot/submission/`.
2. Right-click that folder, choose **New File...**, and name the file `translation.ipynb`. Do not save it at the repository root.
3. Select the recommended Python kernel when VS Code asks.
4. Before running anything, let the Codespace finish setting up. If VS Code shows an “Updating extensions” or “Installing extensions” message in the bottom-right corner, wait for it to finish — running the notebook before the Python and Jupyter extensions are up to date can make the first cell very slow to start. You can confirm in the Extensions panel that the Python and Jupyter extensions are installed and not showing a pending update.
5. Run this first cell by clicking the play button to the left of the cell (or pressing `Shift+Enter`). The output appears directly below the cell:

```
1 print("Hello from Python. My Week 5 notebook is running.")
```

## Use The Variables Pane

The Variables pane helps you audit Python objects the way the Environment pane helps you audit R objects.

Run this cell:

```
1 age = 25
2 course = "HDSx"
```

Then open the notebook Variables pane and check that both variables appear.

## Load A Tiny DataFrame

`pandas` comes preinstalled in the course Codespace, so the import below just works — you do not need to install anything.

```
1 import pandas as pd
2
3 demo = pd.DataFrame({
4     "group": ["A", "A", "B"],
5     "value": [10, 15, 20]
6 })
7
8 demo
```

Double-click `demo` in the Variables pane to open the Data Viewer. This is where you will inspect row counts, column names, and translated outputs later in the week.

## Common Fixes

| Problem                    | First thing to try                                                |
|----------------------------|-------------------------------------------------------------------|
| No kernel appears          | Wait for the Codespace to finish starting, then reload the window |
| Variables pane is missing  | Check that the Jupyter extension is enabled                       |
| Notebook cell will not run | Select a Python kernel from the notebook toolbar                  |

| Problem                          | First thing to try                                                                                         |
|----------------------------------|------------------------------------------------------------------------------------------------------------|
| First cell is very slow to start | Wait for the Python and Jupyter extensions to finish updating (check the Extensions panel), then try again |
| Package import fails             | Check the course baseline in <code>.devcontainer/requirements.txt</code> ; ask before adding a package     |



# Packages and Reproducibility

## R `library()` vs. Python `import`

Both languages require you to load packages before using them.

| R                                | Python                                 | Purpose                           |
|----------------------------------|----------------------------------------|-----------------------------------|
| <code>library(readr)</code>      | <code>import pandas<br/>as pd</code>   | read CSV files                    |
| <code>library(dplyr)</code>      | <code>import pandas<br/>as pd</code>   | filter, group, and summarise data |
| <code>library(ggplot2)</code>    | <code>import seaborn<br/>as sns</code> | create plots                      |
| <code>library(reticulate)</code> | not applicable                         | call Python from R                |

## Why `requirements.txt` Matters

Python projects record package dependencies in `requirements.txt`. This is the Python-side equivalent of saying, “Here is what another person must install before my code will run.”

The course Codespace already provides `pandas` and the notebook tools, so you should not need to install anything for this week’s required work. Only edit `.devcontainer/requirements.txt` if an additional Python package has been approved.

In the course Codespace this file lives at `.devcontainer/requirements.txt` (at the top level of your project). The file itself is simple — one package name per line. For example, a project that depends only on `pandas` would contain:

```
1 pandas
```

If you also create a plot in Python with packages that are not preinstalled, add:

```
1 seaborn
2 matplotlib
```

## Course Rule

If you rely on an approved additional Python package, record it in the root `.devcontainer/requirements.txt` and in your dependency note. Do not create a second submission-folder requirements file.

## R Connection

R reproducibility uses the same devcontainer baseline plus explicit `library()` calls and dependency notes. Week 5 is a bridge: you should be able to explain where dependencies are recorded in both languages.

# Translation Traps

AI translation is useful, but it often hides small language differences. These are the traps to check first.

## Trap 1: Indexing

R starts counting at 1. Python starts counting at 0.

```
1 values <- c(10, 20, 30)
2 values[1]
```

```
1 values = [10, 20, 30]
2 values[0]
```

Python slices are end-exclusive:

```
1 values[0:2]
```

This returns the first two values, not the first three.

Walk through it position by position. Python labels the items position 0 (10), position 1 (20), and position 2 (30). The slice `values[0:2]` starts at position 0 and stops *just before* position 2 — that is what “end exclusive” means: the end position itself is left out. So you get 10 and 20, but not 30. The matching R code is `values[1:2]`, because R starts counting at 1 and includes both ends. This off-by-one difference is one of the most common AI translation bugs, so always count a small example by hand.

## Trap 2: Indentation

R uses braces to mark code blocks. Python uses indentation.

```
1 if (age >= 20) {  
2   "adult"  
3 } else {  
4   "not in adult cohort"  
5 }
```

```
1 if age >= 20:  
2     "adult"  
3 else:  
4     "not in adult cohort"
```

If Python indentation is inconsistent, the code may fail or run with a different meaning.

## Trap 3: Assignment vs. Comparison

| Meaning            | R                       | Python                                            |
|--------------------|-------------------------|---------------------------------------------------|
| assign a value     | <code>x &lt;- 10</code> | <code>x = 10</code>                               |
| compare values     | <code>x == 10</code>    | <code>x == 10</code>                              |
| missing/null check | <code>is.na(x)</code>   | <code>pd.isna(x)</code> or <code>x is None</code> |

## Trap 4: Missing Values

R and Python both have missing values, but they are handled by different functions. Always check missingness before comparing summaries.

```
1 sum(is.na(nhanes$BMI))
```

```
1 nhanes["BMI"].isna().sum()
```

## **Audit Habit**

When a translated result differs, check indexing, indentation, assignment/comparison, missingness handling, and grouping variables before assuming one language is “wrong.”

# Load and Inspect Data

**What to do on this page:** run the Python code below as written, in `translation.ipynb`, one cell at a time — click the play button beside each cell or press **Shift+Enter**, and the output appears below the cell. You are not asked to translate this Python into R or to modify it. Instead, compare each output with the matching R command in the table at the bottom; you built the R side in [R Deepening and Parity Check](#).

## Load The NHANES Snapshot

In `translation.ipynb`, load the same CSV used in the R example:

```
1 import pandas as pd
2
3 nhanes = pd.read_csv("../../examples/nhanes-equity/data/nhanes_equity_v6.csv")
```

## Inspect Before Summarising

```
1 nhanes.head()

1 nhanes.shape

1 nhanes.columns.tolist()

1 nhanes[["BMI", "IncomeGroup", "Gender"]].isna().sum()
```

## Compare To R

| Question                     | R                                    | Python                                  |
|------------------------------|--------------------------------------|-----------------------------------------|
| How many rows?               | <code>nrow(nhanes)</code>            | <code>len(nhanes)</code>                |
| How many columns?            | <code>ncol(nhanes)</code>            | <code>nhanes.shape[1]</code>            |
| What are the column names?   | <code>names(nhanes)</code>           | <code>nhanes.columns.tolist()</code>    |
| How many missing BMI values? | <code>sum(is.na(nhanes\$BMI))</code> | <code>nhanes["BMI"].isna().sum()</code> |

## Student Check

Before asking AI for any transformation, write down the raw row count, column count, and missingness count for BMI.

# Python Tracebacks

## The Bottom Line Is Usually At The Bottom

Python errors often print several lines called a traceback. Read from the bottom upward:

```
1 FileNotFoundError: [Errno 2] No such file or directory: 'Data/patients.csv'
```

This tells you the immediate problem: Python cannot find the file at the path you gave it.

## Common Week 5 Errors

| Error                            | Likely cause                                                        | First fix                                                                        |
|----------------------------------|---------------------------------------------------------------------|----------------------------------------------------------------------------------|
| <code>FileNotFoundError</code>   | path is wrong, file is in another folder, or capitalization differs | check the relative path from the notebook location                               |
| <code>ModuleNotFoundError</code> | package is missing                                                  | add the package to <code>requirements.txt</code> and install it in the Codespace |
| <code>KeyError</code>            | column name is misspelled or not present                            | run <code>nhanes.columns.tolist()</code>                                         |
| <code>IndentationError</code>    | Python block spacing is inconsistent                                | re-indent the block with four spaces                                             |
| <code>NameError</code>           | object has not been created in the current kernel                   | rerun earlier cells in order                                                     |



## AI Debugging Prompt

Read this Python traceback from the bottom up. Identify the error type, the exact object or path causing the issue, and the smallest code change that would fix it. Do not rewrite the full analysis.

## Student Check

When you fix an error, add one sentence to your notebook explaining what failed and how you knew the fix worked.

# One Step, One Cell

## Why This Rule Exists

Large AI-generated code blocks are hard to audit. A single hidden mistake in loading, filtering, grouping, or missingness handling can cascade into a polished but incorrect result.

Use one cell per analytic step:

1. load data;
2. inspect rows and columns;
3. filter missing values;
4. group and summarise;
5. compare with the R result.

## Good AI Prompt

Translate only the data-loading step from R to Python/pandas. Tell me what output I should inspect before moving to the next step.

## Risky AI Prompt

Translate my full analysis, make the plots, and write the interpretation.

The risky prompt gives up too much control. It may change the analytic question without telling you.

## Parity Audit Checks

| Check          | R                                    | Python                                              |
|----------------|--------------------------------------|-----------------------------------------------------|
| Row count      | <code>nrow(df)</code>                | <code>len(df)</code> or<br><code>df.shape[0]</code> |
| Column names   | <code>names(df)</code>               | <code>df.columns.tolist()</code>                    |
| Unique levels  | <code>unique(df\$IncomeGroup)</code> | <code>df["IncomeGroup"].unique()</code>             |
| Missing values | <code>sum(is.na(df\$BMI))</code>     | <code>df["BMI"].isna().sum()</code>                 |
| Summary table  | <code>nrow(summary_df)</code>        | <code>len(summary_df)</code><br>rows                |

## Notebook Habit

After each Python cell, add a short Markdown note:

```
1 Audit check: this cell matches the R output because ...
```

If you cannot fill in that sentence, do not move to the next cell yet.

# Mixing R and Python

## Why Interoperability Matters

Sometimes a project has one useful Python tool, but the rest of the analysis is already in R. The `reticulate` package lets R call Python code and bring Python objects back into the R session. In this course, `reticulate` is awareness-level: you should know what it does and when it might be useful, but you are not expected to build a full mixed-language pipeline.

## Minimal Demo

Run this in an R console or an R Quarto document if `reticulate` is available in your Codespace.

```
1 library(reticulate)
2
3 py_run_string("
4 import pandas as pd
5 numbers = pd.DataFrame({'group': ['A', 'A', 'B'], 'value': [10, 15, 20]})
6 summary = numbers.groupby('group', as_index=False)['value'].mean()
7 ")
8
9 py$summary
```

## Health Data Example

The same idea works with the NHANES case-study CSV:

```

1 library(reticulate)
2
3 py_run_string("
4 import pandas as pd
5 nhanes = pd.read_csv('examples/nhanes-equity/data/nhanes_equity_v6.csv')
6 bmi_rows = nhanes[['BMI', 'IncomeGroup']].dropna()
7 ")
8
9 bmi_rows <- py$bmi_rows
10 head(bmi_rows)

```

## When To Use It

Use `reticulate` when:

- a small Python library solves a specific problem better than the R ecosystem;
- the rest of your report, tables, or dashboard are already in R;
- you can document the Python dependency clearly.

Avoid it when:

- a clean R solution already exists;
- mixing languages makes the project harder to reproduce;
- the Python step cannot run in a fresh Codespace.

## Check Your Understanding

In one sentence, explain what object moves from Python back into R in the NHANES example above.

# Reusable Python Scripts

## Notebook or Script?

Use a notebook when you are exploring, explaining, and checking outputs. Use a script when you have a helper function that should be reused without copying and pasting.

This mirrors the R pattern from Week 3: once a chunk of code has a clear job, turn it into a named function.

## Create `helpers.py`

In your personal workspace Explorer, select `assignments/assignment04-polyglot/submission/`. Create `helpers.py` there, beside `translation.ipynb`; a root-level copy will break the import below.

```
1 from pathlib import Path
2 import pandas as pd
3
4
5 def load_and_preview(path, n_rows=5):
6     """Load a CSV, print its shape, and return the first rows."""
7     csv_path = Path(path)
8     if not csv_path.exists():
9         raise FileNotFoundError(f"Could not find file: {csv_path}")
10
11     data = pd.read_csv(csv_path)
12     print(f"Loaded {data.shape[0]} rows and {data.shape[1]} columns")
13     return data.head(n_rows)
```

## Import It In A Notebook

In `translation.ipynb`, import and use the helper:

```
1 from helpers import load_and_preview
2
3 preview = load_and_preview("../../examples/nhanes-equity/data/nhanes_equity_v6.csv")
4 preview
```

## Reproducibility Check

Before committing:

- verify that `helpers.py` is saved in the same folder as your notebook;
- rerun the notebook from a fresh kernel;
- check that the relative path starts from the notebook location;
- commit both files together with a message such as **Add Week 5 Python helper**.

## Student Output

For Assignment 4, your reusable script should contain at least one function that is actually called from the notebook. A function that is written but never used does not demonstrate reproducibility.

# Git Progress

## Commit The Audit Trail

Week 4 introduced Git as analysis provenance. Week 5 uses that skill to make translation auditable.

Commit after meaningful checkpoints:

1. R pipeline works.
2. R function reproduces the pipeline.
3. Python notebook loads the same data.
4. Python summary matches the R summary.
5. Final parity note is complete.

## Notebook Output Hygiene

Jupyter notebooks can store rendered tables and images inside the `.ipynb` file. That makes Git diffs noisy.

Before committing:

1. Open the Command Palette with `Ctrl+Shift+P`.
2. Run Notebook: Clear All Outputs.
3. Save the notebook.
4. Stage `translation.ipynb`, `helpers.py`, and your Quarto note together.

## Example Commit Messages

```
1 Add R BMI summary for Week 5 parity check
```



```
1 Translate BMI summary to pandas and verify row counts
```

```
1 Complete Assignment 4 parity audit note
```

## Student Check

Your Git history should show the work developing in steps, not one unexplained final upload.

# Proposal Readiness

## Why This Belongs In Week 5

Before the project proposal is due, you should know whether your chosen dataset can be loaded, inspected, and explained in the course workflow. Week 5 adds one extra test: can you verify the basic data shape in both R and Python?

## Proposal Readiness Checklist

Your proposal should include:

- a focused health-data question;
- the dataset name and source;
- a brief provenance and stewardship note;
- a feasibility check showing the file can load in a Codespace;
- one or two planned descriptive summaries or visualizations;
- any known privacy, licensing, or access limitations.

## Feasibility Check

For your project dataset, record:

| Check                | What to write                                      |
|----------------------|----------------------------------------------------|
| File path            | relative path from the repo root                   |
| File format          | CSV, RDS, Excel, API, or other                     |
| Row and column count | basic shape after loading                          |
| Key variables        | names and meanings of the fields you expect to use |

| Check               | What to write                                |
|---------------------|----------------------------------------------|
| Missingness concern | one variable where missing values may matter |
| R/Python check      | whether both languages can load the file     |

## Student Output

Add this feasibility check to your project proposal notes. If your dataset does not load cleanly yet, describe the blocker honestly instead of hiding it.

# Knowledge Check

Answer these in a Markdown cell or short Quarto note.

1. Why is Week 5 called “polyglot awareness” rather than “Python replacement”?
2. Name one task where R is the better default for this course.
3. Name one task where Python might be useful.
4. Why does `my_list[1]` return the second item in Python?
5. What is the difference between `=` and `==`?
6. What does `requirements.txt` document?
7. What is the One Step, One Cell rule?
8. What are three parity checks you should run after translating R code to Python?
9. Why should notebook outputs be cleared before committing?
10. What does `reticulate` allow R to do?

## Short Applied Check

Given the R expression:

```
1 mean(nhanes$BMI, na.rm = TRUE)
```

Write the Python/pandas equivalent and name the missingness check you would run before trusting it.

# Bilingual Translation

## Purpose

This exercise turns the week into a concrete artifact: one small R analysis, one Python translation, and one audit note showing that the outputs agree.

## Required Task

Translate the Week 5 R-deepening summary into Python/pandas:

1. Start from the R pipeline in [R Deepening and Parity Check](#).
2. Create the Python translation in your personal workspace at `assignments/assignment04-polyglot/submission/translation.ipynb`.
3. From that notebook, load the NHANES CSV from `../../examples/nhanes-equity/data/nhanes_equity_v6.csv`.
4. Recreate the BMI-by-income-and-gender summary in Python.
5. Compare the Python output to the R output using row counts, grouping labels, and rounded summary values.
6. Extract one Python helper into `helpers.py` and call it from the notebook.
7. Add a Markdown cell explaining what AI helped with and how you verified the result.
8. Clear notebook outputs before committing.

## Parity Table Template

| Check                          | R result | Python result | Match? | Notes |
|--------------------------------|----------|---------------|--------|-------|
| Raw row count                  |          |               |        |       |
| Filtered row count             |          |               |        |       |
| Number of grouped rows         |          |               |        |       |
| Mean BMI values after rounding |          |               |        |       |
| Group labels                   |          |               |        |       |

## AI Prompt

I have a working R `dplyr` summary. Translate it to Python/pandas one step at a time. After each step, tell me exactly what output to compare with R. Do not add new variables, filters, plots, or interpretation.

## Definition of Done

Your exercise is complete when a reader can open the notebook, rerun the cells in order, and see why the R and Python outputs do or do not match.

# R To Python Cheatsheet

Use this as a translation aid, not as a substitute for checking outputs.

| Task           | R                                                       | Python                                                       |
|----------------|---------------------------------------------------------|--------------------------------------------------------------|
| Load package   | <code>library(dplyr)</code>                             | <code>import pandas<br/>as pd</code>                         |
| Read CSV       | <code>read_csv("file.csv")</code>                       | <code>pd.read_csv("file.csv")</code>                         |
| First rows     | <code>head(df)</code>                                   | <code>df.head()</code>                                       |
| Row count      | <code>nrow(df)</code>                                   | <code>len(df)</code> or<br><code>df.shape[0]</code>          |
| Column names   | <code>names(df)</code>                                  | <code>df.columns.tolist()</code>                             |
| Select columns | <code>select(df, BMI,<br/>Gender)</code>                | <code>df[["BMI",<br/>"Gender"]]</code>                       |
| Filter rows    | <code>filter(df, Age<br/>&gt;= 20)</code>               | <code>df[df["Age"] &gt;=<br/>20]</code>                      |
| Missing values | <code>sum(is.na(df\$BMI))</code>                        | <code>df["BMI"].isna().sum()</code>                          |
| Group summary  | <code>group_by(...)<br/> &gt;<br/>summarise(...)</code> | <code>.groupby(...).agg(...)</code>                          |
| New column     | <code>mutate(df, whtr<br/>= Waist /<br/>Height)</code>  | <code>df["whtr"] =<br/>df["Waist"] /<br/>df["Height"]</code> |
| Sort rows      | <code>arrange(df,<br/>IncomeGroup)</code>               | <code>df.sort_values("IncomeGroup")</code>                   |
| Unique values  | <code>unique(df\$Gender)</code>                         | <code>df["Gender"].unique()</code>                           |

**Dependency note:** `read_csv()` comes from the `readr` package, not `dplyr`. Load it with `library(readr)`, or use `library(tidyverse)` to get `readr` and `dplyr` together. On the Python side, every command in this table needs `import pandas as pd` first.

## Translation Rule

The translation is not finished when the code runs. It is finished when the R and Python outputs have been compared and the comparison is documented.



## **Week 6: Data Visualization**

# Week 6: Data Visualization

## Overview

This week turns a descriptive summary into a visual claim. You will use `ggplot2` as the core tool, learn how to spot the most common ways a plot misleads a reader, practice prompting an AI co-pilot for plot code you can defend, and finish with a polished figure for Assignment 5.

The emphasis is not decoration. A useful plot makes the statistic, grouping, exclusions, and limitation visible enough that a reader can understand what the figure does and does not support.

### ! Required / Draft / Optional This Week

- **Required (graded):** the seven [Assignment 5](#) files, explained one by one in [viz07](#), plus your part of the [M1 proposal initial-visual-plan.md](#).
- **Draft (studio work that feeds A5, not graded on its own):** the corrected plot, caption, and audit note from [viz06](#).
- **Optional (no penalty for skipping):** the alt-text practice in [viz07](#).

Tags like [required] and [optional] are defined in [How To Read Submission Lists](#).

## Objectives

By the end of the week you can:

- build a rerunnable `ggplot2` visualization from the NHANES Health Equity CSV snapshot;
- identify visual choices that can exaggerate, hide, or mislabel a descriptive result;
- revise title, scale, color, grouping, and caption choices to reduce misinterpretation;
- document whether a plot is weighted or unweighted, descriptive or causal, and complete-case or missingness-aware;
- use AI assistance as a draft partner while checking the code and claims yourself.

## Connection

Week 5 checked whether simple summaries agree across R and Python. Week 6 asks whether a visual summary is faithful to the evidence. Week 7 then documents the cohort, missingness, and Table 1 behind those same descriptive claims.

## Case Study Data Analysis

The **NHANES Health Equity data spine** now supports code-reading, EDA, visualization, and reproducibility checks. Use the cached CSV/RDS for routine class work; a CDC retrieval script exists for advanced users.

- Cached RDS: [examples/nhanes-equity/data/nhanes\\_equity\\_v6.rds](#)
- CSV snapshot: [examples/nhanes-equity/data/nhanes\\_equity\\_v6.csv](#)
- Case-study README: [examples/nhanes-equity/README.md](#)

The default classroom path is to use the cached data so the analysis work is reproducible without internet access.

**What to do here:** you may run the code below — it loads the cached CSV and prints two small summaries; reading it carefully is enough unless your week’s page asks for more.

```

1 library(tidyverse) # AI-EDIT(2026-06-23): tidyverse-default - consolidated core library() call
2
3 candidate_paths <- c(
4   "examples/nhanes-equity/data/nhanes_equity_v6.csv",
5   "../examples/nhanes-equity/data/nhanes_equity_v6.csv"
6 )
7
8 nhanes_path <- candidate_paths[file.exists(candidate_paths)][1]
9 if (is.na(nhanes_path)) {
10   stop("Could not find examples/nhanes-equity/data/nhanes_equity_v6.csv")
11 }
12
13 nhanes_analysis <- read_csv(nhanes_path, show_col_types = FALSE)
14
15 nhanes_analysis |>
16   summarise(
17     rows = n(),
18     missing_bmi = sum(is.na(BMI)),
19     missing_income = sum(is.na(IncomeGroup))
20   )
21 #> # A tibble: 1 x 3
22 #>   rows missing_bmi missing_income
23 #>   <int>      <int>      <int>
24 #> 1 105626      9356      9415
25
26 nhanes_analysis |>
27   filter(!is.na(BMI), !is.na(IncomeGroup), Age >= 20, Age <= 80) |>
28   group_by(IncomeGroup) |>
29   summarise(
30     n = n(),
31     mean_bmi = round(mean(BMI), 1),
32     .groups = "drop"
33   )
34 #> # A tibble: 3 x 3
35 #>   IncomeGroup      n mean_bmi
36 #>   <chr>      <int>   <dbl>
37 #> 1 High Income (>3.5) 16330    28.6
38 #> 2 Low Income (<1.3) 15293    29.4
39 #> 3 Middle Income    19228    29.3

```

## Reading order

Work through the pages in order. Each page is short and self-contained.

1. [viz01](#) — Why Visualize, and What For
2. [viz02](#) — Grammar of a ggplot, and Your First Plot
3. [viz03](#) — Worked Example: Descriptive BMI Visualization
4. [viz04](#) — How a Plot Can Mislead
5. [viz05](#) — AI Prompts You Can Defend
6. [viz06](#) — In-Class Studio: AI Visual Audit
7. [viz07](#) — Assignment 5 Walkthrough and Reference

## Class plan

1. Read the worked example ([viz03](#)) and name the statistic, grouping variable, exclusions, and limitation.
2. Inspect the flawed plot code referenced in the [studio](#).
3. Identify at least three ways the plot could mislead a reader.
4. Revise the plot so the scale, geometry, title, labels, and caption match the evidence.
5. Pair-review the corrected caption and check that it does not imply causality or population inference.

## Student output

By the end of class each student has a corrected plot, a caption, a short audit note, and an AI-use note if AI helped draft code or wording.

## Links

- Assignment: [Assignment 5 — Visualization Audit](#)
- Milestone: [M1 — Project Proposal](#) (`initial-visual-plan.md` is the connection point)

## Definition of done

- The worked example renders from source using a relative path.
- The revised figure has clear labels, an honest scale, and a limitation-aware caption.
- The audit note identifies at least three issues across correctness, design, interpretation, or reproducibility.
- The M1 proposal uses the same standard of transparency for dataset choice, feasibility, and stewardship.

## What students leave with

One polished descriptive visualization and a reusable checklist for auditing AI-assisted plots before they appear in a report, dashboard, or slide.

# Why Visualize, and What For

## Goal

Before you write any `ggplot()` code, get clear on what a plot is *for* in a health-data project. A figure is a **claim about evidence** — a reader looks at it, decides what it says, and walks away with a belief. Your job for the next two weeks is to make sure that belief matches what the data can actually support.

## The four-step pipeline you will follow

This week’s pages all support the same short pipeline. Every visualization you make for the course should run through it:

1. **Load and inspect** the data so you know what is actually there.
2. **Plot** a clear descriptive summary — one figure, one comparison.
3. **Audit** the figure against the data and the question.
4. **Caption** the figure so a reader can interpret it without reading your code.

If any step is skipped, the figure is fragile. If all four are present, the figure can survive an AI review, a peer review, or a project grader.

## What a “good” descriptive plot does

A descriptive plot in this course does three things:

- It names the **statistic** (a count, mean, proportion, median).
- It names the **comparison** (across what groups, time, or condition).
- It names the **limitation** (what the figure does *not* support — e.g., not a causal claim, not a national estimate).

A “bad” plot quietly drops one of these and lets the reader fill in the gap. Your audit habit is what catches that.

## Where AI fits

You will use an AI co-pilot to draft `ggplot2` code. You will not let it choose your scale, your title, or your caption without review. The Week 3 rule still holds: every AI-drafted chunk gets a `# AI prompt: ... # Verified: ...` comment line, and you check the column names, the missingness handling, and the claim the figure makes.

## What you produce this week

Two artifacts:

- One revised, captioned `ggplot2` figure based on the NHANES Health Equity CSV (the Assignment 5 deliverable).
- A short M1 visual plan paragraph naming the figure direction for your term project (one part of the Milestone 1 proposal due the same week).

**i** Two file formats you will keep seeing: CSV and RDS

- **CSV (.csv)** is a plain-text table: rows and columns separated by commas. Any tool can open it — R, Python, Excel, even a text editor. The classroom dataset `nhanes_equity_v6.csv` is a CSV.
- **RDS (.rds)** is R’s own format for saving one R object exactly as it was. Only R reads it (with



`readRDS()`). The same classroom dataset is also cached as `nhanes_equity_v6.rds` for R-only workflows.

 Tip

The plot you make for A5 does not need to be the same as the figure you sketch for M1. A5 is practice on a known dataset; M1 is your team's plan for your own project. The audit standard is the same in both places.

## Where this goes next

[viz02](#) teaches the grammar of a ggplot from scratch, with one minimal guided figure on the NHANES CSV.

# Grammar of a ggplot, and Your First Plot

## Where this fits

[viz01](#) framed visualization as a claim about evidence. This page teaches the five moving parts of any `ggplot2` figure and walks you through your first plot on the NHANES Health Equity CSV. Everything builds from these five parts.

## The five moving parts

Every ggplot is built from the same five ingredients. You will see all five in the worked example on the next page, so the names matter.

| Part                         | What it answers                                 | A beginner-friendly analogy                  |
|------------------------------|-------------------------------------------------|----------------------------------------------|
| <b>Data</b>                  | What table are we plotting?                     | The contents of one box                      |
| <b>Mapping (aes)</b>         | Which variable goes where (x, y, color, group)? | Telling R which column to draw on which axis |
| <b>Geometry (geom_*)</b>     | What kind of mark (point, line, bar)?           | The shape of the ink on the page             |
| <b>Scale and coordinates</b> | How are values stretched along an axis?         | The ruler under the ink                      |
| <b>Labels and theme</b>      | What does the reader see in plain text?         | The caption around the picture               |

If a reader cannot answer “which statistic? across what groups? with what limitation?” from your figure, one of these five parts is doing too little.

## Setup

Open your Codespace and start a new code chunk in a `.qmd` file. Run:

```
1 library(tidyverse)
```

### Warning

`library(tidyverse)` is required every session. If you see “could not find function `ggplot`”, run the `library()` line first.

## Load the cached NHANES CSV

This is the same file you will use for the worked example and your A5 deliverable. The candidate-path pattern below works whether your `.qmd` is at the repository root or under `weeks/week06-visualization/`.

```
1 candidate_paths <- c(  
2   "examples/nhanes-equity/data/nhanes_equity_v6.csv",  
3   "../examples/nhanes-equity/data/nhanes_equity_v6.csv"  
4 )  
5 data_path <- candidate_paths[file.exists(candidate_paths)][1]  
6 nhanes <- read_csv(data_path, show_col_types = FALSE)  
7  
8 glimpse(nhanes)
```

`glimpse()` shows the column names and types. Read them carefully. Misspelling a column name (e.g., `bmi` when the data has `BMI`) is the #1 source of broken ggplots in this course.

## Your first plot

Here is the smallest meaningful ggplot you can build from this dataset: a scatter of one numeric variable against age, with no grouping and no transformation. Try it as is, then change `Weight` to `BMI` or `Height` and re-render.

```
1 ggplot(nhanes, aes(x = Age, y = Weight)) +  
2   geom_point(alpha = 0.2) +  
3   labs(  
4     title = "Weight by age, NHANES Health Equity classroom dataset",  
5     subtitle = "One point per record; unweighted, descriptive only",  
6     x = "Age (years)",  
7     y = "Weight (kg)"  
8   ) +  
9   theme_minimal()
```

Map the five parts onto this code so the grammar feels concrete:

- **Data:** `nhanes`
- **Mapping:** `aes(x = Age, y = Weight)`
- **Geometry:** `geom_point(alpha = 0.2)` (translucent points so overlap is visible)
- **Scale and coordinates:** the default linear axes (you have not changed them)
- **Labels and theme:** `labs(...)` + `theme_minimal()`

## Three things to check on every plot

After you run any ggplot, ask:

1. **Column names match the data.** Did `glimpse()` confirm BMI is uppercase, not `bmi`?
2. **Axes tell a clear story.** Are the units readable? Is the range honest?
3. **Title and subtitle match what is plotted.** Does the title overstate (“BMI rises with age across Canada”)? If yes, soften it (“Weight by age in the prepared classroom dataset”).

## Where this goes next

[viz03](#) is the first end-to-end worked example: load the CSV, prepare a small descriptive summary, render a labeled figure, and write a caption.

# Worked Example: Descriptive BMI Visualization

## Where this fits

You learned the grammar of a ggplot and made one minimal scatter on the [previous page](#). This page is the first end-to-end visualization of the term: load the cached NHANES Health Equity CSV, prepare a small descriptive summary, render one labeled figure, and write a caption that does not overclaim. We will return to this same figure in [viz04](#) to see how easy it is to spoil with a few “improvements.”

## Goal

This worked example uses the NHANES Health Equity CSV snapshot to build one descriptive visualization. The plot summarizes mean BMI by income group across NHANES cycle. It is **unweighted** and should be interpreted as a classroom visualization exercise, not a population estimate.

**i** Definition: weighted vs unweighted

**Weighted** means the estimate accounts for survey design. **Unweighted** means it only summarizes the observed classroom/cache data. Use unweighted claims only as descriptive classroom practice.

```
1 library(tidyverse)
2
3 candidate_paths <- c(
4   "examples/nhanes-equity/data/nhanes_equity_v6.csv",
```

```

5   "../..examples/nhanes-equity/data/nhanes_equity_v6.csv"
6   )
7
8   data_path <- candidate_paths[file.exists(candidate_paths)][1]
9   if (is.na(data_path)) {
10     stop("Could not find examples/nhanes-equity/data/nhanes_equity_v6.csv")
11   }
12
13   nhanes <- read_csv(data_path, show_col_types = FALSE)

```

## Prepare a Descriptive Summary

Keep the summary intentionally simple. We are describing the prepared class dataset, not estimating a national trend.

```

1   bmi_by_income_cycle <- nhanes |>
2     filter(
3       !is.na(BMI),
4       !is.na(IncomeGroup),
5       !is.na(Cycle),
6       Age >= 20,
7       Age <= 80
8     ) |>
9     group_by(Cycle, IncomeGroup) |>
10    summarise(
11      n = n(),
12      mean_bmi = mean(BMI),
13      .groups = "drop"
14    ) |>
15    filter(n >= 30)
16
17   bmi_by_income_cycle
18   #> # A tibble: 33 x 4
19   #>   Cycle      IncomeGroup      n mean_bmi
20   #>   <chr>      <chr>      <int>   <dbl>
21   #> 1 1999-2000 High Income (>3.5) 1123    27.9
22   #> 2 1999-2000 Low Income (<1.3) 1091    28.8
23   #> 3 1999-2000 Middle Income      1327    29.0

```

```

24 #> 4 2001-2002 High Income (>3.5) 1491 28.1
25 #> 5 2001-2002 Low Income (<1.3) 1089 28.7
26 #> 6 2001-2002 Middle Income 1561 28.3
27 #> 7 2003-2004 High Income (>3.5) 1284 28.4
28 #> 8 2003-2004 Low Income (<1.3) 1183 28.6
29 #> 9 2003-2004 Middle Income 1573 28.8
30 #> 10 2005-2006 High Income (>3.5) 1480 28.5
31 #> # i 23 more rows

```

**i** Definition: hidden small cell count

A hidden small cell count is a group with very few observations that may be hidden inside a chart or summary. Do not make strong claims from these groups. That is why the code above keeps only groups with `n >= 30`.

## Plot

```

1 bmi_plot <- ggplot(
2   bmi_by_income_cycle,
3   aes(x = Cycle, y = mean_bmi, color = IncomeGroup, group = IncomeGroup)
4 ) +
5   geom_line(linewidth = 1.2) +
6   geom_point(size = 2.5) +
7   scale_color_brewer(palette = "Dark2") +
8   labs(
9     title = "Mean BMI by income group across NHANES cycle",
10    subtitle = "Adults age 20-80 in the prepared class dataset; unweighted descriptive summary",
11    x = "NHANES cycle",
12    y = "Mean BMI",
13    color = "Income group",
14    caption = "Unweighted descriptive summary; not a population estimate."
15 ) +
16 theme_minimal(base_size = 12) +
17 theme(
18   axis.text.x = element_text(angle = 45, hjust = 1),
19   legend.position = "bottom",

```



```

20   plot.caption = element_text(margin = margin(t = 8))
21   )
22
23   bmi_plot

```

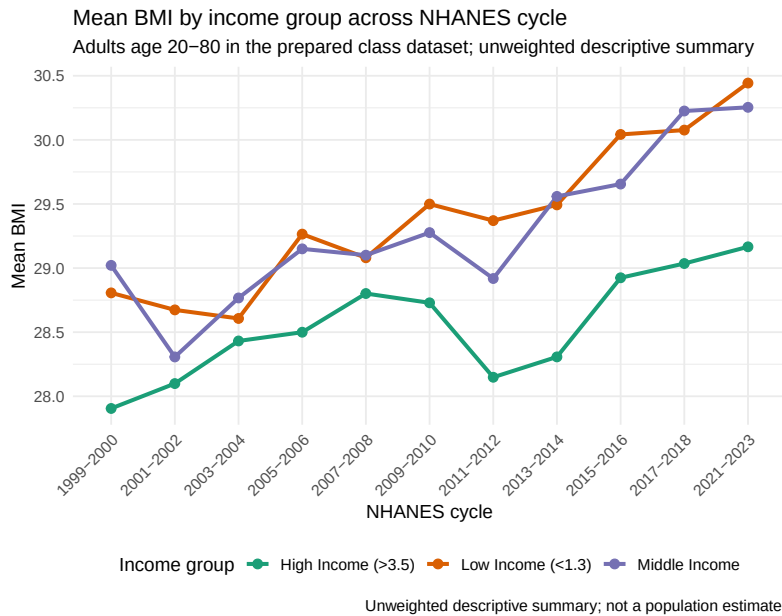


Figure 1: Mean BMI is summarized descriptively from the cached class dataset. Values are unweighted, exclude missing BMI/income/cycle values, and should not be interpreted as causal effects or population estimates.

## Optional Plotly Check

 Demo only — not graded

This pathway is a **demonstration only**. It is not graded and not required. Using it for graded work requires **instructor approval first**. The supported, graded pathway in this course is the Quarto workflow.

`plotly` is an R package that turns a static `ggplot` into an interactive web graphic — you can hover over points to see values and zoom into regions.

The core assignment uses the static `ggplot2` figure above. If `plotly` is installed, you can preview the same plot interactively while checking labels and tooltips. This extension is optional because static Quarto output is the supported submission path.

```
1 # plotly::ggplotly(bmi_plot)
```

## Caption Template

Use this structure when revising your own plot:

This figure shows [measure] by [group] across [time/category] using the prepared NHANES Health Equity classroom dataset. The summary is descriptive and unweighted. Records with missing [variables] were excluded. The plot supports comparison of patterns, not causal claims or population-level estimates.

## Mini Visual Audit Checklist

- Does the title describe what is plotted without overstating the claim?
- Is it clear whether the statistic is weighted or unweighted?
- Are axes, groups, and units labeled plainly?
- Are missing values or exclusions disclosed?
- Is the scale chosen to support honest comparison rather than exaggeration?
- Does the caption state the limitation that this is descriptive, not causal?

## Where this goes next

Hold onto this figure. The [next page](#) walks through six common ways a plot like this can mislead a reader, and [viz06](#) is the in-class studio where you apply the audit to a deliberately flawed version of it.

# How a Plot Can Mislead

## Where this fits

You have a clean, captioned worked example ([viz03](#)). This page is the audit lens: six common ways a figure that *looks fine* can quietly mislead a reader. Each one shows up in real research, and each one is something an AI co-pilot will produce if you do not push back.

## Six failure modes

For each failure mode, you will see the problem in one sentence and the fix in one sentence. Studio time in [viz06](#) revisits these on a deliberately flawed plot.

### 1. Truncated y-axis exaggeration

A small difference in means looks huge when the y-axis starts at 27 instead of 0.

- **Symptom:** `coord_cartesian(ylim = c(27.5, 31.5))` on a BMI mean plot, no narrative explanation.
- **Fix:** Start the axis at 0, or at a value justified in the caption. Use `expand_limits(y = 0)` when in doubt.

## 2. Causal title on a descriptive plot

A title that says “X causes Y” is a causal claim. Descriptive summaries cannot support it.

- **Symptom:** “Income level causes BMI to rise across NHANES cycles.”
- **Fix:** “Mean BMI by income group across NHANES cycles in the prepared classroom dataset.”

**i** Definition: causal overreach

A descriptive chart can show a pattern or association, but it does not prove that one factor caused another. Claiming cause from a descriptive plot is called causal overreach.

## 3. Smoothing across categorical time

`geom_smooth(method = "lm")` draws a line that *implies* a continuous linear trend. NHANES cycles are categorical labels, not a continuous axis.

- **Symptom:** A linear regression line layered across cycle bins.
- **Fix:** Use `geom_line()` plus `geom_point()` to connect descriptive cycle summaries, with no regression overlay.

## 4. Weighted-vs-unweighted ambiguity

If the subtitle says “weighted” but the code does not use a survey weight variable, the reader has been misled.

- **Symptom:** Subtitle: “Weighted BMI trend.” Code: `mean(BMI, na.rm = TRUE)` with no weight.
- **Fix:** State plainly that the summary is unweighted, or compute a weighted estimate using the documented weight variable.

## 5. Silent missingness

Rows with missing BMI or income disappear without a caption.

- **Symptom:** `filter(!is.na(BMI), !is.na(IncomeGroup))` deep inside a pipe with no note in the caption.
- **Fix:** Disclose missing-value exclusions in the caption. Better: report missingness counts before the figure (you will do this on [eda03](#) next week).

## 6. Hidden small-cell counts

A group with  $n = 7$  plots the same way as a group with  $n = 700$ , but the reader cannot tell them apart.

- **Symptom:** A small income group plotted as a line with no indication of sparse cells.
- **Fix:** Filter out groups below a minimum count, *or* label them visibly with  $n$  annotations, *or* report the counts beside the figure.

**i** Definition: hidden small cell count

A hidden small cell count is a group with very few observations that may be hidden inside a chart or summary. Do not make strong claims from these groups.

## Seeing it: a before/after pair

Reading about failure modes is one thing; seeing them side by side is another. The two figures below use the same summary as the worked example on [viz03](#). The first plot combines failure modes #1 and #2. The second plot is the honest version. (This pair is *not* the studio plot — the flawed plot you will audit in [viz06](#) has its own problems for you to find.)

```

1 library(tidyverse) # AI-EDIT(2026-06-23): tidyverse-default - consolidated core library() call
2
3 candidate_paths <- c(
4   "examples/nhanes-equity/data/nhanes_equity_v6.csv",
5   "../examples/nhanes-equity/data/nhanes_equity_v6.csv"
6 )
7 data_path <- candidate_paths[file.exists(candidate_paths)][1]
8 nhanes <- read_csv(data_path, show_col_types = FALSE)
9
10 bmi_by_income_cycle <- nhanes |>
11   filter(!is.na(BMI), !is.na(IncomeGroup), !is.na(Cycle), Age >= 20, Age <= 80) |>
12   group_by(Cycle, IncomeGroup) |>
13   summarise(n = n(), mean_bmi = mean(BMI), .groups = "drop") |>
14   filter(n >= 30)

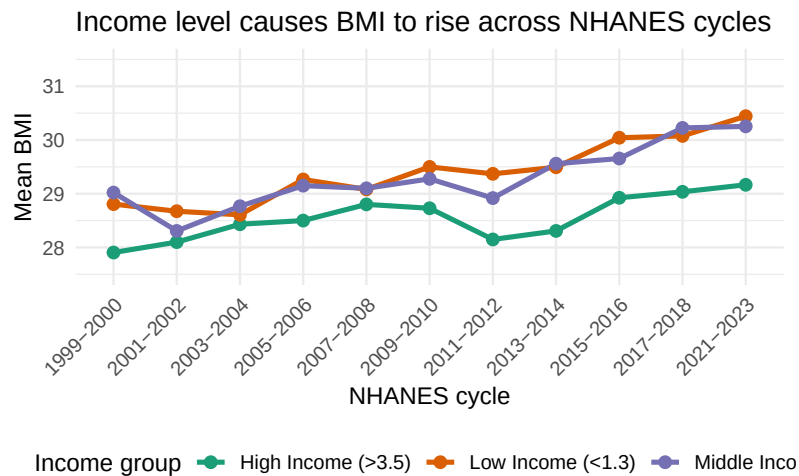
```

**Before — truncated axis and causal title.** The y-axis starts near the data, so small gaps look dramatic, and the title makes a claim the descriptive summary cannot support:

```

1 ggplot(
2   bmi_by_income_cycle,
3   aes(x = Cycle, y = mean_bmi, color = IncomeGroup, group = IncomeGroup)
4 ) +
5   geom_line(linewidth = 1.2) +
6   geom_point(size = 2.5) +
7   scale_color_brewer(palette = "Dark2") +
8   coord_cartesian(ylim = c(27.5, 31.5)) +
9   labs(
10    title = "Income level causes BMI to rise across NHANES cycles",
11    x = "NHANES cycle",
12    y = "Mean BMI",
13    color = "Income group"
14 ) +
15   theme_minimal(base_size = 12) +
16   theme(axis.text.x = element_text(angle = 45, hjust = 1), legend.position = "bottom")

```



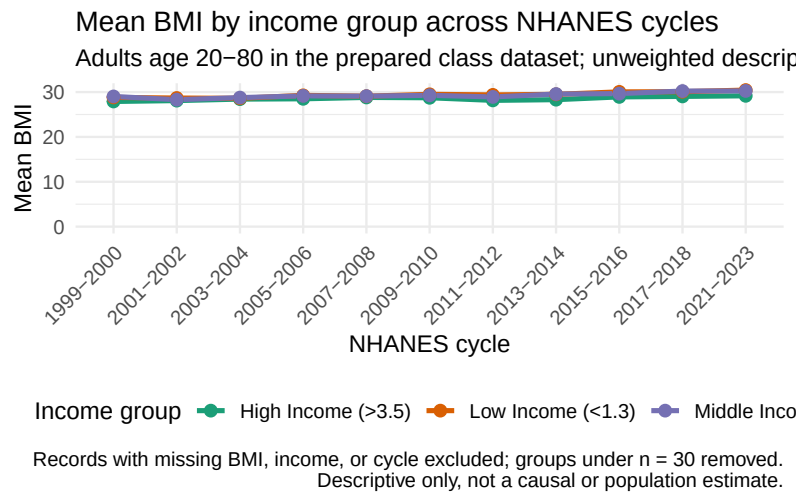
**After — honest scale and descriptive title.** Same data, same geometry. The axis starts at 0, the title describes instead of claiming cause, and the caption discloses the exclusions:

```

1 ggplot(
2   bmi_by_income_cycle,
3   aes(x = Cycle, y = mean_bmi, color = IncomeGroup, group = IncomeGroup)
4 ) +
5   geom_line(linewidth = 1.2) +
6   geom_point(size = 2.5) +
7   scale_color_brewer(palette = "Dark2") +
8   expand_limits(y = 0) +
9   labs(
10    title = "Mean BMI by income group across NHANES cycles",
11    subtitle = "Adults age 20-80 in the prepared class dataset; unweighted descriptive summary",
12    x = "NHANES cycle",
13    y = "Mean BMI",
14    color = "Income group",
15    caption = "Records with missing BMI, income, or cycle excluded; groups under n = 30 removed",
16  ) +
17  theme_minimal(base_size = 12) +
18  theme(axis.text.x = element_text(angle = 45, hjust = 1), legend.position = "bottom")

```





Notice what the fix did *not* change: the data, the geometry, and the colors are identical. Honest visualization is mostly about scale, title, and disclosure.

## Why six and not sixty

You will see other failure modes in the wild — duplicated colors, default rainbow palettes, dual axes, missing legends. The six above account for most of the issues you will catch when auditing an AI-drafted plot for descriptive health data work in this course. Master these first.

## A quick audit habit

When you finish a plot, ask three questions out loud:

1. Could a reader make a causal claim from this title or caption?
2. Could a small difference look bigger than it is because of the axis?
3. Is missingness, exclusion, or small-cell count visible to the reader?

If any answer is “yes” or “I don’t know,” revise the figure or the caption before moving on.

## **Where this goes next**

[viz05](#) shows how to write AI prompts that produce plots you can defend, and how to audit what the AI gives you.

# AI Prompts You Can Defend

## Where this fits

You can build a ggplot from scratch ([viz02](#)) and you can spot six common failure modes ([viz04](#)). This page connects the two: how to ask an AI co-pilot for plot code without inheriting its bad defaults, and how to audit what it gives you.

## Why this matters

If you paste “make me a nice plot of NHANES” into a chat window, the model will produce something — possibly with a flashy palette, a trend line, and a confident title. None of those choices were yours. None of them were audited. By the time the figure is in your report, you own them anyway.

The fix is to write prompts that **constrain the model** instead of decorate the request.

## Three prompt templates that work

### Template 1 — Start from the data, not the picture

```
I have a data frame called bmi_by_income_cycle
with columns Cycle, IncomeGroup, n, and
mean_bmi. Write a ggplot2 figure that shows
mean_bmi on the y-axis and Cycle on the x-axis,
with one line per IncomeGroup. The summary is
unweighted and descriptive. Do not add a trend
line or smoothing.
```

What this prompt does well: it names the data, the variables, the geometry, and what *not* to add.

## Template 2 — Ask for an honest scale and labels

Add `labs(...)` to the figure above so the title describes the comparison without making a causal claim, the subtitle states that the summary is unweighted and uses adults age 20-80, and the caption discloses that records with missing BMI, income, or cycle were excluded.

What this prompt does well: it tells the model what each label should *do*, not just that there should be labels.

## Template 3 — Ask for a caption that names the limitation

Write a one-sentence caption for this figure that names the statistic (mean BMI), the comparison (across NHANES cycles, by income group), the data source (prepared NHANES Health Equity classroom CSV), the exclusions (missing values, ages outside 20-80), and one limitation (descriptive only, not a population estimate).

What this prompt does well: the model now has a checklist of what the caption must contain.

## Auditing what the model gives you

After you paste in any AI-drafted plot code, walk through this checklist before you accept it:

1. **Column names match the data.** Run `glimpse(your_data)` and compare.
2. **Geometry is appropriate.** No smoothing across categorical cycles. No regression lines on a descriptive summary.

3. **Scale is honest.** Does the y-axis start at 0, or is there a stated reason it does not?
4. **Title and subtitle are descriptive.** No “X causes Y” wording.
5. **Caption discloses exclusions and limitation.** Missing values, cohort restrictions, weighted vs unweighted.
6. **na.rm = TRUE is present** wherever the summary uses a numeric column that may have NA.

If any item fails, edit the code or the labels before rendering.

## The paper-trail rule (still in force)

From Week 3: every AI-drafted chunk in your .qmd gets a two-line comment.

```
1 # AI prompt: "Plot mean_bmi by Cycle, one line per IncomeGroup, no smoothing"
2 # Verified: columns match glimpse(); no smoother; descriptive title; caption lists exclusions
3 bmi_plot <- ggplot(bmi_by_income_cycle, aes(...)) + ...
```

This is the same audit log you used in Week 3 for data wrangling. The course AI policy requires it for graded work.

## A note on what NOT to ask the AI

- Do not paste row-level NHANES data into an external chat window. Aggregate first.
- Do not ask the AI to “decide” whether income causes BMI. The figure is descriptive; causal language is your responsibility to remove.
- Do not accept a “looks fine to me” from the model as audit evidence. The audit is yours.

## Where this goes next

[viz06](#) is the in-class studio where you apply all of this to a deliberately flawed plot.

# In-Class Studio: AI Visual Audit

## Where this fits

You have the worked example ([viz03](#)), the audit categories ([viz04](#)), and the AI-prompt pattern ([viz05](#)). This page is the in-class studio where you put all three to work: read a deliberately flawed plot, identify at least three problems, revise the plot, and write a short audit note. The same workflow is what Assignment 5 expects, so think of this as the dress rehearsal.

## Before you start

Take 2 minutes to check that your Codespace is ready:

1. `library(tidyverse)` runs without error.
2. The cached data file resolves from the repository root: `examples/nhanes-equity/data/nhanes_equity_v6.csv`.
3. The rendered [Week 6 worked example](#) opens in the course book; you do not need its source file in your personal workspace.

If any of those three fail, raise it with your debugging partner before moving on.

## Goal

Evaluate one AI-generated visualization for correctness, clarity, bias, aggregation choices, and misleading design. Then revise it so the visual matches the evidence.

## Inputs

- Worked example to compare against: [weeks/week06-visualization/viz03-worked-example.qmd](#)
- Dataset: [examples/nhanes-equity/data/nhanes\\_equity\\_v6.csv](#) ([view on GitHub](#))
- Flawed plot code in your personal workspace (the artifact you will audit): `assignments/assignment05-visualization/flawed-plot-code.R`

## Activity Steps

1. Review the worked example and note how it labels the data source, statistic, and limitation.
2. In your personal workspace, open `assignments/assignment05-visualization/flawed-plot-code.R` and read the comments and output labels before you run it.
3. Run the script from your personal workspace root with `source("assignments/assignment05-visualization/flawed-plot-code.R")`.
4. Look at the figure. It may appear in a plot viewer pane. If it does not, no problem: the script also saves `flawed-plot.png` in your current working folder (run `getwd()` if you are unsure where that is). Find the PNG in the Explorer, right-click it, and choose **Open Preview**.
5. List at least three issues that could mislead a reader. Tag each one with one of the audit categories below.
6. Revise the plot so the title, scale, grouping, and caption match the evidence. Keep the cohort restriction explicit.
7. Write a short note explaining what you changed and why.

## Audit Categories

- **Correctness:** Does the plot show what the title and labels claim?
- **Design:** Do scale, color, ordering, and geometry support fair comparison?
- **Interpretation:** Does the caption avoid causal or population-level claims that the plot cannot support?

- **Bias and aggregation:** Could grouping, missingness, or small counts hide important variation?
- **Reproducibility:** Can another person rerun the code from the repository root using relative paths?

## Required Outputs (studio draft)

These are the draft files you produce in class. The Assignment 5 deliverable list ([viz07](#)) is the same set, polished.

- `visual-audit-note.md` — at least three issues with category, why it matters, and the corrected approach.
- `visualization-audit.qmd` — a rerunnable Quarto file that creates the corrected plot and explains the revision.
- `corrected-plot.png` — the exported corrected plot.
- `caption.md` — a caption that states the statistic, data source, exclusions, and limitations.
- `ai-use-note.md` — if you used AI, explain what it helped with and how you checked the result.

### What a strong audit-note entry looks like

Each entry in `visual-audit-note.md` should give this level of evidence. This example is generic — it is **not** one of the planted issues in this studio's plot:

- **Issue:** the draft layered `geom_smooth(method = "lm")` across NHANES cycles.
- **Category:** design.
- **Why it matters:** cycles are categorical labels, so a regression line implies a continuous trend the data cannot support.
- **Correction:** replaced the smoother with `geom_line() + geom_point()` connecting the cycle summaries.



## Completion Check

You are done with the studio when a peer can rerun your corrected plot, identify the main comparison without reading your code, and see at least one explicit limitation in the caption.

## Where this goes next

Polish the studio drafts into the Assignment 5 submission. [viz07](#) walks through the seven A5 files one by one and connects the work to the M1 visual plan due the same week.

# Assignment 5 Walkthrough and Reference

## Where this fits

You have the worked example, the failure modes, the prompt templates, and the studio drafts. This page is the bridge from in-class work to the Assignment 5 submission. The deliverables listed here are the same set in the [A5 README](#) — this page explains what goes into each one and how to verify it.

## Deadline and link to M1

Assignment 5 is due the Monday after Week 6 class, 4 PM. The Milestone 1 (Project Proposal) `initial-visual-plan.md` is due the same Monday. The two pieces of work share a standard of transparency, but they are *different* artifacts: A5 polishes a plot from the NHANES classroom CSV; M1 sketches one planned figure for your team’s project. Do not conflate them.

## How the audit habit carries into your M1 proposal

The five audit categories you practiced in [viz06](#) map directly onto the evidence and claims in your M1 proposal:

- **Correctness:** the claims in your proposal must match what your chosen dataset can actually show.
- **Design:** the planned figure in `initial-visual-plan.md` should name the statistic and the comparison, not just promise “a chart.”

- **Interpretation:** write proposal claims as descriptive (“we will describe...”) rather than causal (“we will show X causes Y”).
- **Bias and aggregation:** say which groups in your dataset may be small or missing, and how your team will disclose that.
- **Reproducibility:** commit to relative paths and rerunnable code from the first proposal draft.

Your `m1-link-note.md` (deliverable 6 below) is where you record this connection in your own words.

## The seven A5 deliverables, explained

Each file goes in `assignments/assignment05-visualization/submission/`.

### 1. `visualization-audit.qmd`

A rerunnable Quarto file that loads the cached CSV with a relative path, prepares a descriptive summary, renders the final corrected plot, and includes one or two short paragraphs explaining what you revised and why.

- **Verify:** open it in a fresh Codespace, render top to bottom, no errors.

### 2. `visualization-audit.html`

The rendered output of the `.qmd` above. Quarto writes this when you click Render.

- **Verify:** open the HTML in the Codespaces preview and confirm the figure, caption, and audit paragraph all appear.

### 3. corrected-plot.png

The exported corrected plot as a static image, written from code rather than by screenshot.

To save your corrected ggplot, use:

```
1 ggsave(  
2   "assignments/assignment05-visualization/submission/corrected-plot.png",  
3   plot = last_plot(),  
4   width = 7,  
5   height = 5  
6 )
```

Confirm the file appears in the submission folder before committing.

Step by step:

1. Run the code that draws your corrected plot, so it is the most recent figure in your session.
  2. Run the `ggsave()` call above. `plot = last_plot()` means “the most recent ggplot I drew.” If you stored the plot in a variable, pass that instead: `plot = my_corrected_plot`.
  3. The path starts from the repository root, so run it from there. If your R session is already inside the submission folder, shorten the path to `"corrected-plot.png"`.
  4. Check the result in VS Code: find the PNG in the Explorer panel, right-click it, and choose **Open Preview** to confirm it opens and looks right.
- **Verify:** the PNG exists, opens, and matches what the rendered HTML shows.

### 4. visual-audit-note.md

A short note listing at least three issues you identified in the flawed plot from [viz06](#). For each issue, write one sentence on what it was, one sentence on why it matters, and one sentence on how you corrected it.

- **Verify:** three issues, each tagged with a category from [viz04](#) (correctness, design, interpretation, bias/aggregation, or reproducibility).

## 5. caption.md

The caption for your final figure. Use the template from [viz03](#):

This figure shows [measure] by [group] across [time/category] using the prepared NHANES Health Equity classroom dataset. The summary is descriptive and unweighted. Records with missing [variables] were excluded. The plot supports comparison of patterns, not causal claims or population-level estimates.

- **Verify:** the caption names the statistic, the data source, the exclusions, and one limitation.

## 6. m1-link-note.md [brief — 1-3 focused bullets or sentences are enough]

Two or three sentences connecting the visualization standard from this assignment to your M1 visual plan. What did this assignment change about the figure your team plans to make? Did it change your dataset choice, the exclusions you will document, or the caption your team will write?

- **Verify:** the note is specific to *your team's* project, not generic advice.

## 7. ai-use-note.md

If you used AI to draft code or wording, name what it helped with and how you verified the result. If you did not use AI, write one sentence saying so.

Copy your inline `# AI prompt: / # Verified:` comment pairs into the note’s audit trail. The template and a filled example are on the [assignments index](#).

- **Verify:** specific (which chunk? which prompt template?), not vague (“I used Copilot”).

## Reproducibility checklist for A5

Before you click submit:

- The `.qmd` renders in a fresh Codespace with no manual edits.
- The data path is relative (`examples/nhanes-equity/data/nhanes_equity_v6.csv`).
- The PNG is generated by code (or a documented render step), not by screenshot.
- The caption uses the descriptive template and does not imply causality or national prevalence.
- Use the course devcontainer packages. If an additional package was approved, record it in your dependency note and the workspace setup so a fresh Codespace can install it.
- All seven files are committed and pushed to your GitHub repo.

## Accessibility (good practice, not a graded deliverable)

A plain-language alt text sentence next to your figure helps screen-reader users, colour-blind reviewers, and busy policy-makers. The A5 rubric does not score alt text — it is encouraged practice. If you want to try it, add `fig-alt:` as a chunk option:

## ggplot2 quick reference

| Need                       | Code                                                         | Note                                                                                           |
|----------------------------|--------------------------------------------------------------|------------------------------------------------------------------------------------------------|
| Load packages              | <code>library(tidyverse)</code>                              | Every session                                                                                  |
| Set data + mapping         | <code>ggplot(df, aes(x, y))</code>                           | The starting point                                                                             |
| Points                     | <code>geom_point()</code>                                    | Use <code>alpha</code> for overlap                                                             |
| Lines connecting summaries | <code>geom_line() + geom_point()</code>                      | Better than <code>geom_smooth()</code> for descriptive cycle data                              |
| Bars / columns             | <code>geom_col()</code>                                      | For pre-aggregated counts                                                                      |
| Labels                     | <code>labs(title, subtitle, x, y, caption)</code>            | Caption goes under the figure                                                                  |
| Honest y-axis              | <code>expand_limits(y = 0)</code>                            | Start at zero unless you justify otherwise                                                     |
| Theme                      | <code>theme_minimal()</code>                                 | Clean default                                                                                  |
| Save figure                | <code>ggsave("plot.png", plot, width = 6, height = 4)</code> | Code-generated, not screenshot; for A5 use the full submission-folder path (see deliverable 3) |

## Visual audit checklist

A short version of the five questions from [viz04](#):

- Title and caption do not imply causation.
- Y-axis starts at 0 or has a stated reason.
- Geometry suits descriptive data (no smoother across categorical x).
- Weighted vs unweighted is stated.
- Missingness and exclusions are disclosed.
- Small-cell counts are visible or filtered out.

## Five quick knowledge checks

1. What are the five moving parts of a ggplot?
2. Name two failure modes from [viz04](#) and one fix for each.
3. Why is `geom_smooth(method = "lm")` rarely appropriate for NHANES cycle data?
4. What two lines of comment should appear above any AI-drafted chunk?
5. What is the difference in scope between Assignment 5 and the M1 `initial-visual-plan.md`?

## Where this goes next

Next week ([Week 7](#)) builds the EDA evidence behind the descriptive claims you have been plotting: cohort definition, missingness reporting, Table 1, and the AI audit categories.



## **Week 7: EDA, Table 1 & AI Auditing**

# Week 7: EDA, Table 1 & AI Auditing

## Overview

This week builds the audit trail behind a descriptive analysis. You will define an analysis cohort, check missingness, create a Table 1-style summary, and audit a planted-error starter that mimics common AI-assisted mistakes.

A Table 1 is a descriptive summary of the cohort before the main analysis. It usually shows who is included, key variables, missingness, and basic group summaries. It is not a causal result. Worked examples of every piece — cohort definition, exclusions, the missingness table, complete-case analysis, and a sample Table 1 — are in [eda02](#) and [eda03](#).

EDA is treated as **evidence stewardship**: every exclusion, statistic, and interpretation should be visible enough for another person to rerun and challenge.

This week also revisits data provenance and Indigenous Data Sovereignty. The stewardship audit category ([eda05](#)) asks the same questions you met with OCAP and CARE in [Week 1](#): who has rights over this data, and does our use respect them?

### ! Required / Draft / Optional This Week

- **Required (graded):** the six [Assignment 6](#) files, explained one by one in [eda07](#), and the group [M2 preliminary analysis](#) due the same Monday.
- **Draft (studio work that feeds A6, not graded on its own):** the EDA note, missingness table, Table 1, and planted-error audit from [eda06](#).

- **Optional:** nothing extra this week — every listed file is required.

Tags like [required] and [optional] are defined in [How To Read Submission Lists](#).

## Objectives

By the end of the week you can:

- define and document an analysis cohort before summarizing;
- report missingness for key variables rather than dropping rows silently;
- create a Table 1-style descriptive summary with N, mean, and SD;
- distinguish what descriptive statistics claim from what they cannot claim;
- audit code and prose for correctness, reproducibility, interpretation, and stewardship errors.

## Connection

Week 6 focused on whether visual design makes a descriptive claim honest and clear. Week 7 documents the cohort, missingness, and summary statistics behind those claims. Week 8 then turns one audited descriptive finding into a dashboard-style KT prototype.

## Case Study Data Analysis

The **NHANES Health Equity data spine** now supports code-reading, EDA, visualization, and reproducibility checks. Use the cached CSV/RDS for routine class work; a CDC retrieval script exists for advanced users.

- Cached RDS: [examples/nhanes-equity/data/nhanes\\_equity\\_v6.rds](#)

- CSV snapshot: [examples/nhanes-equity/data/nhanes\\_equity\\_v6.csv](#)
- Case-study README: [examples/nhanes-equity/README.md](#)

The default classroom path is to use the cached data so the analysis work is reproducible without internet access.

**What to do here:** you may run the code below — it loads the cached CSV and prints two small summaries; reading it carefully is enough unless your week's page asks for more.

```

1 library(tidyverse) # AI-EDIT(2026-06-23): tidyverse-default - consolidated core library() call
2
3 candidate_paths <- c(
4   "examples/nhanes-equity/data/nhanes_equity_v6.csv",
5   "../..examples/nhanes-equity/data/nhanes_equity_v6.csv"
6 )
7
8 nhanes_path <- candidate_paths[file.exists(candidate_paths)][1]
9 if (is.na(nhanes_path)) {
10   stop("Could not find examples/nhanes-equity/data/nhanes_equity_v6.csv")
11 }
12
13 nhanes_analysis <- read_csv(nhanes_path, show_col_types = FALSE)
14
15 nhanes_analysis |>
16   summarise(
17     rows = n(),
18     missing_bmi = sum(is.na(BMI)),
19     missing_income = sum(is.na(IncomeGroup))
20   )
21 #> # A tibble: 1 x 3
22 #>   rows missing_bmi missing_income
23 #>   <int>      <int>      <int>
24 #> 1 105626      9356      9415
25
26 nhanes_analysis |>
27   filter(!is.na(BMI), !is.na(IncomeGroup), Age >= 20, Age <= 80) |>
28   group_by(IncomeGroup) |>
29   summarise(
30     n = n(),
31     mean_bmi = round(mean(BMI), 1),

```

```

32   .groups = "drop"
33   )
34 #> # A tibble: 3 x 3
35 #>   IncomeGroup      n mean_bmi
36 #>   <chr>          <int>    <dbl>
37 #> 1 High Income (>3.5) 16330     28.6
38 #> 2 Low Income (<1.3)  15293     29.4
39 #> 3 Middle Income     19228     29.3

```

## Reading order

1. [eda01 — EDA as Evidence Stewardship](#)
2. [eda02 — Cohort and Missingness](#)
3. [eda03 — Worked Example: EDA and Table 1](#)
4. [eda04 — What Each Statistic Claims](#)
5. [eda05 — Four Audit Categories for AI-Drafted EDA](#)
6. [eda06 — In-Class Studio: Planted-Error Audit](#)
7. [eda07 — Assignment 6 Walkthrough and Reference](#)

## Class plan

1. Work through the [worked example](#) using the CSV snapshot.
2. Name the cohort definition before inspecting any group summaries.
3. Build a missingness table for the variables used in the summary.
4. Produce a Table 1-style summary with N, BMI mean (SD), and age mean (SD).
5. Audit the planted-error starter in the [studio](#) and categorize at least five issues.
6. Revise the methods note so provenance, missingness, and non-causal framing are explicit.

## Student output

By the end of class each student has a rerunnable EDA note, a Table 1-style summary, a planted-error audit, a provenance/stewardship note, and an AI-use note if AI helped draft code or prose.

## Links

- Assignment: [Assignment 6 — EDA and AI Audit](#)
- Milestone: [M2 — Preliminary Analysis](#)

## Definition of done

- Cohort restrictions are stated before results are interpreted.
- Missingness is checked and reported before complete-case summaries are used.
- Table labels match the statistic actually computed.
- The audit identifies at least five planted issues across correctness, reproducibility, interpretation, and stewardship.
- M2 preliminary analysis renders from source and explains what the descriptive results do not prove.

## What students leave with

A reproducible EDA note and a habit of treating AI-generated summaries as drafts that must be checked against the data, code, and stewardship context.

# EDA as Evidence Stewardship

## Goal

Last week you turned a descriptive summary into a clear figure. This week you build the evidence trail behind that summary. The phrase “exploratory data analysis” often sounds casual — “*just look at the data*”. In a health-data project it is not casual at all: every exclusion, every statistic, and every interpretation should be visible enough that another person can rerun your work and challenge it.

Think of EDA as the **paperwork** that defends a finding. If the paperwork is missing, the finding cannot survive an audit.

## The four artifacts this week produces

Every page this week supports one of these four:

1. A **cohort definition** that names exactly who is in the analysis sample.
2. A **missingness table** that says how much of each key variable is missing.
3. A **Table 1** that summarizes the cohort with N, mean (SD), and equivalents.
4. A **methods note** that states what the table can and cannot claim.

If any one of the four is absent, the work is not ready for Milestone 2 or Assignment 6.

### **i** Key Terms This Week

- **Cohort:** the group of records included in the analysis after stated restrictions — the one-sentence answer to “who is in this analysis?”
- **Missingness:** how many values are absent for each variable you use. Report it before summarizing, never after.
- **Complete-case analysis:** keeping only rows with no missing values for the variables used. Acceptable only when the missingness counts are reported first.
- **Table 1:** the standard descriptive summary table of a cohort — N, mean (SD), and similar statistics by group — produced before any main analysis.

Each term is unpacked on [eda02](#) and in the [worked example](#).

## Why this matters for AI work

When you ask an AI co-pilot for “a Table 1,” the model will produce one. It will not check whether the cohort makes sense, whether missingness was reported first, or whether the labels match the statistics. Those are your responsibilities. The four-artifact workflow is what makes that responsibility manageable: each artifact is a small piece you can audit on its own.

The four audit categories you will use this week — **correctness, reproducibility, interpretation, stewardship** — are introduced on [eda05](#) and practiced in the [studio](#).

## What “descriptive, unweighted, non-causal” means

You will see this phrase repeatedly. Each word matters:

- **Descriptive:** the table reports what is in the data, not what the population looks like.



- **Unweighted:** we are not using NHANES survey weights, so the numbers describe the prepared classroom dataset, not US adults.
- **Non-causal:** the table cannot tell you that one variable *causes* another. Descriptive comparison is not causal inference.

Saying this in your methods note is how you defend the table from being misread.

## Where AI fits

Same rule as Week 3 and Week 6: every AI-drafted chunk in your `.qmd` gets a two-line `# AI prompt: ... # Verified: ...` comment. The verification step on Table 1 work specifically checks: cohort filter, missingness handling, label/statistic match, and interpretation.

## What you produce this week

- A rerunnable `eda-note.qmd` with cohort definition, missingness table, and Table 1 (the Assignment 6 deliverable, also the M2 preliminary analysis).
- A short audit note identifying at least five issues in a planted-error starter.
- A revised provenance/stewardship statement suitable for reuse in M2.

## Where this goes next

[eda02](#) walks through cohort definition and missingness reporting — the first two of the four artifacts.

# Cohort and Missingness

## Where this fits

[eda01](#) named the four artifacts your EDA work produces. This page covers the first two: how to define and document an analysis cohort, and how to report missingness *before* you summarize anything. Both habits are the foundation of every Table 1 you will make for the rest of the term.

## Why cohort definition comes first

A dataset is not an analysis sample. The NHANES Health Equity CSV contains records of all ages, with some variables intentionally coded as NA. If you summarize without restricting, you are answering a question the data cannot answer cleanly.

A cohort definition is a one-sentence answer to “*who is in this analysis?*” — written down before you compute any statistics. Common examples:

- “Adults age 20-80 in the prepared classroom dataset.”
- “Female respondents with non-missing BMI.”
- “Records from NHANES cycles 2013-2014 and 2015-2016.”

Each restriction has a reason. The reason goes in the methods note.

## Documenting the cohort in code

This pattern shows up in the [worked example](#). You will reuse it for A6 and M2.

```
1 # AI-EDIT(2026-06-23): tidyverse-default - consolidated core library() calls into library(tidyverse)
2 library(tidyverse)
3 library(knitr)
4
5 candidate_paths <- c(
6   "examples/nhanes-equity/data/nhanes_equity_v6.csv",
7   "../..examples/nhanes-equity/data/nhanes_equity_v6.csv"
8 )
9 data_path <- candidate_paths[file.exists(candidate_paths)][1]
10 nhanes <- read_csv(data_path, show_col_types = FALSE)
11
12 analysis_df <- nhanes |>
13   filter(Age >= 20, Age <= 80)
14
15 cohort_counts <- tibble(
16   step = c("Prepared CSV rows", "After age 20-80 restriction"),
17   n = c(nrow(nhanes), nrow(analysis_df))
18 )
19
20 kable(cohort_counts)
```

The `cohort_counts` table is your **flow diagram in text form**. It shows N before and after each restriction. If a peer asks “how many people are in your analysis?”, this table is the answer.

## Why missingness comes before the summary

A complete-case summary silently drops rows. If you compute `mean(BMI, na.rm = TRUE)` without first showing how many BMI values were missing, your reader has no idea how representative the mean is.

The audit habit is: **report missingness first, then summarize, then interpret**. Never the other order.

## Counting missingness

The cleanest beginner pattern:

```
1 key_vars <- c("BMI", "Age", "IncomeGroup", "Gender", "Race", "Cycle")
2
3 missingness <- tibble(variable = key_vars) |>
4   mutate(
5     missing_n = vapply(analysis_df[key_vars], function(x) sum(is.na(x)), numeric(1)),
6     missing_percent = round(100 * missing_n / nrow(analysis_df), 1)
7   )
8
9 kable(missingness)
```

This produces a small table: one row per key variable, with the missing count and the percent missing. Three rules apply when you write or accept code like this:

1. **Pick `key_vars` deliberately.** Only include variables you will actually use in the Table 1 or downstream summary.
2. **Use the denominator from the cohort, not the raw dataset.** Missingness as a percent of *your cohort* tells the right story.
3. **Report it, even if it is small.** A row that says “BMI missing: 2 (0.1%)” is still informative.

## What “complete-case” means and why it can bias the summary

After you see the missingness table, you have a choice. The simplest path is **complete-case analysis**: drop rows where any key variable is missing. This is what most beginner tutorials show.

That choice is reasonable for a classroom Table 1, but it is not automatically unbiased. If the rows you drop are systematically different from the rows you keep (for example, people with missing income may be different from people with reported income),

your summary tilts in a direction you cannot see from the table alone.

The honest move is to:

1. Report missingness *first*.
2. Filter to complete cases *with the missingness disclosed*.
3. Add a sentence in the methods note acknowledging the complete-case assumption.

You will see this exact sequence in the [worked example](#).

## A short audit habit

After you produce a cohort plus missingness table, ask three questions:

1. Could a reader tell who was excluded and why?
2. Is the percent missing reported for every variable in the downstream Table 1?
3. Did I describe the cohort *before* I described the summary?

If any answer is “no”, revise before the table.

## Where this goes next

[eda03](#) is the first end-to-end worked example: cohort, missingness, Table 1, methods note — all four artifacts in one re-runnable Quarto document.

# Worked Example: EDA and Table 1

## Where this fits

You met the four-artifact workflow on [eda01](#) and practiced cohort definition and missingness reporting on [eda02](#). This page is the first end-to-end EDA of the term: load the cached CSV, restrict to an adult cohort, report missingness, produce a Table 1, and write a short methods note that survives audit. The same workflow underwrites Assignment 6 and Milestone 2.

## Goal

This worked example creates a small Table 1-style descriptive summary from the NHANES Health Equity CSV snapshot. The goal is transparent EDA: define the cohort, check missingness, summarize key variables, and write a methods note that does not overclaim.

```
1 library(tidyverse) # AI-EDIT(2026-06-23): tidyverse-default - consolidated core library() call
2 library(knitr)
3
4 candidate_paths <- c(
5   "examples/nhanes-equity/data/nhanes_equity_v6.csv",
6   "../examples/nhanes-equity/data/nhanes_equity_v6.csv"
7 )
8
9 data_path <- candidate_paths[file.exists(candidate_paths)][1]
10 if (is.na(data_path)) {
11   stop("Could not find examples/nhanes-equity/data/nhanes_equity_v6.csv")
12 }
```

```

13
14 nhanes <- read_csv(data_path, show_col_types = FALSE)

```

## Define the Analysis Cohort

Decision: restrict to adults age 20-80. This keeps the example focused on adult BMI summaries and avoids mixing children, adolescents, and the oldest age-coded records into one classroom table.

```

1 analysis_df <- nhanes |>
2   filter(Age >= 20, Age <= 80)
3
4 cohort_counts <- tibble(
5   step = c("Prepared CSV rows", "Rows after age 20-80 restriction"),
6   n = c(nrow(nhanes), nrow(analysis_df))
7 )
8
9 kable(cohort_counts)

```

| step                             | n      |
|----------------------------------|--------|
| Prepared CSV rows                | 105626 |
| Rows after age 20-80 restriction | 57137  |

## Missingness Check

```

1 key_vars <- c("BMI", "Age", "IncomeGroup", "Gender", "Race", "Cycle")
2
3 missingness <- tibble(variable = key_vars) |>
4   mutate(
5     missing_n = vapply(analysis_df[key_vars], function(x) sum(is.na(x)), numeric(1)),
6     missing_percent = round(100 * missing_n / nrow(analysis_df), 1)
7   )
8
9 kable(missingness)

```

| variable    | missing_n | missing_percent |
|-------------|-----------|-----------------|
| BMI         | 1009      | 1.8             |
| Age         | 0         | 0.0             |
| IncomeGroup | 5395      | 9.4             |
| Gender      | 0         | 0.0             |
| Race        | 0         | 0.0             |
| Cycle       | 0         | 0.0             |

## Table 1-Style Summary

This example stratifies by `IncomeGroup` and reports N plus mean (SD) for BMI and age. It is unweighted and descriptive.

```

1 mean_sd <- function(x) {
2   sprintf("%.1f (%.1f)", mean(x, na.rm = TRUE), sd(x, na.rm = TRUE))
3 }
4
5 table1 <- analysis_df |>
6   filter(!is.na(IncomeGroup)) |>
7   group_by(IncomeGroup) |>
8   summarise(
9     N = n(),
10    `BMI, mean (SD)` = mean_sd(BMI),
11    `Age, mean (SD)` = mean_sd(Age),
12    .groups = "drop"
13  )
14
15 kable(table1)

```

| IncomeGroup        | N     | BMI, mean (SD) | Age, mean (SD) |
|--------------------|-------|----------------|----------------|
| High Income (>3.5) | 16520 | 28.6 (6.3)     | 49.7 (16.0)    |
| Low Income (<1.3)  | 15646 | 29.4 (7.5)     | 47.4 (18.1)    |
| Middle Income      | 19576 | 29.3 (7.0)     | 50.0 (18.4)    |



## Methods Note Template

We used the prepared NHANES Health Equity classroom CSV snapshot. The analysis cohort included records with age 20-80. We summarized BMI and age by income group using unweighted descriptive statistics. Missing values were checked before summarizing. These summaries describe the prepared class dataset and should not be interpreted as causal effects or population estimates.

## Stewardship Note Template

The dataset is a prepared classroom snapshot derived from public-use NHANES files. It should be used for descriptive learning activities, not for individual-level inference or small-cell claims. Reports should cite the source, describe exclusions, and avoid uploading row-level records to external AI tools.

## Survey Design Note

NHANES has a complex survey design. This classroom Table 1 is intentionally unweighted so that the workflow is easy to audit. A population-representative NHANES analysis would require the appropriate weights, strata, and primary sampling units.

## Reproducibility Notes

- The data path is relative to the repository.
- The cohort restriction is stated before the table.
- Missingness is reported before complete-case summaries.
- The table labels distinguish N, mean, and SD.
- The methods note states that the analysis is descriptive and non-causal.

## Where this goes next

[eda04](#) takes a step back to ask what each statistic in this table actually *claims*. [eda05](#) introduces the four audit categories you will use to evaluate a deliberately flawed version of this analysis in the [eda06](#) studio.

# What Each Statistic Claims (and Does Not)

## Where this fits

You produced a Table 1 on the [previous page](#). Before you write an interpretation of it, pause and ask: what does each cell in that table actually *claim*? This page is a short tour through the five most common descriptive statistics you will see — what each one says, what it leaves out, and the most common ways beginners (and AI tools) mislabel them.

## The five common statistics

| Statistic               | What it answers                            | What it does not answer                                        |
|-------------------------|--------------------------------------------|----------------------------------------------------------------|
| N (count)               | How many records are in this group?        | Whether the group is representative of any population          |
| Mean                    | What is the average value?                 | Whether the distribution is symmetric, skewed, or has outliers |
| SD (standard deviation) | How spread out are values around the mean? | Whether the spread is symmetric or driven by a tail            |
| Median                  | What value falls in the middle?            | How extreme the tails are                                      |

| Statistic                           | What it answers                       | What it does not answer          |
|-------------------------------------|---------------------------------------|----------------------------------|
| <b>IQR</b><br>(interquartile range) | What is the spread of the middle 50%? | The tails outside the middle 50% |

A Table 1 typically reports **N** plus either **mean (SD)** or **median (IQR)** depending on the variable.

## When to prefer mean(SD) vs median(IQR)

Use **mean (SD)** when the variable is roughly symmetric and well-behaved. Age in the adult cohort is a reasonable example.

Use **median (IQR)** when the variable is skewed or has outliers. Healthcare costs, hospital length of stay, and household income are usual suspects. The mean of a skewed variable is pulled by the long tail; the median is not.

If you are unsure, plot a histogram. If you see a long tail on one side, prefer median (IQR).

### Tip

For this course's NHANES classroom Table 1, BMI and age are commonly reported as mean (SD). That is a reasonable convention for adult anthropometric variables. State the choice and the reason in your methods note.

## What “N” alone does not tell you

A cell that says **N = 47** is informative only if the reader knows:

- which cohort the 47 came from (cohort definition, [eda02](#));
- how many of those 47 were excluded for missing values (missingness table, also [eda02](#));

- what the comparison group sizes are (the rest of the Table 1).

If any of those three is invisible, the N is not yet defensible.

## Three common beginner mistakes

These are the mistakes that show up most often in AI-drafted Table 1 code. The [studio](#) revisits them on a deliberately flawed starter.

### 1. Label says “mean” but code computes median()

```
1 # WRONG: label and code disagree
2 summarise(`Mean BMI` = median(BMI, na.rm = TRUE))
```

Either change the label to `Median BMI` or change the function to `mean()`. Decide which one you want — do not let the AI choose silently.

### 2. SD reported without `na.rm = TRUE`

```
1 # WRONG: returns NA whenever any BMI value is missing
2 summarise(sd_bmi = sd(BMI))
```

If a single NA is present, `sd(BMI)` returns NA. Use `na.rm = TRUE` and report the missingness count separately (see [eda02](#)).

### 3. Percent without a denominator

```
1 # WRONG: a "60% with low income" cell with no N
2 summarise(percent_low_income = mean(IncomeGroup == "Low Income (<1.3)") * 100)
```

A percent is only meaningful if the reader can see the denominator. Always include the N in the same row or in an adjacent cell.

## Small-cell suppression: when not to report a number

If a stratified cell has very few records (say, fewer than 10), publishing the exact statistic can be misleading or, in some health data settings, a privacy risk. The convention is to either:

- combine the small group with a neighbor and label it clearly, or
- replace the cell with a “<10” marker.

Your classroom Table 1 with  $n \geq 30$  per group (as in the [worked example](#)) is already cautious. The principle still applies: do not interpret cells where  $N$  is too small to support the claim.

## The audit habit for descriptive statistics

After you build a Table 1, look at each column heading and each row:

1. Does the column heading describe the statistic *actually* computed (mean, median, count, percent)?
2. Is the denominator visible for every percent?
3. Are missing values either filtered out (with the count reported elsewhere) or handled with `na.rm = TRUE`?
4. Are small cells either combined, marked, or excluded?

If any answer is “no”, revise the table or its labels before you write the methods note.

## Where this goes next

[eda05](#) introduces the four AI-audit categories that organize every check you have seen so far this week, ready for the [studio](#) on the planted-error starter.

# Four Audit Categories for AI-Drafted EDA

## Where this fits

You have a clean Table 1 ([eda03](#)) and a clear-eyed view of what each statistic claims ([eda04](#)). This page introduces the four categories you will use to organize your audit of AI-drafted EDA work — both in this week’s [studio](#) and in every future descriptive analysis you submit.

## The four categories

| Category               | The question it asks                                                          |
|------------------------|-------------------------------------------------------------------------------|
| <b>Correctness</b>     | Does the code do what the prose says it does?                                 |
| <b>Reproducibility</b> | Could another person rerun this in a fresh Codespace and get the same output? |
| <b>Interpretation</b>  | Does the narrative overclaim what the descriptive table can support?          |
| <b>Stewardship</b>     | Does the work respect data provenance, privacy, and classroom-use boundaries? |

Each issue you find during an audit should be tagged with one of these four. A complete A6 audit note covers **all four categories** at least once across its five or more issues.

## Beginner-level examples for each

### Correctness

- **Before:** Column labeled “Mean BMI” but code uses `median()`.
- **After:** Either rename the column or change the function. Pick one and document it.
- **Before:** Cohort prose says “adults” but the filter uses `Age >= 0, Age <= 80`.
- **After:** Decide what “adults” means (typically `Age >= 20`) and make the filter match the prose.

### Reproducibility

- **Before:** A path like `"C:/Users/me/Downloads/nhanes.csv"` in the code chunk.
- **After:** A relative path like `"examples/nhanes-equity/data/nhanes_equity_v6.csv"`, with the candidate-path pattern used in the worked example so the chunk runs from either the repo root or the week folder.
- **Before:** Missing `library(...)` calls. The chunk works in your Codespace because you ran `library(tidyverse)` earlier, but a fresh render fails.
- **After:** Every chunk that uses a package names its `library(...)` calls explicitly.

### Interpretation

- **Before:** “The table proves that income group causes differences in BMI.”
- **After:** “Mean BMI differed across income groups in this prepared classroom dataset. This descriptive comparison cannot establish cause.”



- **Before:** “Because missing values were removed, the table is unbiased and ready for publication.”
- **After:** “Complete-case filtering removed X rows. If the missingness is related to BMI or income, the complete-case summary may be biased.”

## Stewardship

- **Before:** Submitted notebook pastes a row-level NHANES record into an external AI chat to “ask for a Table 1.”
- **After:** Aggregate to group counts before any external AI is used. Never share row-level health data with external tools.
- **Before:** The methods note does not name the data source, the access method, or the classroom-use context.
- **After:** The methods note says: prepared NHANES Health Equity CSV snapshot, derived from public-use NHANES files, for classroom learning only.

### Stewardship includes Indigenous Data Sovereignty

Provenance questions do not stop at “where did the file come from?” If a dataset includes Indigenous identifiers, communities, lands, or governance-relevant content, the stewardship audit must also ask who has rights over that data. Revisit the OCAP and CARE principles from [Week 1](#) and flag the relevance in your provenance/stewardship note. The classroom NHANES snapshot does not include Indigenous identifiers, but the audit habit is to check rather than assume.

## What goes in an audit note

Each issue you flag should have three things:

1. **What** the problem is (one sentence).

2. **Why it matters** in terms of correctness, reproducibility, interpretation, or stewardship.
3. **How** you corrected it (one sentence, or a small code block).

This is the structure that A6 expects, and the studio is your rehearsal.

## A worked audit note line

**Issue 3.** The interpretation says “income group causes differences in BMI.” This is an **interpretation** problem because a descriptive summary cannot support a causal claim. Corrected: “Mean BMI differed across income groups in the classroom dataset; the comparison is descriptive and does not establish cause.”

## Stewardship recap: what goes in the provenance statement

The provenance/stewardship statement you produce for A6 (and reuse in M2) should name:

- the data source (prepared NHANES Health Equity classroom CSV);
- the source of the source (derived from public-use NHANES files at the CDC);
- the access method (cached in the repo, no internet access required for class work);
- the classroom-use boundary (descriptive learning only, not population inference);
- one privacy or stewardship caution (e.g., no row-level data uploaded to external AI tools).

Five sentences, written carefully once, reused everywhere.

## The audit habit

Every time you accept AI-drafted EDA code or prose, run through the four categories:

1. Does the code match the prose? (correctness)
2. Will it rerun in a fresh Codespace? (reproducibility)
3. Does any sentence overclaim? (interpretation)
4. Is the data and AI use respectful of the source and privacy? (stewardship)

## Where this goes next

[eda06](#) is the in-class studio where you apply these four categories to a deliberately flawed Table 1 starter.

# In-Class Studio: Planted-Error Audit

## Where this fits

You have produced a clean Table 1 ([eda03](#)), thought about what each statistic claims ([eda04](#)), and learned the four audit categories ([eda05](#)). This page is the in-class studio where you apply the audit to a starter file seeded with realistic AI-style mistakes. The studio drafts feed straight into Assignment 6.

## Before you start

Take 2 minutes to check that your Codespace is ready:

1. `library(tidyverse); library(knitr)` runs without error.
2. The cached data file resolves: `examples/nhanes-equity/data/nhanes_equity_v6.csv`.
3. Open the rendered [Week 7 worked example](#) as your clean reference. You do not need to copy or render its source in the personal workspace. Render only the supplied personal-workspace planted-error starter and your submitted correction.

If any step fails, ask your debugging partner before running the planted-error starter.

## Goal

Audit the planted-error starter, categorize at least five issues, and produce a corrected EDA note that another student could rerun.

## Inputs

- Worked example to compare against: `weeks/week07-eda-ai-audit/eda03-worked-example.qmd`
- Dataset: `examples/nhanes-equity/data/nhanes_equity_v6.csv`
- Planted-error starter in your personal workspace (the artifact you will audit): `assignments/assignment06-eda-ai-audit/planted-error-starter.qmd`

## Activity Steps

1. Review the worked example and identify the cohort definition, missingness check, and Table 1 labels.
2. In your personal workspace, open `assignments/assignment06-eda-ai-audit/planted-error-starter.qmd` and inspect both code and prose before running it.
3. Run or read the starter and annotate at least five issues.
4. Categorize each issue as **correctness**, **reproducibility**, **interpretation**, or **stewardship** (see [eda05](#) if you need a refresher).
5. Produce a corrected EDA note and Table 1.

## Required Outputs (studio draft)

These are the draft files you produce in class. The Assignment 6 deliverable list ([eda07](#)) is the same set, polished.

- `eda-note.qmd` — a rerunnable Quarto note with cohort definition, missingness table, and Table 1.
- `eda-note.html` — rendered output from the Quarto note.
- `table1.csv` — exported Table 1 or equivalent descriptive summary.
- `audit-note.md` — at least five detected issues with category, why it matters, and corrected approach.
- `provenance-stewardship-note.md` — a short note naming the data source, classroom-use context, and one privacy or stewardship caution.
- `ai-use-note.md` — if you used AI, explain what it helped with and how you verified the result.

## Minimum Table 1 Requirements

- Cohort: age 20-80.
- Stratifier: IncomeGroup.
- Variables: N, BMI mean (SD), and age mean (SD).
- Methods note: state that the table is descriptive, un-weighted, and non-causal.

## Completion Check

You are done with the studio when another student can rerun your `eda-note.qmd`, see exactly who is included, understand how missingness was handled, and identify why the planted-error starter was misleading.

## Where this goes next

Polish the studio drafts into the Assignment 6 submission and the Milestone 2 deliverables. [eda07](#) walks through the A6 files one by one and shows which files double as M2 evidence.

# Assignment 6 Walkthrough and Reference

## Where this fits

You have produced the four artifacts in class. This page is the bridge from studio drafts to the Assignment 6 submission and the Milestone 2 preliminary analysis. The deliverables here mirror the [A6 README](#) exactly — this page explains what goes into each file and how to verify it.

## Deadline and link to M2

Assignment 6 is due the Monday after Week 7 class, 4 PM. Milestone 2 (Preliminary Analysis) is due the same Monday. The good news: most of the A6 files double as M2 deliverables, so you are doing the work once.

| A6 deliverable                    | Doubles as M2 deliverable?                                                     |
|-----------------------------------|--------------------------------------------------------------------------------|
| eda-note.qmd                      | Yes — corresponds to preliminary-analysis.qmd                                  |
| eda-note.html                     | Yes — corresponds to preliminary-analysis.html                                 |
| table1.csv                        | Yes — same file                                                                |
| missingness table in eda-note.qmd | Export it as missingness-summary.csv for M2; it is not an extra A6 deliverable |
| audit-note.md                     | A6 only                                                                        |
| provenance-stewardship-note.md    | Yes — feeds the M2 methods-note.md                                             |

| A6 deliverable              | Doubles as M2 deliverable?                                    |
|-----------------------------|---------------------------------------------------------------|
| <code>ai-use-note.md</code> | Yes — same filename in the M2 <code>submission/</code> folder |

A6 is your individual assignment submission. M2 is your group milestone submission. If you reuse a `.qmd` file, copy it into the correct submission folder/repository and use the required filename for each submission. M2 may build on one member's A6; put one attribution sentence in the existing M2 `methods-note.md`.

## How to move your A6 file into the group repository

A6 lives in your individual repository; M2 lives in your group repository. Reusing a file means copying it between the two. The safe beginner path:

1. In your individual repository's Codespace, open your finished `eda-note.qmd`. Select all (Ctrl+A / Cmd+A) and copy.
2. Open a Codespace on the **group** repository.
3. Create the file with the M2 name and location — `milestones/m2-preliminary-analysis/submission/preliminary-analysis.qmd` — paste the contents, and save.
4. Render the file in the group Codespace. If the data path does not resolve, copy the CSV into the group repository with the same relative folder structure, or update the relative path.
5. Export the A6 missingness result by code as `milestones/m2-preliminary-analysis/submission/missingness.qmd`. This export is required for M2 only.
6. Repeat for the other reused files, using the M2 filenames from the table above.
7. Add one attribution sentence to M2 `methods-note.md`, for example: "The preliminary analysis builds on Jordan's A6 `eda-note.qmd`."
8. Commit and Sync Changes in the group repository.

Do not email files to yourself or drag them between browser windows — copying through the editor and committing in the



group repository keeps every step visible in the project history.

## The six A6 deliverables, explained

Each file goes in `assignments/assignment06-eda-ai-audit/submission/`.

### 1. `eda-note.qmd`

A rerunnable Quarto file that loads the CSV with a relative path, defines the analysis cohort, reports missingness, produces the Table 1, and includes a short methods note. Pattern after the [worked example](#).

- **Verify:** open it in a fresh Codespace, render top to bottom, no errors.

### 2. `eda-note.html`

The rendered output of the `.qmd` above.

- **Verify:** open the HTML in the Codespaces preview and confirm the cohort flow, missingness table, and Table 1 all appear.

### 3. `table1.csv`

The Table 1 exported as a CSV. Use `write_csv(table1, "table1.csv")` from inside the `.qmd` so it is generated by code, not by copy-paste.

- **Verify:** the CSV opens, has the same row counts as the rendered table, and contains N, BMI mean (SD), and age mean (SD) by `IncomeGroup`.

#### 4. audit-note.md

A short note listing at least five issues you identified in the planted-error starter ([studio page](#)). Use the structure from [eda05](#):

Issue 1. [What it is]. This is a [category] problem because [why it matters]. Corrected: [how]

- **Verify:** five issues, each tagged with one of the four categories (correctness, reproducibility, interpretation, stewardship), and each with a corrected approach.

#### 5. provenance-stewardship-note.md

A short note suitable for reuse in M2. Use the five-sentence template from [eda05](#): data source, source of source, access method, classroom-use boundary, one privacy caution.

- **Verify:** five named items, each in a complete sentence.

#### 6. ai-use-note.md

If you used AI to draft code or prose, name what it helped with and how you verified the result. If you did not use AI, write one sentence saying so.

Wondering how this note relates to the inline `# AI prompt: / # Verified: comments from eda01`? Both are expected, but only one is graded: `ai-use-note.md` is the graded artifact, and the inline comments are the required working practice that feeds it — copy your comment pairs into the note’s audit trail. The template and a filled example are on the [assignments index](#).

- **Verify:** specific to chunks or prompts, not generic.

## Reproducibility checklist for A6

Before you click submit:

- The `.qmd` renders in a fresh Codespace with no manual edits.
- The data path is relative.
- `table1.csv` is written by code (`write_csv(...)`).
- Missingness is reported *before* the Table 1, not after.
- The methods note avoids causal language and names the data as descriptive and unweighted.
- Use the course devcontainer packages. If an additional package was approved, record it in your dependency note and the workspace setup so a fresh Codespace can install it.
- All six files are committed and pushed.

## A methods-note template you can reuse

We analyzed the prepared NHANES Health Equity classroom CSV snapshot. The cohort included records with age 20-80. We summarized N, BMI mean (SD), and age mean (SD) by `IncomeGroup` using unweighted descriptive statistics. Records with missing BMI, age, or income were excluded after the missingness counts were reported. The summary describes the prepared classroom dataset only and should not be interpreted as causal effects or population estimates.

That is five sentences. Use it as a starting point, edit to match your team's choices.

## A provenance/stewardship sentence you can reuse

The dataset is a prepared classroom snapshot derived from public-use NHANES files at the CDC,

cached in the course repository for offline classroom use. It supports descriptive learning activities only — not individual-level inference, small-cell claims, or upload of row-level records to external AI tools.

## Five quick knowledge checks

1. What is a cohort, in one sentence?
2. Why report missingness *before* dropping rows?
3. When is median (IQR) more honest than mean (SD)?
4. Name the four AI-audit categories.
5. Which A6 file feeds the M2 `methods-note.md`?

## EDA quick reference

| Task          | Code                                                                          | Note                       |
|---------------|-------------------------------------------------------------------------------|----------------------------|
| Load CSV      | <code>read_csv("examples/Relative-equity/data/nhanes_equality_v6.csv")</code> | Relative equity            |
| Cohort filter | <code>filter(Age &gt;= 20, Age &lt;= 80)</code>                               | Document the reason        |
| Cohort counts | <code>tibble(step, n)</code>                                                  | Flow diagram in text       |
| Missingness   | <code>vapply(df[vars], function(x) sum(is.na(x)), numeric(1))</code>          | Use the cohort denominator |
| Mean (SD)     | <code>sprintf("%.1f (%.1f)", mean(x, na.rm=TRUE), sd(x, na.rm=TRUE))</code>   | One readable string        |
| Group summary | <code>group_by(IncomeGroup) %&gt;% summarise(...)</code>                      | One row per group          |
| Export        | <code>write_csv(table1, "table1.csv")</code>                                  | Generated by code          |

## Where this goes next

Next week ([Week 8](#)) turns the audited Table 1 from this assignment into a small dashboard-style KT product. The cleaner your A6/M2 work is, the easier the dashboard will be.

## **Week 8: Dashboard Prototypes for KT**

# Week 8: Dashboard Prototypes for KT

## Overview

This is the first app-forward week. You will move from audited EDA outputs (Week 7) to a small dashboard-style Knowledge Translation product that helps a defined audience explore one descriptive question.

The **required core** pathway is the [Quarto starter dashboard](#): it renders reliably in every Codespace, makes the analysis visible, and is the file you will adapt for Assignment 7. The Shiny exemplar discussed on [dash05](#) is a **demonstration only** — your instructor will run it in class; you are not required to install Shiny, run the app, or produce A7 deliverables from it.

### ! Required / Draft / Optional This Week

- **Required (graded):** the nine [Assignment 7](#) files, explained one by one in [dash07](#).
- **Draft (studio work that feeds A7, not graded on its own):** the render note, audience sentence, before/after text, testing log, and limitation statement from [dash06](#).
- **Demo-only (not graded, do not build on it):** the Shiny exemplar on [dash05](#) — watch it; do not submit A7 work from it.

Tags like [required] and [demo-only] are defined in [How To Read Submission Lists](#).

## Objectives

By the end of the week you can:

- explain the difference between a static report, a Quarto dashboard-style product, and a Shiny app;
- render the Quarto starter dashboard locally and adapt one audience-facing element;
- apply the 1-1-1-1 rule to translate one Week 7 finding into a dashboard prototype;
- distinguish text/UI changes (allowed for A7) from analytic changes (not allowed for A7);
- test a dashboard locally and record what was checked;
- apply privacy-safe publishing guidance before sharing any health-data product.

## Connection

Week 7 established an auditable descriptive foundation. Week 8 asks how a non-technical audience will encounter that evidence in a dashboard. Week 9 turns one dashboard finding into a concise scientific communication product.

## Dashboard Case Study: NHANES Equity

The **NHANES Health Equity Dashboard** is now the foreground case study. Use it as a small knowledge-translation product built on the cached NHANES equity dataset.

- App code: `examples/nhanes-equity/app/app.R`
- Cached data: `examples/nhanes-equity/data/nhanes_equity_v6.rds` and `examples/nhanes-equity/data/nhanes_equity_v6.csv`
- Optional rebuild script: `examples/nhanes-equity/scripts/build_nhanes_equity.R`

Run the dashboard from the repository root:

```
1 shiny::runApp("examples/nhanes-equity/app")
```



The app is offline-first: it loads the local RDS cache when available. Rebuilding from CDC uses `nhanesA`, requires internet access, and is optional unless assigned.

Companion materials:

- Week 10 report template: `weeks/week10-reporting-publishing/report-template.qmd`
- Week 11 static dashboard demo: `weeks/week11-portfolio-surgery/static-dashboard-ojs.qmd`

## Reading order

1. [dash01](#) — What a Dashboard Is, and Three Pathways
2. [dash02](#) — From One Finding to a Prototype: The 1-1-1-1-1 Rule
3. [dash03](#) — Starter Dashboard (required core artifact)
4. [dash04](#) — Audience-Facing Text: What You Can and Cannot Change for A7
5. [dash05](#) — Testing, Privacy-Safe Publishing, and the Shiny Demo
6. [dash06](#) — In-Class Studio: Dashboard Adaptation
7. [dash07](#) — Assignment 7 Walkthrough and Reference

## Class plan

1. Render the [Quarto starter dashboard](#) from the repository root.
2. Choose a stakeholder audience and write one question that audience could answer from the dashboard.
3. Identify one control, title, caption, help text, or interpretation note to revise.
4. Improve that audience-facing text without changing analytic logic.
5. Re-render the dashboard with a different `selected_income` value and record the test result.
6. Watch the optional Shiny exemplar demonstration ([dash05](#)) — for awareness only, not for A7.

## Student output

By the end of class each student has a local render note, an audience sentence, a before/after text change, a testing log entry, and a limitation statement — the studio drafts of every Assignment 7 file.

## Links

- Assignment: [Assignment 7 — Dashboard KT Prototype](#)

## Definition of done

- The Quarto starter dashboard renders locally from the cached NHANES data.
- The audience-facing change improves clarity without changing the analysis.
- The testing log records how the dashboard was checked in Codespaces.
- The privacy-safe publishing note avoids row-level data sharing, small-cell claims, and causal overreach.
- No A7 submission uses the Shiny app as its primary deliverable.

## What students leave with

A tested dashboard adaptation note and a decision rule for the term project: use the simplest pathway that serves the audience, with the Quarto pathway as the supported default and any other pathway requiring instructor approval.

# What a Dashboard Is, and Three Pathways

## Goal

Last week you produced an audited Table 1. This week turns one descriptive finding from that table into a small **audience-facing artifact** — a dashboard-style KT product. Before you build, get clear on what a dashboard is, what it is not, and which pathway you will use in this course.

## A dashboard is not a report

A short way to remember the difference:

- A **report** tells a fixed story to a known audience. It loads, renders, and stays the same.
- A **dashboard** lets a defined audience answer one question with one or two controls. The audience moves through the artifact and pulls out the slice that matters to them.
- A **data warehouse** is neither. It stores everything for someone else to query.

You are building the middle one. The audience is real (an analyst, a project team, a briefing group). The control is small (one dropdown, one filter, or one editable value). The interpretation and the limitation stay attached to whatever the audience sees.

A dashboard is **not** a free pass to be sloppy. It can mislead just as easily as a single figure can. Everything you learned in Weeks 6 and 7 about scale, captioning, missingness, and stewardship still applies.

## Three pathways, three roles in this course

| Pathway                            | What it is                                                                                                       | When it makes sense                                                   | Role in this course                                                |
|------------------------------------|------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------|--------------------------------------------------------------------|
| <b>Quarto dashboard-style page</b> | A rerunnable .qmd with one or two editable values, a visualization, a table, an interpretation, and a limitation | Default; works in every Codespace; no server                          | <b>Required core</b> for A7                                        |
| <b>Shiny app exemplar</b>          | A live R server with reactive UI controls                                                                        | When real-time interactivity matters and you can host or run a server | <b>Demonstration only</b> — your instructor will run this in class |
| <b>Static HTML/JS dashboard</b>    | AI-generated self-contained files                                                                                | When a public-facing portfolio piece must run without a server        | <b>Optional demo</b> in Week 11                                    |

The **required** A7 deliverable is built on the Quarto pathway. Shiny and static HTML/JS are shown so you know they exist; they are not required of any student.

## Decision rule for your term project

When you choose a dashboard pathway for your team's final portfolio, use the simplest pathway that serves your audience:

- If a `.qmd` page with one editable value answers the audience question, that is the right choice.
- If real-time interactivity is essential and your team can run/host a server, the Shiny pathway is open — **with instructor approval**, since it is not a default course pathway.
- If a static, public-facing artifact is required, the Week 11 static HTML/JS demo path is the option to explore.

There is no rubric bonus for using a more complex pathway. The grading rubric measures audience fit, audit quality, and reproducibility — not technical flash.

## What “dashboard-style” means for Quarto

You will see this phrase throughout Week 8. It means a Quarto page that:

- loads the cached classroom dataset with a relative path,
- declares one **control value** as an editable R variable near the top of the file,
- shows one **table** and one **visualization** that update when the control changes,
- ends with one **interpretation** sentence and one **limitation** sentence.

That is what the [starter dashboard](#) does, and that is what your A7 adaptation will continue.

## What “demonstration only” means for Shiny

You will see the Shiny exemplar run live in class. You will be invited to open the URL the instructor shares and click around. You will **not** be required to:

- install or load Shiny in your own Codespace,
- run `shiny::runApp(...)` yourself,
- modify the Shiny app code,
- produce any A7 deliverable from the Shiny app.

If you find Shiny interesting and want to explore it for your term project, talk with your instructor before committing — Shiny is not a default approved A7 pathway and not a default approved final-portfolio pathway.

## **Where this goes next**

[dash02](#) shows the “1-1-1-1-1 rule” that lets you translate one Week 7 finding into a dashboard sketch.

# From One Finding to a Prototype: The 1-1-1-1-1 Rule

## Where this fits

You know what a dashboard is and which pathway is required ([dash01](#)). This page is the translation step: how do you take one Week 7 finding and turn it into a small dashboard prototype? The answer is a rule with five ones.

## The 1-1-1-1-1 rule

For your first dashboard prototype, commit to exactly:

- **1 audience** — name a real person or role (e.g., “a public-health analyst preparing a briefing”).
- **1 question** — phrased the way the audience would ask it.
- **1 control** — one thing the audience can change (one dropdown, one filter, one editable value).
- **1 visualization** — one chart that updates when the control changes.
- **1 limitation** — one sentence about what the artifact does *not* support.

Beginners try to build a dashboard with three controls, four charts, and a tab system. That dashboard is harder to make, harder to test, and easier to break. The 1-1-1-1-1 rule keeps you honest.

## A worked example using your Week 7 Table 1

Recall the Table 1 from [eda03](#): mean BMI by income group across NHANES cycles. Here is the 1-1-1-1-1 translation:

| Field         | Filled in                                                                                   |
|---------------|---------------------------------------------------------------------------------------------|
| Audience      | A public-health analyst preparing a short briefing on descriptive BMI patterns              |
| Question      | “Among adults age 20-80, how does mean BMI vary across NHANES cycles for one income group?” |
| Control       | A single <code>selected_income</code> value the analyst can change                          |
| Visualization | A line plot of mean BMI by cycle for the selected income group                              |
| Limitation    | The summary is descriptive and unweighted; it does not support causal or population claims  |

That is exactly what the [starter dashboard](#) implements.

## How to choose each field for your own project

### Audience

Be specific. “The public” is too broad. “A first-year MPH student doing a course assignment” is specific. “A regional health officer planning a fall briefing” is specific. The more specific the audience, the easier every later choice becomes.



## Question

Phrase it the way your audience would phrase it. Not “an exploration of the multivariate structure of NHANES BMI distributions” but “How does average BMI compare across income groups in this dataset?”. Plain language sharpens the rest of the design.

## Control

One control. If you find yourself wanting two, pick the one your audience would change first. The second can wait for a later version.

Reasonable Quarto controls for this course:

- An editable R variable near the top of the `.qmd` (the starter pattern — change the value, re-render).
- A Quarto parameter declared in the YAML header (`params:` block), updated at render time.
- A `selectInput`-style control inside a Shiny app — but only if you have approval for the Shiny pathway.

## Visualization

One figure. Apply everything from Week 6: honest scale, descriptive title, weighted/unweighted disclosure, missing-value note, caption template. The dashboard does not relax those standards — if anything, it raises them, because an audience-facing artifact reaches more readers.

## Limitation

One sentence. Use the templates from [eda05](#) on what descriptive comparison cannot establish. Place the sentence where the audience will actually see it — at the bottom of the figure caption, or under the table, or beside the control.

## What to write down before you build

Before you touch any code, write three lines in a note:

1. My audience is: \_\_\_\_\_
2. My audience would ask: \_\_\_\_\_
3. The control I will offer is: \_\_\_\_\_

If you cannot finish those three lines, the dashboard is not ready to build yet. Go back to Week 7 and pick a finding you can actually frame.

## A note on scope creep

Once you build your first prototype, you will be tempted to add a second control, a second figure, an “advanced settings” panel, or interactive tooltips. Resist. Every addition multiplies the surface area for misleading the audience. The A7 rubric does not reward extra controls; it rewards a clearly framed audience-facing artifact.

## Where this goes next

[dash03](#) is the working Quarto starter dashboard that implements the 1-1-1-1-1 worked example. You will adapt this file for A7.

# Starter Dashboard

## Where this fits

You named the three pathways on [dash01](#) and applied the 1-1-1-1 rule on [dash02](#). This page is the **required core artifact** for Week 8: the Quarto-rendered dashboard-style page you will adapt for Assignment 7. The Shiny exemplar discussed on [dash05](#) is a demonstration only — your A7 deliverable is built from the file on this page, not from the Shiny app.

## Goal

This starter shows how to turn the Week 7 descriptive summary into a dashboard-style KT product. It uses one editable control value, one visualization, one table, one interpretation, and one caution — the 1-1-1-1 rule made concrete.

## Audience Question

Audience: a public-health analyst preparing a short briefing on descriptive BMI patterns in the classroom NHANES equity dataset.

Question: among adults age 20-80, how does mean BMI vary across NHANES cycles for one selected income group?

## Setup

```

1 library(tidyverse) # AI-EDIT(2026-06-23): tidyverse-default - consolidated core library() call
2 library(knitr)
3
4 candidate_paths <- c(
5   "examples/nhanes-equity/data/nhanes_equity_v6.csv",
6   "../examples/nhanes-equity/data/nhanes_equity_v6.csv"
7 )
8
9 data_path <- candidate_paths[file.exists(candidate_paths)][1]
10 if (is.na(data_path)) {
11   stop("Could not find examples/nhanes-equity/data/nhanes_equity_v6.csv")
12 }
13
14 nhanes <- read_csv(data_path, show_col_types = FALSE)

```

## Control

In a Shiny app, this would become a dropdown. In this static starter, change the value of `selected_income` and render again.

```

1 available_income_groups <- sort(unique(na.omit(nhanes$IncomeGroup)))
2 selected_income <- "Low Income (<1.3)"
3
4 if (!selected_income %in% available_income_groups) {
5   selected_income <- available_income_groups[1]
6 }
7
8 selected_income
9 #> [1] "Low Income (<1.3)"

```

For example, change the line to `selected_income <- "Middle Income"` — the other valid value in the classroom dataset is "High Income (>3.5)". Copy the value exactly, including the quotes and any parentheses.

## Summary

```
1 dashboard_summary <- nhanes |>
2   filter(
3     Age >= 20,
4     Age <= 80,
5     IncomeGroup == selected_income,
6     !is.na(BMI),
7     !is.na(Cycle)
8   ) |>
9   group_by(Cycle) |>
10  summarise(
11    n = n(),
12    mean_bmi = round(mean(BMI), 1),
13    .groups = "drop"
14  )
15
16 kable(dashboard_summary)
```

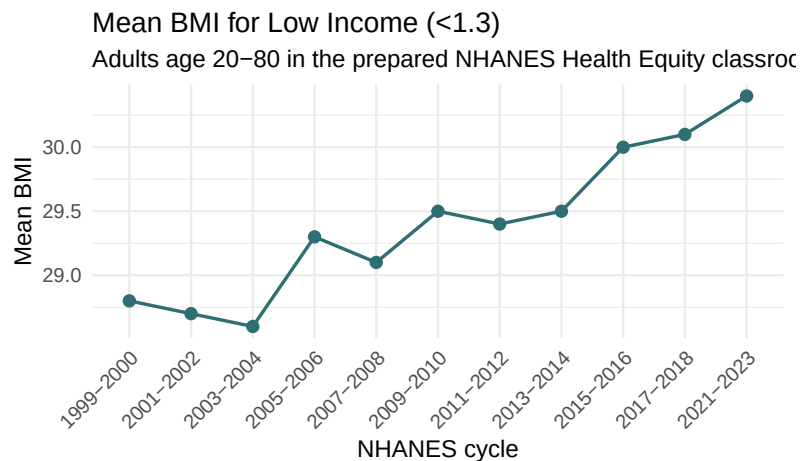
| Cycle     | n    | mean_bmi |
|-----------|------|----------|
| 1999-2000 | 1091 | 28.8     |
| 2001-2002 | 1089 | 28.7     |
| 2003-2004 | 1183 | 28.6     |
| 2005-2006 | 1091 | 29.3     |
| 2007-2008 | 1552 | 29.1     |
| 2009-2010 | 1817 | 29.5     |
| 2011-2012 | 1724 | 29.4     |
| 2013-2014 | 1762 | 29.5     |
| 2015-2016 | 1568 | 30.0     |
| 2017-2018 | 1275 | 30.1     |
| 2021-2023 | 1141 | 30.4     |

## Visualization

```

1 ggplot(dashboard_summary, aes(x = Cycle, y = mean_bmi, group = 1)) +
2   geom_line(linewidth = 0.8, color = "#2f6f73") +
3   geom_point(size = 2.4, color = "#2f6f73") +
4   labs(
5     title = paste("Mean BMI for", selected_income),
6     subtitle = "Adults age 20-80 in the prepared NHANES Health Equity classroom dataset",
7     x = "NHANES cycle",
8     y = "Mean BMI",
9     caption = "Descriptive, unweighted classroom summary. Missing BMI, income group, cycle, and out-of-cohort ages are excluded."
10  ) +
11  theme_minimal(base_size = 12) +
12  theme(axis.text.x = element_text(angle = 45, hjust = 1))

```



om summary. Missing BMI, income group, cycle, and out-of-cohort ages are excluded.

## Interpretation

For the selected income group, the table and figure describe the mean BMI pattern across available NHANES cycles in the prepared classroom dataset. The dashboard should help the audience ask better questions about pattern, timing, and context rather than treat the line as a causal explanation.

## Limitation

This starter is unweighted and descriptive. It does not account for NHANES survey design, does not estimate national prevalence, and should not be used to make causal claims about income and BMI.

## Testing

Before submitting a dashboard-style product:

1. Render this page from the repository root.
2. Change `selected_income` to another value from `available_income_groups`.
3. Render again and check that the table, title, and plot all update.
4. Verify that the caption still names the dataset, exclusions, and limitation.
5. Record any warning or error in your testing log.

### 💡 How to render: the course default

1. Open the `.qmd` file you want to render, then open the **Command Palette** (**View** → **Command Palette**, or press **Ctrl+Shift+P** on Windows/Linux / **Cmd+Shift+P** on Mac) and run **Quarto: Render Document**. The Quarto extension renders the file and opens a preview pane.
2. **Always-works fallback (terminal):** type the render command with the file's path as shown in the Explorer, for example:

```
1 quarto render practice_report.qmd
```

Always name the file — a bare `quarto render` rebuilds the *whole* project and takes much longer.

3. If you see a **Render** (or **Preview**) button in the editor toolbar, it does the same thing as step 1.

**Where did the output go?** Watch the render log for the `Output created:` line — it names the exact `.html` file created. By default the file appears next to your `.qmd`; in projects that set an output directory (like this book’s `docs/` folder), it appears there instead. If no preview opened automatically, find that `.html` file in the Explorer, right-click it, and choose **Download** to open it in your browser (or **Open Preview** if available).

HTML is the expected output for this dashboard — you do not need a PDF. If the render log warns about a missing TeX installation or a PDF problem, you can ignore it as long as the `.html` file was created. [dash05](#) walks through the full six-step test routine for this same page.

## Privacy-Safe Publishing

- Use aggregate summaries, not row-level records.
- Avoid small-cell interpretations when `n` is low.
- State that outputs are descriptive and unweighted.
- Keep cached classroom data separate from any raw NHANES retrieval workflow.
- Do not upload row-level data to external AI tools.

## Where this goes next

[dash04](#) shows what counts as an audience-facing text change (allowed) versus an analytic change (not allowed for A7). [dash05](#) walks through how to test this page in Codespaces and shows the optional Shiny exemplar that your instructor will demo in class.



# Audience-Facing Text: What You Can and Cannot Change for A7

## Where this fits

[dash03](#) is the file you will adapt for Assignment 7. This page draws the line between **audience-facing text changes** (allowed for A7) and **analytic changes** (not allowed for A7). Knowing where that line is keeps the assignment focused and keeps your audit trail clean.

## The rule, in one sentence

For Assignment 7, you may change anything the audience reads. You may not change anything the analysis computes.

## Allowed changes (text-only)

These are the kinds of changes the A7 rubric rewards. They sharpen the audience experience without touching the analysis.

- **Rename a label.** “Cycle” becomes “Survey wave (NHANES cycle)” so a non-expert audience knows what to read.
- **Expand a caption.** A two-word caption becomes a one-sentence caption that names the statistic, source, exclusions, and limitation.

- **Add a help text line.** Beside the control, a sentence that says “This dashboard summarizes adults age 20-80 from the prepared classroom dataset.”
- **Add a limitation sentence.** Beside the figure, one sentence on what the artifact does not support.
- **Improve a README sentence.** A README that says “Dashboard for NHANES” becomes “Descriptive dashboard of unweighted mean BMI by income group across NHANES cycles, for classroom use.”
- **Adjust a title for the audience.** A neutral title becomes one that names the audience-facing question.

## Not allowed for A7 (analytic-only changes)

These changes touch the analysis. They are not part of A7. If you find a real analytic issue while testing, raise it with your instructor — do not silently change it.

- **Changing the cohort filter.** Switching from `Age >= 20` to `Age >= 18` is an analytic decision.
- **Adding or removing a survey weight.** Switching from unweighted to weighted is an analytic decision.
- **Adding a regression line.** Smoothing across cycles is an analytic decision (and usually inappropriate — see [viz04](#)).
- **Replacing `mean()` with `median()`.** Changing the central tendency statistic is an analytic decision.
- **Changing the grouping variable.** Switching from `IncomeGroup` to `Education` is an analytic decision.

If you are unsure whether a change is text or analytic, ask: “would a reader looking at the rendered HTML notice this change if I didn’t tell them?” If yes, it is probably text. If the *numbers* change, it is analytic.

## Before / after examples

### Before / after: a label

**Before:** `selectInput("strat_var1", "Primary Grouping (Rows):", ...)`

**After:** `selectInput("strat_var1", "Compare across (rows of the chart):", ...)`

The label is friendlier to a non-technical audience. The variable, choices, and computation are unchanged.

### Before / after: a caption

**Before:** `caption = "BMI by income"`

**After:** `caption = "Mean BMI by income group across five NHANES cycles. Adults age 20-80 in the prepared classroom dataset. Unweighted descriptive summary; records with missing BMI, income, or cycle are excluded."`

The caption now names the statistic, source, cohort, exclusions, and disclosure. The figure itself is unchanged.

### Before / after: a help text line

**Before:** (no help text near the control)

**After:** Right under the control: `> This dashboard uses adults age 20-80. To see a different income group, change the value and re-render.`

The audience now understands what the artifact covers and how to use it.

## Before / after: a README sentence

**Before:** # Dashboard

**After:** ‘# NHANES Health Equity Classroom Dashboard

Descriptive, unweighted mean BMI by income group across NHANES cycles. For classroom learning; not for population inference.’

The README now names the audience and the boundary.

## What to record in your A7

### before-after-text.md

A7 asks you to submit one before-after text change.

Create `before-after-text.md` in the A7 submission folder. In the Explorer panel of your Codespace, expand `assignments/assignment07-dashboard/`, right-click the submission folder, choose **New File**, and name it `before-after-text.md`. Plain Markdown is fine — a short heading for each part below is enough.

For that file, paste:

- the **original text** verbatim,
- the **revised text** verbatim,
- one sentence on **why** the change helps the audience,
- one sentence confirming **no analytic logic was changed**.

That last sentence is what keeps the audit trail clean.

## Where this goes next

[dash05](#) covers how to test the starter dashboard locally, how to keep your publishing privacy-safe, and (briefly) the Shiny exemplar your instructor will demo in class.

# Testing, Privacy-Safe Publishing, and the Shiny Demo

## Where this fits

You have the starter dashboard ([dash03](#)) and you know what kind of changes are allowed ([dash04](#)). This page covers three things you do *before* you submit:

1. Test the starter dashboard in your Codespace.
2. Apply the privacy-safe publishing checklist.
3. Watch (but do not run) the optional Shiny exemplar your instructor will demo in class.

## Part 1 — Testing the starter dashboard

The A7 deliverable list asks for a `testing-log.md`. Use the routine below for the entry.

### A six-step test routine

1. **Render from the personal-workspace root.** Copy the workspace starter to `assignments/assignment07-dashboard/submission/dashboard.qmd`, open that submitted copy, and render it from the Command Palette (**Quarto: Render Document**). Confirm there are no errors.
2. **Change the control.** Edit `selected_income` to a different value from `available_income_groups`. Save the file.

3. **Render again.** Confirm the title, table, and plot all updated.
4. **Verify the caption.** The caption should still name the dataset, exclusions, and limitation.
5. **Open the HTML preview.** If no preview opened automatically, find the rendered `.html` file in the Explorer (the render log's `Output created:` line names it), right-click it, and choose **Open Preview** — or **Download** to open it in your browser. If **Open With** only offers a text editor showing raw HTML code, the render still worked — use **Open Preview** or **Download** instead. Confirm the rendered page looks correct in the browser, not just in the editor.
6. **Record the test.** In `testing-log.md`, note the date, the command or render path used, the values you tried, and the result.

#### How to render: the course default

1. Open the `.qmd` file you want to render, then open the **Command Palette** (**View** → **Command Palette**, or press **Ctrl+Shift+P** on Windows/Linux / **Cmd+Shift+P** on Mac) and run **Quarto: Render Document**. The Quarto extension renders the file and opens a preview pane.
2. **Always-works fallback (terminal):** type the render command with the file's path as shown in the Explorer, for example:

```
1 quarto render practice_report.qmd
```

Always name the file — a bare `quarto render` rebuilds the *whole* project and takes much longer.

3. If you see a **Render** (or **Preview**) button in the editor toolbar, it does the same thing as step 1.

**Where did the output go?** Watch the render log for the `Output created:` line — it names the exact `.html` file created. By default the file appears next to your `.qmd`;

in projects that set an output directory (like this book’s `docs/` folder), it appears there instead. If no preview opened automatically, find that `.html` file in the Explorer, right-click it, and choose **Download** to open it in your browser (or **Open Preview** if available).

HTML is the expected output for the dashboard — a PDF is not needed. Ignore TeX- or PDF-related warnings as long as the `.html` file was created; do not try to install TeX.

### Common failure modes and quick fixes

| Failure                          | What you see                                                     | Quick fix                                                                                                |
|----------------------------------|------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------|
| Wrong working directory          | “Could not find examples/nhanes-equity/data/nhanes_ExplorerCons” | Open the Quarto file via the Explorer pane, confirm Codespaces is rooted at the repo root                |
| Missing package                  | “there is no package called ‘ggplot2’ ”                          | Run <code>library(ggplot2)</code> once; if it still fails, raise with your TA — do not install at random |
| Stale <code>_freeze</code> cache | Output reflects the old control value                            | Delete the relevant <code>_freeze/</code> subfolder and re-render; or change the chunk slightly          |
| Preview pane blank               | Render succeeds but preview is empty                             | Click the preview pane refresh icon, or open the HTML file from <code>docs/</code> (if configured)       |

### Package check before you finish

The starter uses only packages in the course devcontainer. If an additional package is approved, record it in the workspace setup and your dependency note, then repeat the test in a fresh Codespace.

## Part 2 — Privacy-safe publishing

Five rules apply every time you share a dashboard rendering, a screenshot, or a Quarto preview.

1. **No row-level data.** Your figures and tables show aggregate summaries. Never paste a row-level NHANES record into a screenshot, a chat with an external AI, or a public README.
2. **No small-cell interpretations.** If a group has very few records, do not interpret the cell. Combine, mark, or exclude the group (see [eda04](#)).
3. **State descriptive limits.** Every interpretation should say the artifact is descriptive and unweighted. No causal language. No “national prevalence” language.
4. **Keep cached data separate from raw retrieval.** The course uses a cached CSV intentionally. If you write code that retrieves raw NHANES from the CDC, document that as separate work — do not put a live API call in your dashboard.
5. **Do not upload row-level data to external AI tools.** Aggregate first. The AI conversation can see counts and means; it should not see individual records.

### Three of those terms, with examples

- **Row-level data** is one person’s record. “Participant 93704: age 47, BMI 31.2, Low Income” is row-level. “n = 214, mean BMI 29.1” is an aggregate summary and is safe to show.



- **A small cell** is a group with very few people behind it — for example, a table cell where  $n = 3$ . With numbers that small, a person could be identifiable and the estimate is unstable, so combine, mark, or exclude the group instead of interpreting it.
- **Causal overreach** is wording that claims the data shows cause. “Low income causes higher BMI” is overreach. “Mean BMI was higher in the low-income group in this classroom dataset” stays descriptive and is fine.

### What to write in `privacy-check.md`

A6 and A7 both expect a short privacy check. For A7, write three sentences:

- One naming the data source and the classroom-use boundary (reuse from [eda05](#)).
- One confirming the dashboard renders aggregate summaries only.
- One confirming you did not share row-level records with any external AI tool.

## Part 3 — The Shiny exemplar (demonstration only)

 Demo only — not graded

This pathway is a **demonstration only**. It is not graded and not required. Using it for graded work requires **instructor approval first**. The supported, graded pathway in this course is the Quarto workflow.

Your instructor will demo a Shiny app version of the NHANES dashboard in class. **You are not required to run this app, install Shiny, or produce anything from it.** Your A7 deliverable is built from the Quarto starter dashboard, not from this app.

What the demo will show you:

- How a Shiny app turns the same kind of analysis into a live, server-backed dashboard with reactive controls.
- The trade-offs: more interactivity, but it requires a server (or a Codespaces port forward), more packages, and more failure modes.
- Where the app code lives (`examples/nhanes-equity/app/app.R`) if you want to read it.

If, after watching the demo, you decide you want to use Shiny for your term-project final dashboard, that pathway is **not approved by default**. Talk with your instructor before committing. The Quarto pathway is the supported default for every student.

 Warning

Do **not** put `shiny::runApp(...)` in your A7 submission as your “dashboard.” A7 expects an adaptation of the Quarto starter. The Shiny app is shown to you; it is not what you build on.

## Where this goes next

[dash06](#) is the in-class studio where you produce draft versions of every A7 deliverable from the Quarto starter.

# In-Class Studio: Dashboard Adaptation

## Where this fits

You have everything you need: the worked Quarto starter ([dash03](#)), the rule for allowed changes ([dash04](#)), and the testing and privacy routines ([dash05](#)). This page is the in-class studio — about 60 minutes of structured work that produces draft versions of every Assignment 7 deliverable.

## Before you start

Take 2 minutes to confirm your Codespace is ready:

1. The cached CSV resolves: `examples/nhanes-equity/data/nhanes_equity_v6.csv`.
2. You copied the personal-workspace starter to `assignments/assignment07-dashboard/submission/dashboard` and that copy renders cleanly with the default `selected_income`.
3. You have decided which `IncomeGroup` to set `selected_income` to for your adaptation (it does not have to be the default).

## Goal

Produce draft versions of every A7 file from the Quarto starter. Tomorrow's polish will make them assignment-ready; today's work makes sure the bones are there.

## The 60-minute studio sequence

You will move through six short steps. Each step produces one or two artifacts.

### Step 1 — Audience and question (10 min)

Write two short paragraphs in `dashboard-adaptation-note.md`:

- **Audience.** One specific role and what they need to decide.
- **Question.** Phrased the way the audience would phrase it.

Use the templates from [dash02](#). Be specific.

### Step 2 — Identify your control and visualization (5 min)

In the same note, add:

- **Control.** One control and the values it can take.
- **Visualization.** One chart and what it shows when the control changes.

For most students, this is the existing `selected_income` control and the line chart of mean BMI by cycle from the starter. That is fine.

### Step 3 — Pick one audience-facing change (10 min)

Read your starter file with the eyes of your stated audience. Find one element that could be clearer — a label, a caption, a help text line, a README sentence, or a limitation. Pick the one that would help the audience most.

Create `before-after-text.md` in the A7 submission folder: in the Explorer panel, expand `assignments/assignment07-dashboard/`, right-click the `submission` folder, and choose **New File**. Include the original wording, your revised wording, and one

sentence explaining why the revision is clearer for the audience.  
In full, the file needs:

- The original text.
- The revised text.
- One sentence on why the change helps.
- One sentence confirming no analytic logic changed (see [dash04](#)).

#### Step 4 — Apply the change and test (15 min)

Edit `assignments/assignment07-dashboard/submission/dashboard.qmd` with the revised text. Then run the six-step test routine from [dash05](#):

1. Render.
2. Change the control to a different value.
3. Render again.
4. Verify the caption.
5. Open the HTML preview.
6. Record the test in `testing-log.md`.

#### Step 5 — Write the limitation and privacy check (10 min)

In `limitation-note.md`, write **one sentence** stating what the dashboard does *not* support. Use the pattern from [eda05](#) on descriptive-vs-causal.

In `privacy-check.md`, write the **three sentences** from [dash05](#):

- One naming the data source and classroom-use boundary.
- One confirming aggregate-only outputs.
- One confirming no row-level data was shared with external AI.

`testing-log.md`, `limitation-note.md`, and `privacy-check.md` are brief evidence files. One to three focused sentences or bullets are enough unless the assignment asks for more.

## Step 6 — Reproducibility and AI use (10 min)

In `reproducibility-note.md`, write one paragraph naming:

- The command or render path you used.
- The relative path to the dataset.
- A sentence on whether the dashboard renders in a fresh Codespace.

In `ai-use-note.md`, name what AI helped with (if anything) and how you verified the result.

## At the end of the studio

You should leave class with draft versions of all the files A7 expects. Polish at home and submit by the Monday 4 PM deadline.

## Required studio outputs (drafts)

- `dashboard-adaptation-note.md`
- `dashboard.qmd`
- `before-after-text.md`
- `testing-log.md`
- `limitation-note.md`
- `privacy-check.md`
- `reproducibility-note.md`
- `ai-use-note.md`
- One screenshot of the rendered “after” state (this becomes `dashboard-preview.png` after a final render)

## Completion check

You are done with the studio when a peer can read your `dashboard-adaptation-note.md` and identify your audience, your question, your control, your visualization, your

text change, and your test result — without rerunning the dashboard first.

## **Where this goes next**

[dash07](#) walks through how each studio file becomes the corresponding A7 deliverable and what verification each file needs before submission.

# Assignment 7 Walkthrough and Reference

## Where this fits

You have studio drafts from [dash06](#). This page is the bridge from studio work to a submitted Assignment 7. The deliverables here mirror the [A7 README](#) exactly — this page explains what goes into each file, how to verify it, and how the work connects to your term-project final dashboard.

## Deadline and the decision rule

Assignment 7 is due the Monday after Week 8 class, 4 PM. The required pathway is the **Quarto starter dashboard** ([dash03](#)).

**Decision rule pinned at the top:** adapt the Quarto starter. The Shiny app is a demonstration only and is **not** an approved A7 submission pathway. If you are interested in using Shiny for your final-portfolio dashboard, talk with the instructor — Shiny is not a default-approved pathway and requires instructor approval.

## The nine A7 deliverables, explained

Each file goes in `assignments/assignment07-dashboard/submission/`.

`testing-log.md`, `limitation-note.md`, and `privacy-check.md` are brief evidence files. One to three focused sentences or bullets are enough unless the assignment asks for more.



All nine files are [required]. The [A7 README](#) marks the brief ones with [brief — 1-3 focused bullets or sentences are enough]; the full tag vocabulary is in [How To Read Submission Lists](#).

### 1. `dashboard.qmd`

The adapted copy of the personal-workspace Quarto starter. Make the allowed audience-facing text change here and retain the source with the evidence files.

- **Verify:** render this exact submitted file in a fresh Codespace.

### 2. `dashboard-adaptation-note.md`

Audience, question, control, visualization, the element you changed, and the KT rationale. Pattern after the studio Step 1 and Step 2 outputs.

- **Verify:** a peer can read this and identify your audience and question without seeing your code.

### 3. `before-after-text.md`

The original text, the revised text, why the change helps the audience, and a confirmation that no analytic logic changed. Pattern after the studio Step 3 output.

- **Verify:** the change is genuinely audience-facing text. If you cannot say “no analytic logic changed”, the change does not belong in A7 (see [dash04](#)).

### 4. `testing-log.md`

Date tested, the command or render path you used, the values you tried, and the result. Pattern after the studio Step 4 output.

- **Verify:** another student could reproduce your test by following your log.

## 5. `limitation-note.md`

One sentence on what the dashboard does not support. Use the descriptive-vs-causal patterns from [eda05](#).

- **Verify:** the sentence belongs *in or beside* the dashboard, not just in a separate file.

## 6. `privacy-check.md`

Three sentences from [dash05](#): data source and classroom-use boundary, aggregate-only confirmation, no row-level data shared with AI.

- **Verify:** three sentences, each specific.

## 7. `dashboard-preview.png`

A screenshot or exported preview of the affected dashboard area (after the text change).

- **Verify:** the preview clearly shows the revised audience-facing element. No row-level table previews.

## 8. `reproducibility-note.md`

One paragraph naming the render command, the relative path, and whether the dashboard renders in a fresh Codespace.

- **Verify:** the path is relative, and the note matches what `testing-log.md` says.

## 9. `ai-use-note.md`

What AI helped with (or a sentence saying you did not use it) and how you verified the result.

- **Verify:** specific (which prompt? which file?), not vague.

## Reproducibility checklist for A7

Before you click submit:

- The adapted Quarto file renders in a fresh Codespace.
- The data path is relative (`examples/nhanes-equity/data/nhanes_equity_v6.csv`).
- The preview screenshot shows the revised text change clearly.
- The dashboard contains a visible limitation sentence (not just in a separate file).
- No row-level data appears in any screenshot or shared output.
- All nine files are committed and pushed.

## A pathway decision template you can reuse

When the term-project conversation comes up later in the term, your team will face the same pathway choice for the final portfolio. Use this short template to document the decision:

Our audience is \_\_\_\_\_. We need \_\_\_\_\_ kind of interactivity. The simplest pathway that serves this need is [**Quarto dashboard-style page / Shiny app / static HTML/JS**]. [If Shiny or static HTML/JS:] We have instructor approval recorded on \_\_\_\_\_.

Default to Quarto. Justify deviations.

## Privacy-safe publishing checklist

Reused from [dash05](#) for your reference:

- No row-level data in screenshots, AI chats, or public README.
- No small-cell interpretations.
- Descriptive limits stated; no causal language; no national-prevalence language.

- Cached classroom data kept separate from any raw retrieval workflow.
- Row-level data never uploaded to external AI tools.

## Paths you will use repeatedly

| What                                  | Where                                                                            |
|---------------------------------------|----------------------------------------------------------------------------------|
| Cached dataset (CSV)                  | <code>examples/nhanes-equity/data/nhanes_equity_v6.csv</code>                    |
| Cached dataset (RDS)                  | <code>examples/nhanes-equity/data/nhanes_equity_v6.rds</code>                    |
| Student starter dashboard             | <code>assignments/assignment07-dashboard/dashboard-starter.qmd</code>            |
| Adapted submitted source              | <code>assignments/assignment07-dashboard/submission/dashboard.qmd</code>         |
| Shiny exemplar (instructor demo only) | <code>examples/nhanes-equity/app/app.R</code>                                    |
| Adaptation template                   | <code>assignments/assignment07-dashboard/dashboard-adaptation-template.md</code> |
| Submission folder                     | <code>assignments/assignment07-dashboard/submission/</code>                      |

## Five quick knowledge checks

1. Name three differences between a static report and a dashboard.
2. Which pathway is the required core for A7?
3. In one sentence, why is the Quarto starter dashboard preferred over editing the Shiny app for A7?
4. Name two changes that count as “UI-text-only.” Name two that do not.
5. What three sentences go in `privacy-check.md`?

## Where this goes next

Next week ([Week 9](#)) takes one finding from your dashboard and turns it into a concise scientific communication product. The cleaner your A7 limitation sentence is, the easier the Week 9 plain-language summary will be.

## **Week 9: Communication of Scientific Findings**

# Week 9: Communication of Scientific Findings

## Overview

Science communication means explaining scientific evidence clearly and responsibly for a specific audience. In this course, that means turning one audited health data finding into a short, plain-language message that includes evidence, limitations, and appropriate caution.

This week turns one audited result into a short scientific communication product. The goal is not to make a slide deck bigger. The goal is to make one claim easier to understand, easier to verify, and harder to overstate.

You will reuse work you already have: a corrected plot from Assignment 5, a Table 1 or methods note from Assignment 6, and a dashboard adaptation or limitation statement from Assignment 7.

### ! Required / Draft / Optional This Week

- **Required (graded):** the [Assignment 8](#) files, explained in [comm05](#), plus the peer-feedback exchange in [comm04](#).
- **Draft (class work that feeds A8, not graded on its own):** the claim-evidence-limitation table and the 3-5 slide draft from the class plan below.
- **Optional (no penalty for skipping):** extra slides beyond the 3-5 slide story — the deck and the outline are graded identically, so do not pad.

Tags like [required] and [optional] are defined in [How To Read Submission Lists](#).

## Objectives

By the end of the week you can:

- identify an audience and decision context for one result;
- write a claim that includes evidence and a visible limitation;
- build or outline a short Quarto reveals slide story;
- cite the data source for a figure, table, or dashboard screenshot;
- use peer feedback to improve clarity without inflating the claim.

### Key Terms This Week

- **Knowledge translation (KT):** turning evidence into a form that helps a specific person understand it, question it, or act on it.
- **Decision context:** the decision, conversation, or question your audience is facing when they see your result.
- **Audience-facing text:** writing meant for your audience to read directly — not notes for yourself or your team.
- **Claim-evidence-limitation:** the three-part pattern used for every message in this course: what you say, what supports it, and what it cannot prove.
- **Plain-language takeaway:** the one sentence a non-expert should remember, written without jargon.

## Connection

Week 8 built a small dashboard-style KT product. Week 9 turns one dashboard finding, plot, or table into a concise slide story. Week 10 then expands the same evidence into a reproducible Quarto report.

## Dashboard Case Study: NHANES Equity

The **NHANES Health Equity Dashboard** is now the foreground case study. Use it as a small knowledge-translation product built on the cached NHANES equity dataset.

- App code: `examples/nhanes-equity/app/app.R`
- Cached data: `examples/nhanes-equity/data/nhanes_equity_v6.rds` and `examples/nhanes-equity/data/nhanes_equity_v6.csv`
- Optional rebuild script: `examples/nhanes-equity/scripts/build_nhanes_equity.R`

Run the dashboard from the repository root:

```
1 shiny::runApp("examples/nhanes-equity/app")
```

The app is offline-first: it loads the local RDS cache when available. Rebuilding from CDC uses `nhanesA`, requires internet access, and is optional unless assigned.

Companion materials:

- Week 10 report template: `weeks/week10-reporting-publishing/report-template.qmd`
- Week 11 static dashboard demo: `weeks/week11-portfolio-surgery/static-dashboard-ojs.qmd`

## Reading Order

Work through the pages in order. Each page asks you to reuse an artifact rather than start over.

1. [comm01 - Why Communication Is KT](#)
2. [comm02 - Claim, Evidence, Limitation](#)
3. [comm03 - Revealjs Starter](#)
4. [comm04 - Peer Feedback](#)
5. [comm05 - Assignment 8 Walkthrough](#)

The raw starter file is [starter-slides.qmd](#). Use it as a renderable template, not as a reading page.



## Class Plan

1. Choose one finding from A5, A6, or A7.
2. Write the audience and decision context in one sentence.
3. Build a claim-evidence-limitation table for that finding.
4. Draft a 3-5 slide story using the revealjs starter or a structured slide outline.
5. Exchange peer feedback focused on clarity, evidence, limitation, and source note.
6. Revise one slide sentence or limitation statement before submission.

## Student Output

By the end of class each student has a draft slide deck or outline, a plain-language summary sentence, a peer feedback note, a source note, and an AI-use note if AI helped with wording or structure.

## Links

- Assignment: [Assignment 8 - Scientific Communication](#)
- Portfolio component: [Presentation \(narrated deck\)](#)
- Reusable starter: [starter-slides.qmd](#)

## Definition of Done

- The slide story has a named audience.
- The main claim is supported by one visible plot, table, or dashboard output.
- The limitation appears on the slide or in nearby notes.
- The source note names the dataset or source.
- Peer feedback led to at least one concrete revision.

## **What Students Leave With**

A short, audience-facing communication product that can be reused later in the narrated presentation deck and final portfolio.

# Why Communication Is KT

## Where This Fits

You already have descriptive evidence. A plot, a Table 1, and a dashboard prototype can all be technically correct and still fail if the intended audience cannot tell what matters.

Science communication means explaining scientific evidence clearly and responsibly for a specific audience. In this course, that means turning one audited health data finding into a short, plain-language message that includes evidence, limitations, and appropriate caution.

Communication is part of knowledge translation. It turns evidence into a form that helps a person make sense of a question, ask a better follow-up, or avoid a bad interpretation.

## The Week 9 Task

This week asks for one small communication product:

- one audience;
- one question;
- one figure, table, or dashboard output;
- one plain-language takeaway;
- one limitation;
- one source note.

That is enough. More slides are not automatically better.

## Match the Message to the Audience

Before writing, identify your audience: experts may need methods, policymakers may need decision relevance, patients may need clear practical meaning, and the public may need plain language and careful context.

| Audience                        | What they usually need             | What that means for your message                                        |
|---------------------------------|------------------------------------|-------------------------------------------------------------------------|
| Experts (analysts, researchers) | Methods and definitions            | Name the data source, the variables, and how groups were formed         |
| Policymakers                    | Decision relevance                 | Lead with the takeaway and say what decision or discussion it informs   |
| Patients                        | Clear practical meaning            | Explain what the finding means in everyday terms, without causing alarm |
| General public                  | Plain language and careful context | Avoid jargon and keep the limitation right next to the claim            |

The finding and the evidence stay the same. What changes is the emphasis, the wording, and how much technical detail you show.

## What Counts as Evidence

Use work from earlier weeks:

| Earlier artifact            | How it can support Week 9                            |
|-----------------------------|------------------------------------------------------|
| A5 corrected plot           | Evidence slide or visual example                     |
| A5 caption                  | Source, statistic, exclusion, and limitation wording |
| A6 Table 1                  | Descriptive context or supporting table              |
| A6 methods note             | Data and cohort explanation                          |
| A6 provenance note          | Source and stewardship note                          |
| A7 dashboard adaptation     | Audience and KT purpose                              |
| A7 limitation/privacy check | Caution slide or speaker note                        |

## Keep the Claim Small

Good Week 9 claims sound like this:

In this prepared classroom dataset, mean BMI differs across income groups, but the comparison is descriptive and unweighted.

Weak Week 9 claims sound like this:

Income causes BMI differences.

The second claim is not supported by the course workflow. The first claim is cautious, understandable, and grounded in the evidence students have actually produced.

## AI Use

AI may help simplify wording, draft speaker notes, or suggest slide order. It must not invent sources, make results stronger than the code supports, or remove limitations to make a message sound cleaner.

## Check Yourself

Before moving on, answer:

- Who is the audience?

- What decision, conversation, or question does the communication support?
- Which prior artifact is the evidence?
- What limitation must remain visible?

# Claim, Evidence, Limitation

## The Core Pattern

Every scientific communication product in this course should keep three pieces together:

1. **Claim:** what you want the audience to understand.
2. **Evidence:** the plot, table, dashboard output, code result, or citation that supports it.
3. **Limitation:** what the evidence cannot prove.

If one piece is missing, the message is either hard to trust or too easy to overstate.

## Build the Table First

Before opening revealjs, make a small table.

| Piece      | Draft                                                                                |
|------------|--------------------------------------------------------------------------------------|
| Audience   | Public-health analyst<br>preparing a briefing                                        |
| Question   | How does mean BMI vary by<br>income group in the<br>classroom NHANES dataset?        |
| Claim      | Mean BMI differs across<br>income groups in the<br>prepared dataset.                 |
| Evidence   | A5 corrected plot and A6<br>Table 1                                                  |
| Limitation | Descriptive, unweighted,<br>classroom snapshot; not<br>causal or population-weighted |

| Piece       | Draft                                                             |
|-------------|-------------------------------------------------------------------|
| Source note | CDC/NCHS NHANES<br>public-use data, prepared for<br>classroom use |

This table is not an extra deliverable. It is a drafting tool for A8.

## Reuse Earlier Work

### From A5

Use the corrected plot and caption. The caption already names the statistic, source, exclusions, and limitation. Shorten it for a slide, but do not remove the limitation.

### From A6

Use the Table 1, methods note, and provenance/stewardship note. They help the audience understand who is included, which values are missing, and why the result is descriptive.

### From A7

Use the audience note and limitation/privacy check. They remind you that a KT product is designed for someone, not for everyone.

## Plain-Language Revision

Technical draft:

The grouped BMI means show heterogeneity across  
IncomeGroup strata.

Plain-language revision:



Average BMI is not the same across income groups in this classroom dataset.

Limitation sentence:

This is a descriptive, unweighted comparison and does not show that income causes BMI differences.

Here is a second example, this time about missing data.

Technical draft:

Rows with missing BMI values were excluded via `filter(!is.na(BMI))` prior to aggregation.

Plain-language revision:

Before calculating the group averages, we left out people whose BMI was not recorded.

Limitation sentence:

The averages describe only people with a recorded BMI, so groups with more missing values are less fully represented.

## What Not to Do

Avoid these common shortcuts:

- turning a descriptive pattern into a causal statement;
- hiding the limitation in tiny text;
- showing a figure without naming the data source;
- copying a dashboard screenshot that exposes row-level data;
- asking AI to “make it more persuasive” without checking accuracy.

## Slide-Level Check

For each slide, ask:

- What is the one claim?
- Where is the evidence?
- Where is the limitation?
- Is the source visible or easy to find?

If a peer cannot answer those questions, revise before submission.

# Revealjs Starter

## Where This Fits

This page explains how to use the slide starter shipped in your personal workspace at `assignments/assignment08-communication/starter-slides.qmd`. The copy in this course book is a reference; do not assume it appears in the same Explorer as your personal work.

You may also submit a structured slide outline for A8. Both options are graded identically — pick the format that fits where you are. The revealjs path is recommended if you want practice with Quarto slides before M5.

## Copy, Then Render

The workspace starter is configured for its submitted location. Copy and rename it first using the steps below, then render the submitted `slides.qmd`; do not render or edit the untouched template in place.

## Copy the Starter into Your A8 Folder

The starter stays in the weeks folder as a template. Your graded files belong in the assignment submission folder.

Before editing, copy `assignments/assignment08-communication/starter-slides.qmd` within your personal workspace into `assignments/assignment08-communication/submission/` and rename it `slides.qmd`. Render from the repository root so the required `slides.qmd` and `slides.html` are created in the correct folder.

Step by step in VS Code:

1. In your personal workspace Explorer, open `assignments/assignment08-communication/`, right-click `starter-slides.qmd`, and choose **Copy**.
2. Right-click the `assignments/assignment08-communication/submission/` folder and choose **Paste**.
3. Right-click the pasted copy, choose **Rename**, and type `slides.qmd`.
4. Edit `slides.qmd` there, then render it by name:

```
1 quarto render assignments/assignment08-communication/submission/slides.qmd
```

If you choose the outline path instead, name your file `slide-outline.qmd` in the same submission folder.

## The Three-Slide Story

The starter uses three content slides:

1. **Question and audience:** who needs the result and why.
2. **Evidence:** one plot, table, or dashboard output.
3. **Limitation and next step:** what the result does not prove and what the audience should do next.

For A8, 3-5 slides is enough. A short, honest deck is better than a long deck full of weak claims.

## Add Evidence

You can add evidence in three ways:

- insert a figure exported from A5;
- copy a small table from A6;
- include a screenshot or preview of the A7 dashboard output if it does not expose row-level data.

Use relative paths for images. For example:

```
1 ![../assignment05-visualization/submission/corrected-plot.png)
```

That example is relative to the A8 **submission/** folder. If the path does not work, keep the figure beside your slide deck and use a local relative path such as:

```
1 ![] (corrected-plot.png)
```

## Add a Source Note

The starter includes a citation to **@cdc-nhanes**. Keep a source note near the evidence slide. If you use your project dataset instead of the NHANES example, replace the source note with the source that matches your project.

## Speaker Notes

Speaker notes are useful, but they are not a place to hide essential limitations. Put the key limitation on the slide or in a visible caption. Use notes for backup details such as cohort definition, package versions, or AI-use checks.

## Before Submitting

Check that:

- the deck or outline renders;
- the main evidence is visible;
- the limitation is not hidden;
- the source note is present;
- the AI-use note says what AI helped with and what you checked.

# Peer Feedback

## Goal

Peer feedback helps you test whether a scientific message is clear, grounded, and honest before it reaches the final presentation or portfolio.

In Week 9, feedback is not about personal preference. It is about whether the audience can see the claim, evidence, limitation, and source.

## How and When to Exchange Feedback

Use the in-studio partner review as the pre-submission feedback source for A8. The Canvas Discussion exchange closes Monday after the Week 9 class at 4:00 PM, the same deadline as A8, and can support later presentation-deck or portfolio revision.

Use both channels:

1. **In-studio partner review (during the Week 9 class):** show your draft to a partner, complete the feedback table, and make at least one documented A8 revision before submitting.
2. **Canvas Discussion (by the standard deadline above):** post your plain-language summary and rendered slides or outline, then complete the feedback table for at least one classmate. Feedback arriving at the deadline is for later revision, not the required pre-submission A8 change.

You do not need to give classmates access to your repository. To share an in-progress deck, show your screen in studio, or download the rendered `slides.html` from the Explorer (right-click the file, then **Download**) and attach it to your Canvas Discussion post.

Feedback you receive can also inform revisions to this work for the final portfolio.

## Three-Slide Structure

Use this structure for a revealjs deck or slide outline:

1. **Question and audience:** who needs this finding, and why?
2. **Evidence:** show one plot, table, or dashboard output and state the descriptive pattern.
3. **Limitation and action:** name what the data can and cannot support, then state the next decision or question.

Use [starter-slides.qmd](#) if you want a renderable revealjs scaffold.

## Plain-Language Check

For each slide, ask:

- What is the one sentence the audience should remember?
- Which word or phrase would confuse someone outside the course?
- What limitation must remain visible to avoid overclaiming?
- Does the source note match the evidence shown?

## Peer Feedback Form

Copy this table into your A8 `peer-feedback.md`.

| Prompt                                     | Peer note |
|--------------------------------------------|-----------|
| Main message in one sentence               |           |
| Intended audience                          |           |
| Evidence shown                             |           |
| Limitation or caution shown                |           |
| Source note or citation shown              |           |
| Wording that needs plain-language revision |           |
| Accessibility issue, if any                |           |
| One concrete improvement before submission |           |

## Citation and Accessibility Checklist

- Data source is named.
- Figure or table has a clear title.
- Axes, units, and group labels are readable.
- Color choices work without relying on color alone.
- Slide text is brief enough to read during a talk.
- Limitation is visible.
- AI-assisted wording has been checked against the evidence.

## Good Feedback Sounds Like

Helpful:

The main claim is clear, but the limitation is only in speaker notes. Put the unweighted/descriptive caution under the figure.

Less helpful:

Looks good.



## Student Output

Submit the rendered slide deck or slide outline, completed peer feedback form, plain-language summary, source note, and AI-use note required by [Assignment 8](#).

# Assignment 8 Walkthrough

## Where This Fits

This page connects the Week 9 studio work to [Assignment 8](#). It does not add new deliverables. It explains what each required file should contain and how to check it before submitting.

Assignment 8 is due the Monday after Week 9 class, 4 PM. It also gives you reusable material for the final portfolio's narrated presentation deck.

## The Six A8 Files

Each file goes in `assignments/assignment08-communication/submission/`.

### 1. `slides.qmd` or `slide-outline.qmd`

Use one of these options:

- `slides.qmd`: a revealjs deck based on the personal-workspace `assignments/assignment08-communication/starter-slides.qmd`;
- `slide-outline.qmd`: a structured outline if you are not ready to render revealjs.

Both options are graded identically — choose the format that fits where you are. If you choose the outline for A8, convert the revised outline into a renderable `slides.qmd` for the presentation deck. The outline remains fully eligible for the A8 score; the deck is the later slide-production stage.

To create `slides.qmd`, copy the starter within your personal workspace from `assignments/assignment08-communication/`

into its **submission/** folder and rename it before editing. Render from the personal-workspace root so **slides.html** is created in the correct place. Step-by-step instructions are in [comm03](#).

Include 3-5 slides or sections: context, evidence, takeaway, limitation, and next step.

- **Verify:** the source file renders or is readable as a complete outline.

## 2. **slides.html or slide-outline.html**

This is the rendered output. If you use revealjs, the result should open in the browser as slides.

- **Verify:** open the rendered file and check that the evidence and limitation are visible.

## 3. **plain-language-summary.md**

Write 150-250 words for the intended audience. Use plain language, but keep the evidence honest.

Include:

- who the audience is;
- what the main result says;
- what the result does not prove;
- what action, question, or conversation should follow.

## 4. **peer-feedback.md**

Use the feedback table from [comm04](#). Include the feedback you received and one sentence explaining what you changed.

- **Verify:** at least one revision is tied to a peer comment.

## 5. `source-note.md`

Name the dataset or source behind the evidence. For the NHANES classroom example, the source note can say:

Source: prepared classroom snapshot derived from CDC/NCHS NHANES public-use data. Results are descriptive and unweighted.

For a term project, replace this with the correct project source.

## 6. `ai-use-note.md`

If AI helped draft, simplify, or reorganize your communication, state:

- what it helped with;
- what you changed;
- how you checked the claim against the evidence.

If you did not use AI, write one sentence saying so.

# Reproducibility Check

Before submission:

- the deck or outline source is committed;
- the rendered HTML is committed;
- any image paths are relative;
- the evidence can be traced to A5, A6, A7, or project files;
- the source note and limitation are visible;
- the AI-use note is specific.

## Accessibility Check

Slides should be readable in a live presentation. Check:

- titles are short;
- body text is large enough;
- figure labels are readable;
- color is not the only way to distinguish groups;
- the final slide gives the audience a clear next step.

## Where This Goes Next

Week 10 expands the same communication discipline into a report: the claim becomes a results paragraph, the evidence becomes a code-generated table or figure, and the limitation becomes part of the written interpretation.

## **Week 10: Writing & Publishing Reports**

# Week 10: Writing & Publishing Reports

## Overview

This week turns a slide-sized finding into a short reproducible report. A report does more than present a result. It shows where the data came from, how the output was generated, what the evidence supports, and what remains uncertain.

The default publishing pathway is a private preview unless the data, repository, and group privacy checks show that public sharing is safe.

If you want the big picture first, the [Private Preview](#) page opens with a workflow diagram showing how editing, rendering, pre-viewing, committing, and publishing fit together.

### ! Required / Draft / Optional This Week

- **Required (graded):** the [Assignment 9](#) files and the group [M3 project update](#), explained in [report05](#).
- **Draft (class work that feeds A9/M3, not graded on its own):** the report outline, rendered template, and grounding audit from the class plan below.
- **Optional (no penalty for skipping):** public publishing — the course default is a private preview ([report04](#)).

Tags like [required] and [optional] are defined in [How To Read Submission Lists](#).

## Objectives

By the end of the week you can:

- structure a short Quarto report around one descriptive question;
- generate one table and one figure from code;
- cite a data source using a BibTeX workflow;
- trace key claims to code output, citations, or limitations;
- choose a private or public preview route based on privacy and stewardship checks.

## Connection

Week 9 turned one finding into a concise slide story. Week 10 expands the same evidence into a reproducible report and M3 project update. Week 11 then stress-tests whether the report and KT product run in a fresh environment.

## Dashboard Case Study: NHANES Equity

The **NHANES Health Equity Dashboard** is now the foreground case study. Use it as a small knowledge-translation product built on the cached NHANES equity dataset.

- App code: `examples/nhanes-equity/app/app.R`
- Cached data: `examples/nhanes-equity/data/nhanes_equity_v6.rds` and `examples/nhanes-equity/data/nhanes_equity_v6.csv`
- Optional rebuild script: `examples/nhanes-equity/scripts/build_nhanes_equity.R`

Run the dashboard from the repository root:

```
1 shiny::runApp("examples/nhanes-equity/app")
```

The app is offline-first: it loads the local RDS cache when available. Rebuilding from CDC uses `nhanesA`, requires internet access, and is optional unless assigned.

Companion materials:



- Week 10 report template: `weeks/week10-reporting-publishing/report-template.qmd`
- Week 11 static dashboard demo: `weeks/week11-portfolio-surgery/static-dashboard-ojs.qmd`

## Reading Order

1. [report01 - Report Goals](#)
2. [report02 - Report Structure](#)
3. [Report Template](#)
4. [report03 - Citations and Grounding](#)
5. [report04 - Private Preview](#)
6. [report05 - Assignment 9 and M3](#)

## Class Plan

1. Start from one Week 9 claim or one M3-ready finding.
2. Draft a report outline: question, data, methods, results, limitations.
3. Render the [Report Template](#) and identify the table, figure, citation, and limitation.
4. Add or revise one citation/source note.
5. Complete a grounding audit for three claims.
6. Decide whether the preview should stay private or can be public-safe.

For step 3, always name the file you are rendering. In your personal workspace, use:

```
1 quarto render assignments/assignment09-reporting/submission/report.qmd
```

### How to render: the course default

1. Open the `.qmd` file you want to render, then open the **Command Palette** (**View** → **Command Palette**, or press **Ctrl+Shift+P** on Windows/Linux / **Cmd+Shift+P** on Mac) and run **Quarto: Render Document**. The Quarto extension renders the file and opens a preview pane.

2. **Always-works fallback (terminal):** type the render command with the file's path as shown in the Explorer, for example:

```
1 quarto render practice_report.qmd
```

Always name the file — a bare `quarto render` rebuilds the *whole* project and takes much longer.

3. If you see a **Render** (or **Preview**) button in the editor toolbar, it does the same thing as step 1.

**Where did the output go?** Watch the render log for the **Output created:** line — it names the exact `.html` file created. By default the file appears next to your `.qmd`; in projects that set an output directory (like this book's `docs/` folder), it appears there instead. If no preview opened automatically, find that `.html` file in the Explorer, right-click it, and choose **Download** to open it in your browser (or **Open Preview** if available).

## Student Output

By the end of class each group has a report outline, a rendered template or draft report, a citation/source note, three grounded claims, and an M3 feedback request.

## Links

- Assignment: [Assignment 9 - Reproducible Report](#)
- Milestone: [M3 - Project Update](#)
- Final portfolio: [Final Portfolio](#)

## Definition of Done

- The report source renders.

- One table and one figure are generated by code.
- Citations resolve.
- The limitation section avoids unsupported causal or population-level claims.
- The preview route protects data and group privacy.

## **What Students Leave With**

A short report draft and project update that can be reviewed in Week 11 for reproducibility, clarity, auditability, and KT value.

# Report Goals

## Where This Fits

Slides help an audience understand one message quickly. Dashboards help an audience explore one question. Reports show the audit trail: data, methods, code-generated evidence, interpretation, limitations, citations, and reproducibility.

Week 10 asks you to make that audit trail visible.

## Who Is the Report For?

Write the main report for a non-technical decision-maker: someone who could act on your finding but does not read code. Keep the body in plain language — question, evidence, what it means, what it does not prove. Put technical material such as code detail, package notes, and render instructions in a technical appendix at the end. This is the same audience as your A8 slides, so the two should feel consistent.

## The Report Is Not Bigger Slides

A report should answer:

- What question is being asked?
- Which data support the answer?
- What was included or excluded?
- Which outputs were generated by code?
- What does the result mean?
- What does the result not prove?
- How can another person rerun or inspect it?

## Reuse Earlier Work

The report should reuse artifacts instead of creating a new analysis from scratch.

| Earlier artifact   | Report role                   |
|--------------------|-------------------------------|
| A5 corrected plot  | Figure                        |
| A5 caption         | Figure caption and limitation |
| A6 Table 1         | Descriptive table             |
| A6 methods note    | Methods paragraph             |
| A6 provenance note | Data source paragraph         |
| A7 dashboard note  | Audience and KT context       |
| A8 slide claim     | Results paragraph             |

## Keep the Scope Descriptive

The term project does not require formal modelling, prediction, causal inference, or hypothesis testing. A strong report can be descriptive if it is transparent, reproducible, and useful for the intended audience.

## Report and M3

M3 is a project update, not a final report. The draft should show:

- what evidence is ready;
- what still needs feedback;
- how the dashboard-style product connects to the report;
- what reproducibility risks remain.

## Check Yourself

Before writing paragraphs, write one sentence:

This report answers \_\_\_\_\_ for \_\_\_\_\_ using  
\_\_\_\_\_, and the main limitation is \_\_\_\_\_.

If that sentence is hard to finish, narrow the report question.

# Report Structure

## A Beginner-Friendly Structure

Use this structure for A9 and M3 unless your project clearly needs a small adaptation:

1. Question
2. Data source
3. Methods
4. Results
5. Limitations
6. Reproducibility and AI-use note

The structure is simple on purpose. It gives readers enough information to understand, check, and reuse the work.

## Question

A good report question is specific:

How does mean BMI vary by income group among adults age 20-80 in the prepared NHANES classroom dataset?

Avoid broad questions such as:

What affects health?

## Data Source

Name:

- the original data source;
- the version or snapshot used;
- the access route;
- restrictions or stewardship cautions.

Use your A6 provenance/stewardship note as the starting point.

## Methods

State choices before showing results:

- cohort or inclusion criteria;
- variables used;
- missingness handling;
- whether the summary is descriptive, unweighted, weighted, predictive, or causal.

For this course, most reports should be descriptive unless the instructor explicitly approved another approach.

## Results

Include at least:

- one code-generated table;
- one code-generated figure;
- a short paragraph that states the pattern without over-stating it.

Do not paste screenshots when code can generate the output.



## Limitations

Limitations belong in the report, not only in an appendix. Include limitations that affect interpretation, such as unweighted summaries, missing data, prepared classroom data, small cells, or restricted public sharing.

## Reproducibility and AI Use

End with enough information for a reviewer:

- how the report was rendered;
- where data came from;
- where package/dependency notes live;
- what AI helped with;
- how AI-assisted content was checked.

# Report Template: NHANES Equity Summary

## Standalone YAML

If you copy this template into a project report outside the book, start the file with this YAML header:

```
1 ---
2 title: "NHANES Equity Summary"
3 format: html
4 bibliography: references.bib
5 ---
```

## Purpose

This runnable template shows how to pair a dashboard with a short reproducible report. The dashboard supports exploration; the report fixes one question, documents the data, and cites the source.

## Question

How does mean BMI vary by income group among adults age 20-80 in the prepared NHANES Health Equity classroom dataset?

## Data Source

The report uses the cached CSV snapshot at `examples/nhanes-equity/data/nhanes_equity_v6.csv`. The source data are derived from CDC/NCHS NHANES public-use files (Centers for Disease Control and Prevention, National Center for Health Statistics 2026).

```
1 library(tidyverse) # AI-EDIT(2026-06-23): tidyverse-default - consolidated core library() call
2 library(knitr)
3
4 candidate_paths <- c(
5   "examples/nhanes-equity/data/nhanes_equity_v6.csv",
6   "../..examples/nhanes-equity/data/nhanes_equity_v6.csv"
7 )
8
9 data_path <- candidate_paths[file.exists(candidate_paths)][1]
10 if (is.na(data_path)) {
11   stop("Could not find examples/nhanes-equity/data/nhanes_equity_v6.csv")
12 }
13
14 nhanes <- read_csv(data_path, show_col_types = FALSE)
```

## Methods

We restricted the classroom dataset to adults age 20-80 and summarized BMI by income group. The summaries are un-weighted and descriptive. They should not be interpreted as causal effects or national population estimates.

```
1 analysis_df <- nhanes |>
2   filter(Age >= 20, Age <= 80)
3
4 missingness <- analysis_df |>
5   summarise(
6     rows = n(),
7     missing_bmi = sum(is.na(BMI)),
8     missing_income = sum(is.na(IncomeGroup)),
9     missing_age = sum(is.na(Age))
10  )
```

```
kable(missingness, caption = "Missingness check for key report variables.")
```

Table 1: Missingness check for key report variables.

| rows  | missing_bmi | missing_income | missing_age |
|-------|-------------|----------------|-------------|
| 57137 | 1009        | 5395           | 0           |

## Results

```
income_summary <- analysis_df |>
  filter(!is.na(BMI), !is.na(IncomeGroup)) |>
  group_by(IncomeGroup) |>
  summarise(
    n = n(),
    mean_bmi = mean(BMI),
    sd_bmi = sd(BMI),
    .groups = "drop"
  ) |>
  mutate(
    `BMI, mean (SD)` = sprintf("%.1f (%.1f)", mean_bmi, sd_bmi)
  ) |>
  select(IncomeGroup, n, `BMI, mean (SD)`)

kable(income_summary, caption = "Descriptive BMI summary by income group.")
```

Table 2: Descriptive BMI summary by income group.

| IncomeGroup        | n     | BMI, mean (SD) |
|--------------------|-------|----------------|
| High Income (>3.5) | 16330 | 28.6 (6.3)     |
| Low Income (<1.3)  | 15293 | 29.4 (7.5)     |
| Middle Income      | 19228 | 29.3 (7.0)     |

```
plot_df <- analysis_df |>
  filter(!is.na(BMI), !is.na(IncomeGroup)) |>
```

```

3 group_by(IncomeGroup) |>
4 summarise(
5   n = n(),
6   mean_bmi = mean(BMI),
7   .groups = "drop"
8 )
9
10 ggplot(plot_df, aes(x = IncomeGroup, y = mean_bmi)) +
11   geom_col(fill = "#2C7FB8") +
12   geom_text(aes(label = paste0("n=", n)), vjust = -0.4, size = 3.5) +
13   labs(
14     title = "Mean BMI by income group",
15     subtitle = "Adults age 20-80; unweighted descriptive summary",
16     x = "Income group",
17     y = "Mean BMI"
18   ) +
19   expand_limits(y = 0) +
20   theme_minimal(base_size = 12)

```

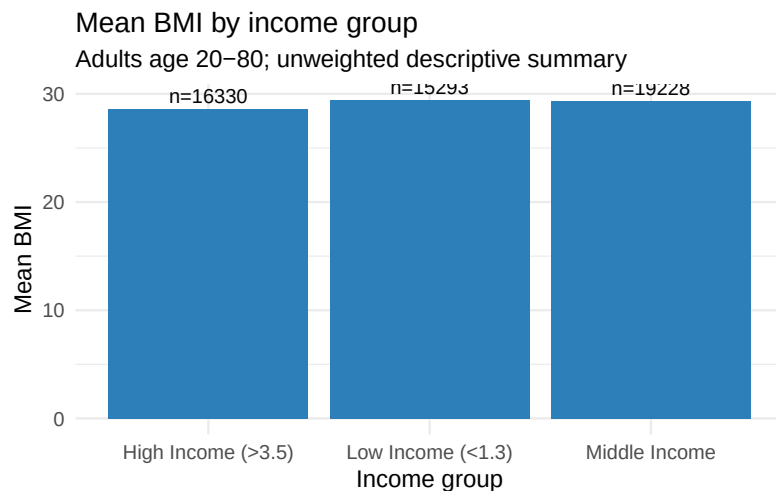


Figure 2: Mean BMI by income group among adults age 20-80 in the prepared NHANES Health Equity classroom dataset. Values are unweighted and descriptive.

## Limitations

- The table and figure use a prepared classroom dataset rather than raw NHANES files.
- The summaries are unweighted.
- The comparison is descriptive and does not establish causality.
- Missing BMI or income group values are excluded from the grouped summary.

## Plain-Language Summary

In this classroom dataset, mean BMI differs across income groups. The result is useful for practicing reproducible reporting and dashboard interpretation, but it should be described as a descriptive pattern rather than a causal finding.

## AI-Use and Audit Note

Document any AI assistance used to draft code, captions, or interpretation. State exactly which output you checked against the code, data dictionary, or rendered report.

## Publishing Checklist

- Report renders from source.
- Citations resolve.
- Figures and tables are generated by code.
- Private preview is used unless the data, repository, and group privacy checks support public sharing.
- Public preview contains no restricted data, private links, credentials, or sensitive row-level examples.
- Staged files were reviewed before committing: no raw row-level data, credentials, `.RData` workspaces, or stray generated files, and `.gitignore` covers the data folder and rendered outputs — see the [pre-commit privacy checklist](#).

- Private repository access is granted to course staff when needed.
- Canvas submission includes the repository link, report path, and latest commit hash.

## **Grounding Audit Prompt**

Select three claims from the report. For each claim, write the supporting evidence source: code chunk, table, figure, or citation. Revise any claim that cannot be traced to evidence in the report.

# Citations and Grounding

## Why This Matters

Reports can sound confident even when claims are weakly supported. A grounding audit forces each important claim to point back to a source: code output, a table, a figure, a citation, or a stated limitation.

## Citation Workflow

The course uses BibTeX references through `references.bib`. A report YAML header can include:

```
1 ---
2 title: "Project Report"
3 format: html
4 bibliography: ../../references.bib
5 ---
```

Then cite a source in text:

```
1 The NHANES classroom example is derived from CDC/NCHS NHANES public-use data [@cdc-nhanes].
```

If your project uses a different source, add the relevant bibliography entry to the report's local `references.bib` or to the project bibliography.



## Grounding Audit

For A9, choose at least three claims and trace them.

| Claim                             | Evidence type | Evidence location              | Decision                       |
|-----------------------------------|---------------|--------------------------------|--------------------------------|
| Mean BMI differs by income group. | Figure/table  | Results plot and summary table | Keep, with descriptive wording |
| Income causes BMI differences.    | None          | Not supported                  | Remove or rewrite              |
| The summary is unweighted.        | Methods/code  | Methods paragraph and code     | Keep                           |

## What Counts as Evidence

Evidence can be:

- a code-generated table;
- a code-generated figure;
- a data source citation;
- a methods paragraph;
- a limitation statement;
- a reproducibility log.

Evidence is not:

- an AI-generated sentence without verification;
- a visual impression with no source;
- a claim copied from a website without citation;
- a dashboard interaction that cannot be reproduced.

## AI-Use Check

If AI helped draft report prose, audit the generated text for:

- invented sources;
- stronger claims than the code supports;
- causal language;
- missing limitations;
- mismatched variable names or statistics.

## **Before Moving On**

If a claim cannot be grounded, do one of three things:

- add the missing evidence;
- soften the wording;
- remove the claim.

# Private Preview

## How the Pieces Fit Together

Quarto turns your source file into HTML. GitHub stores both. The private preview route stops there: course staff open the committed HTML or render the report in a fresh Codespace. GitHub Pages is an optional last step that turns the rendered HTML into a public website.



Figure 3: The publishing workflow: edit `report.qmd`, render with Quarto, preview the HTML, then commit and sync. A privacy check routes to a private preview by default, or optionally to a public GitHub Pages site.

## Default Path

For this course, publishing usually means creating a rendered, shareable output for grading or private review. Public publishing is optional and should only happen after privacy and safety checks.

The default publishing route is a private preview unless the project passes privacy and stewardship checks for public sharing.

Private preview means:

- the report renders to HTML;
- the rendered file is committed if the assignment asks for it;

- the repository link is submitted on Canvas;
- course staff have access if the repository is private.

This is enough for A9 and M3.

#### **i** Commit required rendered artifacts

The submission list is authoritative. If an assignment or milestone explicitly requires a rendered HTML or PDF, commit it with the source so staff can inspect the work. Do not add global `*.html` or `*.pdf` patterns to the course workspace `.gitignore`.

## Pre-Commit Privacy Checklist

Run this check every time, before you commit and sync:

1. Open the Source Control panel and read the list of staged files.
2. Unstage anything that should not go online: raw restricted or row-level data, credentials or tokens, `.RData` workspaces, OS junk such as `.DS_Store`, and unrelated generated files. Keep every rendered artifact named in the assessment brief staged.
3. Open `.gitignore` and confirm it covers secrets, machine-specific files, and temporary support folders without globally excluding required HTML or PDF. If an unwanted file keeps reappearing, add a narrow rule for that file type or path.
4. Skim the rendered HTML one last time for private URLs, tokens, or row-level records in printed output.

If any item fails, fix it before syncing. It is much easier to keep a file out of Git than to remove it after it has been pushed.

## When Public Publishing Is Safe

GitHub Pages or another public preview is appropriate only when:

- the data are public and allowed to be shared;
- no restricted raw data are committed;
- no sensitive row-level records appear in outputs;
- no private URLs, tokens, or credentials appear;
- the group agrees that public sharing fits the project context;
- the README explains data source and sharing limits.

If any item is uncertain, keep the preview private.

#### GitHub Pages sites are always public

A GitHub Pages site is public on the open internet even when the source repository is private. Turning on Pages publishes the rendered content for anyone to see. If your project must stay private, the “private equivalent” is course staff previewing in a fresh Codespace or reading the committed HTML in your private repository — not a Pages site.

## GitHub Pages as an Optional Extension

#### Demo only — not graded

This pathway is a **demonstration only**. It is not graded and not required. Using it for graded work requires **instructor approval first**. The supported, graded pathway in this course is the Quarto workflow.

GitHub Pages is useful for public portfolios, but it is not required for every student project. Public publishing should never outrank privacy, data-use rules, or stewardship obligations.

If your group uses GitHub Pages, record:

- the public URL;
- the commit hash;
- what was checked before publishing;
- what data or outputs are intentionally excluded.

## Private Preview Checklist

For the default route, record:

- report path;
- rendered HTML path;
- repository link;
- latest commit hash;
- who has access;
- any files intentionally excluded from public sharing.

## Common Errors and Fixes

Most rendering and publishing problems this week have boring causes. Check this table before asking for help.

| What you see                                                  | Likely cause                                        | Fix                                                                                                                                          |
|---------------------------------------------------------------|-----------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| <code>quarto: command not found</code>                        | The terminal opened before Codespace setup finished | Open a new terminal and retry; if it persists, rebuild the Codespace                                                                         |
| Render stops with red text naming an R package                | The package is not installed in this environment    | Read the first error line for the package name, then follow the week's setup steps; do not change the environment beyond that without asking |
| <code>cannot open file or ... does not exist</code> for a CSV | Absolute path, or wrong working folder              | Make the path relative to the project root and re-render (see the <a href="#">path check on the Assignment 9 and M3 page</a> )               |

| What you see                                          | Likely cause                                   | Fix                                                                                                                                                          |
|-------------------------------------------------------|------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Render finishes but no preview opens                  | The preview pane did not launch                | Find the <code>.html</code> file named in the <b>Output created:</b> line, right-click it in the Explorer, and choose <b>Download</b> or <b>Open Preview</b> |
| Citations appear as <code>[@key]</code> in the output | Bibliography file not found, or key misspelled | Check the <b>bibliography:</b> path in the YAML header and the key spelling in <b>references.bib</b>                                                         |
| Pages site shows a 404 or old content                 | Pages is not enabled, or is still deploying    | Wait a few minutes and refresh; check the repository's <b>Settings</b> → <b>Pages</b> ; remember Pages is optional                                           |

If the fix is not obvious after a careful read of the first error line, copy the exact error message into your help request.

## Final Portfolio Connection

The same decision returns in the final portfolio. Public repositories can be excellent portfolio artifacts, but private repositories are the right choice when sharing would create privacy, stewardship, or licensing risk.

# Assignment 9 and M3

## Where This Fits

This page connects Week 10 class work to [Assignment 9](#) and [M3: Project Update](#). It does not add new deliverables. It explains how the same work can support both.

## A9 Submission Files

Each A9 file goes in `assignments/assignment09-reporting/submission/`.

### `report.qmd`

A rerunnable Quarto report with question, data, methods, results, limitations, citations, and AI-use or audit notes.

- **Verify:** render from the repository root or clearly documented project folder.

To render it, always name the file, for example: `quarto render assignments/assignment09-reporting/submission/report.qmd`.

#### How to render: the course default

1. Open the `.qmd` file you want to render, then open the **Command Palette** (**View** → **Command Palette**, or press **Ctrl+Shift+P** on Windows/Linux / **Cmd+Shift+P** on Mac) and run **Quarto: Render Document**. The Quarto extension renders the file and opens a preview pane.
2. **Always-works fallback (terminal):** type the ren-



der command with the file's path as shown in the Explorer, for example:

```
1 quarto render practice_report.qmd
```

Always name the file — a bare `quarto render` rebuilds the *whole* project and takes much longer.

3. If you see a **Render** (or **Preview**) button in the editor toolbar, it does the same thing as step 1.

**Where did the output go?** Watch the render log for the **Output created:** line — it names the exact `.html` file created. By default the file appears next to your `.qmd`; in projects that set an output directory (like this book's `docs/` folder), it appears there instead. If no preview opened automatically, find that `.html` file in the Explorer, right-click it, and choose **Download** to open it in your browser (or **Open Preview** if available).

## report.html

The rendered report.

After rendering, open the HTML file in browser preview. Confirm that the title, table, figure, citations, limitations, and references appear correctly before committing your work.

Yes, commit this HTML file. The course rule is simple: when an assessment explicitly lists a rendered artifact, commit it with its source.

- **Verify:** table, figure, citations, and limitations appear.

## references.bib

Bibliography entries used by the report. This may be a copied subset of the project bibliography if that makes the submission easier to inspect.

- **Verify:** citations resolve during render.

### **outputs/**

Use this folder only if your report writes external files such as CSV tables or exported figures.

- **Verify:** outputs are generated by code, not manually edited.

### **publishing-checklist.md**

Record either the private preview route or the public publishing route. Private preview is the default unless public sharing is safe.

### **grounding-audit.md**

Trace at least three claims to evidence. Use [report03](#) as the model.

### **m3-update-note.md**

State what is ready for project update review and what feedback would help most.

### **ai-use-note.md**

Name what AI helped with and how the group checked the result.

## Edit These, Not Those

Edit the source `.qmd` and supporting text files. Do not manually edit generated HTML files, cached outputs, raw data, credentials, `.RData` workspaces, or system/configuration files unless instructed.

- **Edit:** `report.qmd`, `references.bib`, and the notes (`publishing-checklist.md`, `grounding-audit.md`, `m3-update-note.md`, `ai-use-note.md`).
- **Do not hand-edit:** `report.html` (re-render instead), files in `outputs/` (regenerate them from code), raw data files, and configuration files such as `_quarto.yml`.

## M3 Submission Files

M3 uses `milestones/m3-project-update/submission/`.

The A9 report draft can support M3:

| A9 artifact                          | M3 connection                                 |
|--------------------------------------|-----------------------------------------------|
| <code>report.qmd</code>              | <code>report-draft.qmd</code>                 |
| <code>report.html</code>             | <code>report-draft.html</code>                |
| <code>publishing-checklist.md</code> | <code>reproducibility-note.md</code>          |
| <code>grounding-audit.md</code>      | <code>evidence for feedback-request.md</code> |
| dashboard preview from A7            | <code>dashboard-preview.md</code>             |

Groups may reuse work, but the files should be placed in the required folders with the required names.

For M3, place the report draft and dashboard-style KT product in the locations listed below so peers can find both during review:

- **Report draft:** `milestones/m3-project-update/submission/report-draft.qmd` and `report-draft.html`.

- **Dashboard/KT product:** the retained A7 source is `assignments/assignment07-dashboard/submission/dashboard.qmd`. Then `milestones/m3-project-update/submission/dashboard-preview.md` points reviewers to that source plus its screenshot or rendered-preview path. A static preview is enough; a live dashboard is not required.

## Feedback Request

A useful M3 feedback request is specific:

- Is the main claim supported by the table and figure?
- Is the audience clear?
- Is the limitation strong enough?
- Would the dashboard preview help the intended audience?
- What reproducibility issue should be fixed before Week 11?

## Before Submitting

Before rendering for submission, search your code for local paths such as `/Users/`, `C:\`, `Downloads`, or `Desktop`. Replace them with project-relative paths. The Search panel (the magnifying glass icon in the left sidebar) checks every file at once.

Before committing, check the staged files. If you see raw row-level data, credentials, `.RData`, OS junk, or generated files that are not allowed, unstage them and check `.gitignore` before syncing. The full [pre-commit privacy checklist](#) is on the Private Preview page.

Check:

- report renders;
- table and figure are code-generated;
- citations resolve;
- preview route is private unless public-safe;
- M3 feedback request names the decisions that still need review;

- all required files are committed and synced.

## **Week 11: Portfolio Surgery**

# Week 11: Portfolio Surgery

## Overview

This week is about making the project survive review. The goal is not to add features. The goal is to find and repair the failures that would stop another person from opening, rendering, understanding, or trusting the final portfolio.

Portfolio surgery means fixing, simplifying, documenting, and polishing the work you already have — not building something new.

The core pathway is a fresh Codespace reproducibility test, README and path repair, peer review, and final risk list. Advanced deployment pathways stay optional and demo-only.

### ! Required / Draft / Optional This Week

- **Required (graded):** the [M4 peer review](#). Portfolio Surgery itself is an in-class working session (not separately graded); its outputs feed M4 and the Final Portfolio, as explained in [port06](#).
- **Draft (class work that feeds M4 and the portfolio, not graded on its own):** the fresh-test failure log and repair notes from [port02](#).
- **Optional / demo-only (no penalty for skipping):** the deployment and Python pathways in [port05](#).

Tags like [required] and [demo-only] are defined in [How To Read Submission Lists](#).

## Objectives

By the end of the week you can:

- run a project in a fresh Codespace using only documented instructions;
- log failures before repairing them;
- improve README, path, dependency, citation, and output hygiene;
- use M4 peer review to find correctness, reproducibility, auditability, and KT risks;
- decide whether optional deployment pathways are worth pursuing after the core portfolio works.

## Connection

Week 10 produced a report and M3 project update. Week 11 tests whether that work can be reviewed in a fresh environment. The narrated presentation deck and final portfolio revisions then turn that evidence into the submitted portfolio.

## Dashboard Case Study: NHANES Equity

The **NHANES Health Equity Dashboard** is now the foreground case study. Use it as a small knowledge-translation product built on the cached NHANES equity dataset.

- App code: `examples/nhanes-equity/app/app.R`
- Cached data: `examples/nhanes-equity/data/nhanes_equity_v6.rds` and `examples/nhanes-equity/data/nhanes_equity_v6.csv`
- Optional rebuild script: `examples/nhanes-equity/scripts/build_nhanes_equity.R`

Run the dashboard from the repository root:

```
1 shiny::runApp("examples/nhanes-equity/app")
```



The app is offline-first: it loads the local RDS cache when available. Rebuilding from CDC uses `nhanesA`, requires internet access, and is optional unless assigned.

Companion materials:

- Week 10 report template: `weeks/week10-reporting-publishing/report-template.qmd`
- Week 11 static dashboard demo: `weeks/week11-portfolio-surgery/static-dashboard-ojs.qmd`

## Reading Order

1. [port01 - Surgery Goals](#)
2. [port02 - Fresh Codespace Test](#)
3. [port03 - README and Repo Hygiene](#)
4. [port04 - Peer Review and M4](#)
5. [port05 - Optional Pathways](#)
6. [port06 - Portfolio Surgery and the Final Portfolio](#)

The optional [static dashboard source file \(view on GitHub\)](#) remains a demo-only example. It is not a required final portfolio pathway.

## Class Plan

1. Open the project in a fresh Codespace or clean equivalent environment.
2. Follow the README exactly.
3. Render the report and open or run the dashboard-style KT product.
4. Log failures before fixing them.
5. Repair README, path, dependency, citation, or stale-output issues.
6. Exchange M4 peer review focused on reproducibility and audience fit.
7. Write the final risk list.

## Student Output

By the end of class each group has a fresh-environment test log, a failure log, a repair summary, peer review notes, and a final risk list for M4 and final portfolio planning.

## Links

- Session guide: [Portfolio Surgery and the Final Portfolio](#)
- Milestone: [M4 - Peer Review](#)
- Final package: [Final Portfolio](#)

## Definition of Done

- A fresh environment test was attempted.
- Failures were logged before repair.
- README and path issues were corrected or documented.
- M4 feedback is specific enough for the receiving group to act on.
- Remaining risks are named rather than hidden.

## What Students Leave With

A project that is closer to final submission because it has been tested, repaired, reviewed, and triaged.

# Surgery Goals

## Where This Fits

Portfolio surgery is the week when you stop asking, “What else can we add?” and start asking, “What would fail for a reviewer?”

Portfolio surgery means revising existing work so it is reproducible, readable, professional, privacy-safe, and understandable to someone outside the course. In practice, that is four kinds of work: fixing what is broken, simplifying what is confusing, documenting what is undocumented, and polishing what is rough.

The strongest final projects are not the ones with the most tools. They are the ones another person can open, rerun, inspect, and understand.

## The Core Standard

The surgery covers all of it — the analysis code, the report, the dashboard-style KT product, the README, and the repository itself. You are not revising just one piece; you are revising everything a reviewer will touch.

By the end of this week, the group should know:

- whether the report renders;
- whether the dashboard-style KT product opens or runs;
- whether paths are relative;
- whether data access is documented;
- whether citations resolve;
- whether AI-use notes are specific;
- whether README instructions match reality.

## **Do Not Add New Work Unless It Reduces Risk**

Good Week 11 changes:

- fix a broken path;
- add a missing package note;
- rewrite unclear README steps;
- regenerate stale outputs;
- document a known limitation;
- remove an unsupported claim.

Risky Week 11 changes:

- replacing the dashboard pathway;
- adding a new package without documenting it;
- adding a new analysis that is not needed for the report;
- switching to public publishing without privacy review.

## **Group Artifact Mindset**

M4 and the final portfolio are group project artifacts unless a brief explicitly says otherwise. Divide roles during testing, but keep the repository evidence in one shared place.

## **Check Yourself**

At the end of the week, a reviewer should be able to say:

I can find the report, open the KT product, trace the data source, understand the limitation, and see what still might fail.

# Fresh Codespace Test

## Goal

Test whether a project can run in a fresh environment and document the results and any failures before final submission.

This page is the main Week 11 runbook. Use it for the Portfolio Surgery session, M4 peer review, and final portfolio preparation.

## Roles

For group testing, assign roles:

- **Driver:** runs the project exactly as documented.
- **Navigator:** reads instructions aloud and watches for ambiguity.
- **Auditor:** logs paths, dependencies, citations, data provenance, and AI-use issues.

Rotate after the first major failure or after 20 minutes.

## Runbook

Complete these steps in a new Codespace or equivalent clean environment. Work from the repository root unless the project README says otherwise.

1. Open the repository from GitHub.
2. Record the latest commit hash.
3. Read the README before running code.
4. Follow the documented setup steps only.

5. Restore or install dependencies using documented instructions.
6. Render the final report or book.
7. Open or run the dashboard-style KT product.
8. Check that data paths are relative.
9. Check that citations resolve.
10. Check that figures and tables are generated from code.
11. Check that AI-use notes are present and specific.
12. Record the check results and any failure, warning, unclear instruction, or repair.
13. Commit any repairs with clear messages.
14. Submit the updated repository link and commit hash on Canvas.

## Failure Log

Use this table for the Portfolio Surgery session and for peer review.

| Check                                 | Result | Evidence | Fix made | Owner |
|---------------------------------------|--------|----------|----------|-------|
| Repository opens in fresh environment |        |          |          |       |
| Packages install or load              |        |          |          |       |
| Report renders                        |        |          |          |       |
| Dashboard or KT product opens         |        |          |          |       |
| Data paths are relative               |        |          |          |       |
| Citations resolve                     |        |          |          |       |
| README instructions are accurate      |        |          |          |       |
| AI-use note is complete               |        |          |          |       |

If every check passes, keep the evidence for each check and state that no failures were identified and no repairs were required. You will not lose marks for having no failures to repair. Do not create an artificial failure or make an unnecessary change.

## Example Failure Log

| Check                                 | Result       | Evidence                        | Fix made                                  | Owner     |
|---------------------------------------|--------------|---------------------------------|-------------------------------------------|-----------|
| Repository opens in fresh environment | Pass         | Codespace opened from GitHub    | None                                      | Group     |
| Packages install or load              | Needs repair | library(tidyverse) failed       | Added dependency note to README           | Analyst   |
| Report renders                        | Pass         | report.html created             | None                                      | Writer    |
| Dashboard opens                       | Needs repair | data path failed from repo root | Replaced absolute path with relative path | Developer |
| Citations resolve                     | Pass         | reference list appears          | None                                      | Writer    |

## NHANES Exemplar Checks

For the NHANES case study, the checks are:

- load `examples/nhanes-equity/data/nhanes_equity_v6.csv`;
- render the Week 10 report template;
- confirm the README explains cached data and optional CDC retrieval;
- if using the Shiny exemplar as a demo, launch it from the repository root;
- if using the static dashboard demo, open `weeks/week11-portfolio-surgery/static-dashboard-ojs.qmd`.

The CDC rebuild and static dashboard are optional/demo pathways, not required portfolio work.

## Optional Python Pathway

If the project uses Python, add a short dependency note with:

- how Python is launched;
- required packages;
- whether a notebook, script, or Quarto Python chunk is the primary workflow;
- how outputs are regenerated.

Keep the Python pathway documented, but do not add it unless it supports the core audience-facing product.

## Final Evidence and Repair Checklist

- ☐ Required outputs render or run from a fresh environment.
- ☐ README instructions match the tested workflow.
- ☐ Data paths are relative.
- ☐ Data access instructions are present.
- ☐ Citations resolve.
- ☐ AI-use notes identify what was drafted and how it was checked.
- ☐ Remaining risks are documented in the failure log.



# README and Repo Hygiene

## Why This Matters

A reviewer should not need private instructions to understand the project. The README and file structure are part of the scientific record.

Assume the reader has not taken this course. Your README and portfolio page should explain the project purpose, data source, main result, limitations, and how to reproduce the work.

Good repo hygiene reduces grading friction and makes the final portfolio easier to trust.

## README Minimums

The project README should name:

- project title and audience;
- project question;
- data source and access route;
- folder structure;
- how to render the report;
- how to open or run the dashboard-style KT product;
- package or environment setup;
- known limitations or risks;
- public/private sharing status.

## Weak vs Strong: A README Example

A weak README tells the reader almost nothing:

```
1 # Our Project
2 Class project for SPPH 381H. Run the code to see the results.
```

A reviewer who opens this repository does not know what the project asks, where the data comes from, or which file to run.

A strong README answers those questions directly:

```
1 # Sleep and Blood Pressure in US Adults (NHANES 2017-2018)
2
3 Audience: community health staff planning sleep-screening outreach.
4 Question: Is short sleep associated with higher blood pressure?
5 Data: NHANES 2017-2018 public files; see data/README.md for the access route.
6 Main result: adults reporting under 6 hours of sleep had higher average
7   systolic blood pressure; see reports/final-report.html.
8 Reproduce: open reports/final-report.qmd and render it
9   (Command Palette > "Quarto: Render Document").
10 KT product: open dashboard/index.qmd and render it the same way.
11 Limitations: cross-sectional data; no causal claims.
12 Sharing: private repository; course staff have access.
```

The difference is not length. The strong README answers the reviewer's questions before they are asked: what the project is, who it is for, where the data comes from, what was found, and how to rerun it.

## Path Hygiene

Use relative paths. Avoid:

```
1 C:\Users\...
2 E:\GitHub\...
3 /Users/name/...
```

Prefer paths that work from the repository root, such as:

```
1 data/clean/analysis.csv
2 reports/final-report.qmd
3 dashboard/index.qmd
```

## Dependency Notes

If the project uses R packages beyond the course defaults, document them. If the project uses Python as an optional extension, document how it is launched and which packages are needed.

Keep the dependency note short. The goal is enough information for a reviewer to rerun the work.

## Output Hygiene

Generated outputs should be traceable to source code. For each important output, ask:

- Which source file generates it?
- Which data file does it use?
- Can it be regenerated in a fresh Codespace?
- Is it safe to share publicly?

## Git Hygiene

Before final submission:

- commit repairs with meaningful messages;
- avoid committing credentials, tokens, or local cache files;
- keep `.gitignore` aligned with the project;
- sync the final commit before submitting the repository link.

## Quick Repair Pattern

Use this structure in `repair-summary.md`:

| File changed                | Problem fixed          | How verified                         |
|-----------------------------|------------------------|--------------------------------------|
| <code>README.md</code>      | Missing render command | Followed command in fresh Codespace  |
| <code>report.qmd</code>     | Absolute data path     | Rendered with relative path          |
| <code>references.bib</code> | Missing citation       | Citation appeared in rendered report |

# Peer Review and M4

## Where This Fits

M4 is the structured peer review checkpoint. Treat it as a group project artifact unless your course instructions explicitly say otherwise.

The goal is actionable feedback: comments specific enough that the receiving group can improve the final portfolio without guessing.

## What to Review

Use four lenses:

| Lens            | Question                                          |
|-----------------|---------------------------------------------------|
| Correctness     | Do the claims match the code output and evidence? |
| Reproducibility | Can the report and KT product run or render?      |
| Auditability    | Are data source, AI use, and limitations visible? |
| KT value        | Does the product fit the intended audience?       |

## Review Sequence

1. Open the repository.
2. Read the README first.
3. Try the documented render or run path.
4. Inspect the report, dashboard-style product, and data/source notes.
5. Complete the M4 files listed in [M4: Peer Review](#).
6. Name one blocking risk or state why no blocking risk was found.

## Useful Feedback

Helpful:

The report renders, but the dashboard path in the README points to an old folder. Update the README and rerun the dashboard test before final submission.

Less helpful:

Good project.

## M4 Files

M4 uses `milestones/m4-peer-review/submission/`.

The required files are:

- `peer-review.md`
- `reproducibility-check.md`
- `auditability-check.md`
- `kt-feedback.md`
- `ai-use-note.md`

Do not invent extra files unless your group needs them for organization.

At the end of `peer-review.md`, add the three short reflection prompts from the M4 brief: what the review changed in your own thinking, one limitation/bias/ stewardship issue that deserves attention, and one review habit you will apply to your own project. No separate reflection file is submitted.

## AI Use in Peer Review

AI may help organize feedback or identify unclear prose. It must not invent evidence. Reviewers still need to inspect the repository, rendered output, citations, and limitations themselves.

## **Before Sending Feedback**

Check that your comments are:

- specific;
- respectful;
- tied to evidence;
- prioritized;
- useful before final submission.

# Optional Pathways

 Demo only — not graded

This pathway is a **demonstration only**. It is not graded and not required. Using it for graded work requires **instructor approval first**. The supported, graded pathway in this course is the Quarto workflow.

The advanced deployment and Python-flavoured pathways are optional demonstrations. Complete the required portfolio polish first, then explore optional pathways only if you have time and instructor approval.

Everything on this page is [optional — include only if relevant to your project]. Skipping it carries no penalty (tag vocabulary: [How To Read Submission Lists](#)).

## Keep the Core First

Optional pathways are only useful after the core portfolio works:

- report renders;
- dashboard-style KT product opens or runs;
- README explains the workflow;
- data source and limitations are clear;
- fresh environment test has been attempted.

Do not switch tools in Week 11 unless the change reduces risk or the instructor has approved the pathway.



## Optional Shiny

Shiny can support richer interactivity, but it introduces extra testing and deployment questions. Use it only when the audience need justifies a running app and the group can document how to launch it.

For this course, the NHANES Shiny app is an exemplar and demonstration. It is not the default required pathway for every project.

## Optional Static Dashboard

The source file [static-dashboard-ojs.qmd](#) ([view on GitHub](#)) demonstrates a no-server dashboard that reads the NHANES CSV snapshot in the browser. It is a *source file to read or run yourself*, not a page of this book; the instructor demonstrates the rendered version in class. It is useful as an optional portfolio extension after the required report and KT product are reproducible.

This file remains demo-only in this pass. Do not treat it as a portfolio requirement.

## Optional Python

Python may be useful for a small extension if it supports the project and runs in Codespaces. Keep it awareness-level unless the group has a clear reason and a documented environment.

If Python is included, document:

- where the Python code lives;
- how to run it;
- which packages are required;
- what outputs it creates;
- how it connects to the R/Quarto core.

## Public Deployment

Public deployment is optional and should follow the Week 10 rule: private preview unless public-safe.

Before public sharing, check:

- no restricted data;
- no credentials;
- no row-level sensitive examples;
- aggregate outputs only;
- README explains data use and limitations.

## Decision Rule

Use the simplest pathway that serves the audience and can be reproduced by course staff. Optional tools are welcome only when they make the project clearer, not when they make the project harder to review.

# Portfolio Surgery and the Final Portfolio

## Where This Fits

This page connects the Week 11 [Portfolio Surgery](#) session to [M4 peer review](#) and the [Final Portfolio](#). Portfolio Surgery is an in-class working session, not a separately graded assignment: the artifacts you produce here feed your M4 peer review and your Final Portfolio. It does not add new deliverables.

## What To Produce in the Session

Work through these in your project repository. They are working artifacts for M4 and the portfolio, not a separate graded submission.

### Fresh-environment reproducibility test

Document the fresh-environment test using [port02](#). Record the latest commit hash, environment, commands, and result.

If every reproducibility check passes, that is a good outcome: document the checks you ran and their results, and note that no failures were found. You are not penalised for having nothing to repair — do not invent a failure or make an unnecessary change just to have something to log.

## Failure log

List any failures before repairs: warnings, broken paths, missing packages, unclear README steps, citation issues, and stale outputs. Give each failure evidence and a result after repair or triage.

## Repairs

Name any files you changed and why, or state that no repairs were required. This can be short.

| File changed | Reason               | Verification                        |
|--------------|----------------------|-------------------------------------|
| README.md    | Added render command | Followed command in fresh Codespace |

## Peer review (feeds M4)

Record the feedback you received or gave, keeping it specific and actionable. Your [M4 peer review](#) is the graded milestone; see [port04](#) for how the session feeds it.

## Final risk list

Name remaining risks or state that no known risks remain. Honest risk lists are better than hidden problems. This carries directly into the Final Portfolio.

## AI-use note

State what AI helped with and how any repairs or wording were checked.

## Final Portfolio Bridge

The revision target is all of it — the analysis project, the report, the dashboard-style KT product, and the repository with its README — not just one piece.

The final portfolio is not a separate new analysis. It gathers the strongest version of:

- final report source and rendered output;
- dashboard-style KT product or static preview;
- data provenance and stewardship documentation;
- reproducibility instructions;
- narrated presentation slide deck;
- final AI-use audit;
- clear README and repository access route.

Your dashboard-style KT product may be submitted as a static preview; a live dashboard is not required.

Use the [Final Portfolio Checklist](#) before submission.

Two rules make this easier:

- **You may revise earlier work — and you are encouraged to.** If feedback on an earlier assignment showed you a better way, fix it before it goes into the portfolio. The portfolio is graded on the strongest version of your work, not the first version you submitted.
- **Required files are prescribed; the rest of the structure is yours.** The checklist names the files every portfolio must contain. Beyond those files, your group is free to organize and present the portfolio however serves your project best.

Not sure what “strong” looks like? A weak portfolio is a pile of files the reader must decode. A strong portfolio leads with a README that answers what the project is, who it is for, what was found, and how to rerun it. See the side-by-side [weak vs strong README example](#).

## Public or Private Repository

Use [Repository Sharing Guidance](#). Private repositories are appropriate when public sharing would create data, privacy, licensing, or stewardship risk.

Public is not better by default. Reviewable, safe, and reproducible is the target.

## Before You Finish the Session

Check:

- fresh environment test attempted;
- any failures logged before repair, or passing checks documented;
- report and KT product rerun or failure clearly explained;
- README matches the tested workflow;
- M4 feedback is recorded;
- final risk list is honest.

## Before Final Portfolio

After Portfolio Surgery, focus on:

- repairing any remaining blockers;
- aligning presentation slides with final report claims;
- checking repository access;
- confirming public/private sharing choice;
- syncing the final commit before Canvas submission.

# **Project Presentation (Online)**

# Project Presentation (Narrated Deck)

## Overview

The term-project presentation is a **narrated 5-7 slide deck submitted online with the [Final Portfolio](#)** on Dec 11. There is no in-class presentation session. The deck is the knowledge-translation version of your project: it should emphasize scientific clarity, knowledge translation, reproducibility, and auditability.

Full requirements and the 10-point rubric are in the [Presentation brief](#); this page is the how-to for recording and submitting it.

## What To Record

Record narration over your slides so a viewer can follow the story without you in the room. Any of these is acceptable:

- a narrated `slides.html` or `slides.pdf` export;
- a short screen recording that walks through the slides with voice-over.

Keep it to the length of a 5-7 slide talk. Aim for a clear spoken explanation of the audience, question, data, main finding, and limitation — not a read-aloud of every word on the slides.



## Slide Submission

Before the portfolio deadline, commit and sync:

```
1 milestones/m5-presentation/submission/slides.qmd
2 milestones/m5-presentation/submission/slides.html
```

If you render to PDF instead of HTML, submit `slides.pdf`. Commit the narrated recording alongside them. Submit the GitHub repository link on Canvas with the latest commit hash and the slide path.

### What is a commit hash?

A commit hash is the unique identifier GitHub assigns every saved version of your work. It looks like `a3f92bc`. To find it, go to your repository on GitHub, click **Commits**, and copy the short code on the right. You can also run `git log --oneline -1` in the terminal.

## Slide Checklist

Before submitting, check that:

- slides render from source in a fresh Codespace;
- fonts and figures are readable on screen;
- every chart has a title, units, and limitation cue;
- source and data notes are visible;
- the final slide gives the audience a clear takeaway;
- the narration is audible and matches what is on screen;
- backup notes include data source, cohort, and AI-use details.

## Portfolio Reminder

The presentation is scored as the **Presentation** component (10 points) of the Final Portfolio. Use the M4 peer feedback you

received to revise the final report, dashboard-style KT product, and deck together, then confirm the [Final Portfolio Checklist](#) before submission.

## **Definition Of Done**

The presentation is done when the narrated slides are committed, rendered, and submitted with the Final Portfolio, and play without relying on local-only files.

## **Assignments and Milestones**

# Assignment Schedule

One page for everything you submit this term: what is due, when, and whether it is your own work or your project group's. Click a deliverable to open its full brief.

! Canvas is the source of truth

The dates below follow the Sept-Dec 2026 syllabus rule: **weekly assignments are due the Monday after class at 4:00 PM**, and milestone dates come from the syllabus milestone table. If Canvas shows a different date or time, Canvas wins.

## How to read this page

- **Individual** items are the weekly assignments. You submit them alone.
- **Group** items are the term-project milestones. One submission per project group.
- **Taught in** links to the week whose class prepares you for that deliverable.
- Grading rules (Complete/Incomplete, best 4 of 6, project points) are explained on the [Assignments](#) and [Milestones](#) pages.

## Term at a glance

| Due                           | Type         | Deliverable                                        | Taught in                            | Grading                                                        |
|-------------------------------|--------------|----------------------------------------------------|--------------------------------------|----------------------------------------------------------------|
| Before<br>Tue Sept<br>15      | Individual   | <a href="#">Week 0<br/>onboard-<br/>ing tasks</a>  | <a href="#">Week 0</a>               | Not<br>graded;<br>must be<br>done<br>before the<br>first class |
| Week 1<br>module on<br>Canvas | Individual   | <a href="#">Data<br/>Intake<br/>Card</a>           | <a href="#">Week 1<br/>(Sept 15)</a> | Required<br>check-<br>point,<br>un-<br>weighted                |
| Mon Sept<br>28, 4 PM          | Individual   | <a href="#">A1:<br/>Modern<br/>Workflows</a>       | <a href="#">Week 2<br/>(Sept 22)</a> | Complete/Incomplete                                            |
| Mon Oct<br>5, 4 PM            | Individual   | <a href="#">A2: R<br/>With AI</a>                  | <a href="#">Week 3<br/>(Sept 29)</a> | Complete/Incomplete                                            |
| Mon Oct<br>12, 4 PM           | Individual   | <a href="#">A3: Git<br/>Collabora-<br/>tion</a>    | <a href="#">Week 4<br/>(Oct 6)</a>   | Complete/Incomplete                                            |
| †<br>Mon Oct<br>12, 4 PM      | <b>Group</b> | <a href="#">M0:<br/>Group<br/>Formation</a>        | <a href="#">Week 4<br/>(Oct 6)</a>   | Complete/Incomplete                                            |
| †<br>Mon Oct<br>19, 4 PM      | Individual   | <a href="#">A4:<br/>Polyglot<br/>Parity</a>        | <a href="#">Week 5<br/>(Oct 13)</a>  | Best 4 of<br>6                                                 |
| Mon Oct<br>26, 4 PM           | Individual   | <a href="#">A5: Visu-<br/>alization<br/>Audit</a>  | <a href="#">Week 6<br/>(Oct 20)</a>  | Best 4 of<br>6                                                 |
| Mon Oct<br>26, 4 PM           | <b>Group</b> | <a href="#">M1:<br/>Project<br/>Proposal</a>       | <a href="#">Week 6<br/>(Oct 20)</a>  | 5 points                                                       |
| Mon Nov<br>2, 4 PM            | Individual   | <a href="#">A6: EDA<br/>and AI<br/>Audit</a>       | <a href="#">Week 7<br/>(Oct 27)</a>  | Best 4 of<br>6                                                 |
| Mon Nov<br>2, 4 PM            | <b>Group</b> | <a href="#">M2: Pre-<br/>liminary<br/>Analysis</a> | <a href="#">Week 7<br/>(Oct 27)</a>  | 5 points                                                       |

| Due               | Type         | Deliverable                                                                         | Taught in                        | Grading     |
|-------------------|--------------|-------------------------------------------------------------------------------------|----------------------------------|-------------|
| Mon Nov 9, 4 PM † | Individual   | <a href="#">A7: Dashboard KT Prototype</a>                                          | <a href="#">Week 8 (Nov 3)</a>   | Best 4 of 6 |
| Mon Nov 23, 4 PM  | Individual   | <a href="#">A8: Scientific Communication</a>                                        | <a href="#">Week 9 (Nov 17)</a>  | Best 4 of 6 |
| Mon Nov 23, 4 PM  | <b>Group</b> | <a href="#">M3: Project Update (draft)</a>                                          | <a href="#">Week 10 (Nov 24)</a> | 5 points    |
| Mon Nov 30, 4 PM  | Individual   | <a href="#">A9: Reproducible Report</a>                                             | <a href="#">Week 10 (Nov 24)</a> | Best 4 of 6 |
| Mon Nov 30, 4 PM  | <b>Group</b> | <a href="#">M4: Peer Review</a>                                                     | <a href="#">Week 11 (Dec 1)</a>  | 5 points    |
| Tue Dec 1         | —            | No new submission.<br>Week 11 class is portfolio surgery and acting on M4 feedback. | <a href="#">Week 11 (Dec 1)</a>  | —           |
| Fri Dec 11, 4 PM  | <b>Group</b> | <a href="#">Presentation: narrated 5-7 slide deck</a>                               | <a href="#">Week 11 (Dec 1)</a>  | 10 points   |
| Fri Dec 11, 4 PM  | <b>Group</b> | <a href="#">Final Portfolio</a>                                                     | <a href="#">Week 11 (Dec 1)</a>  | 35 points   |

† These Mondays are UBC closure days (Oct 12 is Thanksgiving; Nov 9 falls in the Nov 9-11 mid-term break). The “Monday after class” rule lands on them, so check Canvas for the confirmed date and time.

There is no class on Tue Nov 10 (mid-term break). Week 9

meets Tue Nov 17.

## Individual assignments only

Nine weekly assignments, all due Monday at 4 PM. Assignments 1-3 are pass/fail gates; only the best four scores from Assignments 4-9 count (8.75% each, 35% total).

| Due          | Assignment                                   | Brief                    |
|--------------|----------------------------------------------|--------------------------|
| Mon Sept 28  | Assignment 1:<br>Modern Workflows            | <a href="#">A1 brief</a> |
| Mon Oct 5    | Assignment 2: R<br>With AI                   | <a href="#">A2 brief</a> |
| Mon Oct 12 † | Assignment 3: Git<br>Collaboration           | <a href="#">A3 brief</a> |
| Mon Oct 19   | Assignment 4:<br>Polyglot Parity             | <a href="#">A4 brief</a> |
| Mon Oct 26   | Assignment 5:<br>Visualization<br>Audit      | <a href="#">A5 brief</a> |
| Mon Nov 2    | Assignment 6:<br>EDA and AI Audit            | <a href="#">A6 brief</a> |
| Mon Nov 9 †  | Assignment 7:<br>Dashboard KT<br>Prototype   | <a href="#">A7 brief</a> |
| Mon Nov 23   | Assignment 8:<br>Scientific<br>Communication | <a href="#">A8 brief</a> |
| Mon Nov 30   | Assignment 9:<br>Reproducible<br>Report      | <a href="#">A9 brief</a> |

## Group milestones only

Seven project checkpoints, one submission per group. M1-M4 are 5 points each; the Presentation and Final Portfolio are submitted together on Dec 11.

| Due          | Milestone                          | Brief                                 |
|--------------|------------------------------------|---------------------------------------|
| Mon Oct 12 † | M0: Group Formation                | <a href="#">M0 brief</a>              |
| Mon Oct 26   | M1: Project Proposal               | <a href="#">M1 brief</a>              |
| Mon Nov 2    | M2: Preliminary Analysis           | <a href="#">M2 brief</a>              |
| Mon Nov 23   | M3: Project Update (draft)         | <a href="#">M3 brief</a>              |
| Mon Nov 30   | M4: Peer Review                    | <a href="#">M4 brief</a>              |
| Fri Dec 11   | Presentation (narrated slide deck) | <a href="#">Presentation brief</a>    |
| Fri Dec 11   | Final Portfolio                    | <a href="#">Final Portfolio brief</a> |

New to milestones? The [NHANES worked example](#) walks one project from M0 to the Final Portfolio.

## For the instructor: setting up Canvas

Every row in the [term at a glance](#) table maps to one Canvas assignment with the listed due date at 4:00 PM. Weekly assignments are individual submissions; milestones are group submissions (one per project group), with the Presentation and Final Portfolio as two Canvas items sharing the Dec 11 deadline. Weeks 0 and 1 hold the two ungraded checkpoints. Resolve the two † dates before publishing the Canvas calendar.



# Assignments

💡 Looking for due dates?

The [Assignment Schedule](#) lists every assignment and milestone in date order, marks which are individual and which are group work, and links to each brief.

Weekly assignments are scaffolded practice. Assignments 1-3 are foundational gates graded Complete/Incomplete. Assignments 4-9 support the term portfolio and use the best-4-of-6 grading policy described in the syllabus.

## Assignment Map

| Assignment   | Week | Focus                                              | Grading mode        | Brief                              |
|--------------|------|----------------------------------------------------|---------------------|------------------------------------|
| Assignment 1 | 2    | Modern work-flows: Codespaces, Quarto, commit/sync | Complete/Incomplete | <a href="#">Assignment 1 Brief</a> |
| Assignment 2 | 3    | R with AI: import, inspect, transform, audit       | Complete/Incomplete | <a href="#">Assignment 2 Brief</a> |

| Assignment   | Week | Focus                                                    | Grading mode        | Brief                    |
|--------------|------|----------------------------------------------------------|---------------------|--------------------------|
| Assignment 3 | 4    | Git collaboration: branches, pull requests, repo hygiene | Complete/Incomplete | <a href="#">A3 brief</a> |
| Assignment 4 | 5    | Polyglot awareness and R deepening                       | Best 4 of 6         | <a href="#">A4 brief</a> |
| Assignment 5 | 6    | Visualization and AI visual audit                        | Best 4 of 6         | <a href="#">A5 brief</a> |
| Assignment 6 | 7    | EDA, Table 1, planted-error audit                        | Best 4 of 6         | <a href="#">A6 brief</a> |
| Assignment 7 | 8    | Dashboard-style KT prototype                             | Best 4 of 6         | <a href="#">A7 brief</a> |
| Assignment 8 | 9    | Scientific communication and peer feedback               | Best 4 of 6         | <a href="#">A8 brief</a> |
| Assignment 9 | 10   | Reproducible report, citations, publishing check         | Best 4 of 6         | <a href="#">A9 brief</a> |

## Grading Logic

Assignments 1-3 must be completed to pass the course. They are not numerically weighted, but they establish the workflow students need for every later assignment.

Assignments 4-9 are scored on Canvas. Only the best 4 of these 6 scores count toward the final grade, so students have flexibility while still practicing consistently after the foundation weeks.

## Common Reproducibility Contract

Every assignment must:

- run or render in GitHub Codespaces;
- use relative paths rather than machine-specific paths;
- keep source files and every explicitly required rendered output in the requested submission folder and commit both;
- commit and sync the final work before submitting the repository link on Canvas;
- document packages or dependencies when adding anything beyond the course defaults.

Do not add global `*.html` or `*.pdf` rules to the course workspace `.gitignore`. Use the assessment's submission list as the authority: a listed render is committed; an unlisted practice render does not become a deliverable.

## How To Read Submission Lists

Assignment and milestone pages use these bracketed tags. Anything without a tag is required.

- **[required]** — must be submitted; graded.
- **[optional]** — **include only if relevant to your project** — never required; no penalty for skipping.

- **[if applicable — required only if your workflow creates this file]** — submit it when your chosen pathway produces it.
- **[brief — 1-3 focused bullets or sentences are enough]** — short is correct; do not pad.
- **[demo-only]** — instructor demonstration material; not graded, and using the pathway for graded work needs instructor approval first.

## Common AI-Use Expectation

AI may help students brainstorm, debug, translate, refactor, or improve wording. Students remain responsible for checking every code path, column name, statistic, citation, and interpretation. When AI is used, the submission must include an AI-use note explaining what the tool helped with, what changed, and how the result was verified.

## The Official AI Documentation Rule

Two pieces work together, and only one is graded:

1. **ai-use-note.md is the graded artifact.** Every assignment (A2 onward) and milestone submission includes one — even if you did not use AI (then it is a single sentence saying so).
2. **Inline # AI prompt: / # Verified: comments are required practice** in AI-assisted code (taught in Weeks 3, 6, and 7). They are the working audit trail you keep while coding; graders read the note, not every comment — your note summarizes them.

## ai-use-note.md Template

Copy this into your submission folder and fill it in:

```

1 # AI-Use Note
2
3 ## Audit trail
4 <!-- Copy your "# AI prompt:" / "# Verified:" comment pairs here. -->
5
6 ## What AI helped with
7 <!-- 1-3 sentences: which task, which tool. -->
8
9 ## What I changed
10 <!-- 1-3 sentences: what you edited, fixed, or rejected. -->
11
12 ## How I verified the result
13 <!-- 1-3 sentences: what you checked it against (data, docs, course notes). -->

```

If you did not use AI, the whole note is: AI was not used for this submission.

## Filled Example

```

1 # AI-Use Note
2
3 ## Audit trail
4 # AI prompt: ggplot2 bar chart of mean BMI by income group, with labels
5 # Verified: column names against names(nhanes); compared means with summary()
6
7 ## What AI helped with
8 Copilot drafted the initial ggplot2 code for the income-group plot and
9 suggested axis label wording.
10
11 ## What I changed
12 The draft used a column called "income" that does not exist; I replaced it
13 with IncomeGroup. I also removed a misleading y-axis limit it added.
14
15 ## How I verified the result
16 I re-ran the summary statistics without the plot and checked that the bar
17 heights match the group means; I confirmed the caption says the result is
18 descriptive and unweighted.

```

# Assignment 1: Modern Workflows

## Purpose

This required Complete/Incomplete checkpoint confirms that your personal course workspace works before the analysis weeks begin. You will edit the supplied root-level Quarto file, render it, commit both source and output, sync the commit, and submit the repository link.

## Due Timing

Submit by **Monday after the Week 2 class at 4:00 PM.**

## Learning Objectives

- Launch or reopen a Codespace from your personal workspace fork.
- Edit YAML, Markdown, and a simple R code chunk.
- Render a self-contained HTML file and locate it in the Explorer.
- Commit and sync a reproducible change.
- Verify the commit on GitHub and stop the Codespace.

## Inputs

- Personal workspace linked from Canvas
- Root starter: `practice_report.qmd`

- Synthetic practice data: `data/patients.csv`
- Week 2 workflow page

## Tasks

1. Open your personal workspace fork and launch or reopen its Codespace.
2. At the repository root, open the supplied `practice_report.qmd`.
3. Change the YAML `author` field to your name.
4. Add a `## My Notes` section with 2–3 sentences about what you learned.
5. Use at least one Markdown feature such as bold text, italics, or a list.
6. Modify an existing code chunk or add one small R calculation.
7. Render the document and confirm `practice_report.html` appears beside the source file and contains your edits and chart.
8. In Source Control, inspect the changes and stage both `practice_report.qmd` and `practice_report.html`.
9. Commit with a clear message, sync, and verify the commit on GitHub.
10. Stop the Codespace when finished.

## AI-Use Expectations

AI may help explain an error or Quarto syntax. If you use AI, add one sentence under `## My Notes` naming what it helped with and how you verified the result. No separate AI-use file is required for A1.

## Reproducibility Requirements

- Work in the supplied personal workspace rather than the read-only course book.
- Keep `practice_report.qmd` and `practice_report.html` at the repository root.

- Use only relative paths.
- Commit the required rendered HTML with its source.

## What To Submit

Submit your personal workspace repository link on Canvas. The linked, synced commit must contain these two root-level files:

- `practice_report.qmd`
- `practice_report.html`

No reflection file, commit-evidence file, screenshot, or nested A1 submission folder is required. The repository link and Git history provide the commit evidence.

## Grading Checklist

Complete:

- the root starter contains the required YAML, Markdown, and code edits;
- `practice_report.html` renders from that source and is committed;
- paths are relative;
- the final commit is synced and visible through the submitted link.

Needs work:

- either required root file is missing;
- the rendered output does not match the source;
- the code does not render or uses a machine-specific path;
- the final change is not synced.



## Definition of Done

A1 is done when the Canvas link opens your synced personal workspace and the root-level source and rendered HTML show the completed workflow check.

# Assignment 2: R With AI

## Purpose

This assignment turns the Week 3 R basics into a small auditable analysis note. You will load data with a relative path, inspect it before summarizing, use `dplyr` verbs, write one custom function, and document how any AI-generated code was checked.

## Learning Objectives

- Load R packages and import a CSV with a relative path.
- Inspect rows, columns, variable types, and missingness before analysis.
- Use `filter()`, `mutate()`, `group_by()`, and `summarise()` for a descriptive summary.
- Write one small custom function and test it on the dataset.
- Audit AI-generated code for wrong column names, missingness mistakes, and overclaiming.

## Inputs

- Week 3 page: `weeks/week03-r-with-ai/index.qmd`
- Starter note: `assignments/assignment02-r-with-ai/starter-analysis.qmd`
- Dataset: `examples/nhanes-equity/data/nhanes_equity_v6.csv`
- Submission folder: `assignments/assignment02-r-with-ai/submission/`

## Tasks

1. Create the folder `assignments/assignment02-r-with-ai/submission/`.
2. Copy the starter structure into `analysis-note.qmd`.
3. Load `readr`, `dplyr`, and any other package used in the file.
4. Import the NHANES CSV using a relative path from the repository root.
5. Inspect the dataset with row count, column names, variable types, and missingness.
6. Create one cleaned or analysis-ready table using at least three `dplyr` verbs.
7. Write one custom function, such as a function that returns mean and SD for a numeric variable.
8. Save a small output file named `output/summary_table.csv` — this is the required filename, and the starter note already writes it.
9. Add AI audit comments near code that was drafted, debugged, or revised with AI assistance.
10. Render the Quarto note to HTML.
11. Commit and sync the source, rendered output, saved output file, and AI-use note.

## AI-Use Expectations

AI may help draft code, explain an error, or suggest a function structure. You must verify that all column names exist, missing values are handled intentionally, outputs match the code, and interpretations remain descriptive. Include `ai-use-note.md` even if AI was not used.

## Reproducibility Requirements

- The analysis must render in GitHub Codespaces.
- The data path must be relative.
- Output files must be created by code, not manually edited.

- Load every package used in the Quarto source and list it in `dependency-note.md`.
- The course devcontainer is the dependency baseline. Do not initialize `renv`. If you think an additional package is necessary, ask before adding it and document the approved change so the work still runs in a fresh Codespace.

## What To Submit

Place all submission files in:

```
1 assignments/assignment02-r-with-ai/submission/
```

Submit these files:

- `analysis-note.qmd` [required]: Quarto source with data import, inspection, cleaning, function, and AI audit comments.
- `analysis-note.html` [required]: rendered output.
- `output/summary_table.csv` [required]: code-generated output. Use the required filename listed in this submission checklist — do not rename it to `clean_data.csv`.
- `ai-use-note.md` [required]: what AI helped with and how you checked it, or a sentence saying AI was not used. Template and filled example: [The Official AI Documentation Rule](#).
- `dependency-note.md` [required]: packages used and whether any were added beyond the course environment.

## Submission Route

Submit your GitHub repository link on Canvas after committing and syncing the required files.

## Grading Checklist

Complete:

- CSV is imported with a relative path.
- Dataset inspection includes size, column names/types, and missingness.
- `dplyr` workflow uses at least three verbs, as stated in the task list.
- One custom function is defined and used.
- Output CSV is generated by code.
- AI-use note and dependency note are present.
- Quarto file renders in Codespaces.

Incomplete:

- Data path is absolute or machine-specific.
- Code does not render.
- Missingness is ignored or silently dropped.
- Function is absent or unused.
- Output file is missing or manually created.
- AI-use or dependency note is missing.

## Definition of Done

You are done when `analysis-note.qmd` renders, the output CSV is regenerated by code, and the Canvas submission links to the synced repository folder.

# Assignment 3: Git Collaboration

## Purpose

This assignment checks that you can make a small, reviewable change using Git. You will create a branch, make a documentation-only edit, inspect the diff, open a pull request, and connect the workflow to M0 group repository readiness.

## Learning Objectives

- Make meaningful commits with clear messages.
- Read a diff before committing.
- Use `.gitignore` to keep generated or sensitive files out of version control.
- Create a branch and open a pull request.
- Document a limited change so collaborators can review it.

## Inputs

- Week 4 page: `weeks/week04-git-collaboration/index.qmd`
- M0 connection: `milestones/m0-group-formation/README.md`
- Submission folder: `assignments/assignment03-git/submission/`

## Tasks

1. Create the folder `assignments/assignment03-git/submission/`.
2. Create a new branch with a short name such as `docs/git-audit-note`.
3. Make one documentation-only edit in your repository. Good choices include clarifying a README sentence, adding a path note, or improving a caption.
4. Inspect the diff and confirm the change is limited to documentation or text.
5. Check `.gitignore` and add a safe pattern if a generated or temporary file is being tracked.
6. Commit with a message that explains the change.
7. Push the branch and open a pull request.
8. If a pull request cannot be opened, do not stop: take a screenshot of your branch and diff, and write a short explanation instead. Both go in `pull-request-evidence.md`.
9. In the pull request description, explain what changed and how you verified the scope.
10. Add a short note on whether your group repository is ready for M0: members, access, communication plan, and Codespaces access.

## AI-Use Expectations

AI may help summarize a diff or draft a pull request description. You must verify the diff yourself and make sure the pull request does not include unrelated file changes. Include an AI-use sentence in `git-audit-note.md`.

Save the evidence as you work: copy the AI prompt or output you used into `git-audit-note.md` (for example: “I used AI to summarize the diff; I re-read the diff myself and confirmed the summary is accurate.”), and paste the pull request link plus one explanatory line into `pull-request-evidence.md` (for example: “PR link: ... — a documentation-only edit to one README sentence.”). Fuller worked examples of both files are on the Week 4 [AI Auditing page](#).

## Reproducibility Requirements

- All changes must be visible in Git history.
- The pull request must show a focused diff.
- `.gitignore` must not hide source files needed to reproduce the work.
- Evidence files must use relative paths or GitHub URLs.

## What To Submit

Place all submission files in:

```
1 assignments/assignment03-git/submission/
```

Submit these files:

- `pull-request-evidence.md` [required]: pull request URL, branch name, and final commit hash. If a pull request cannot be opened, include screenshot filenames and a written explanation.
- `documentation-change.md` [required]: the before/after text or a link to the changed documentation file.
- `git-audit-note.md` [required]: diff summary, `.gitignore` check, AI-use sentence, and M0 readiness note.

## Submission Route

Submit your GitHub repository link on Canvas after committing and syncing the required files.

## Grading Checklist

Complete:

- Branch was created for the work.
- Documentation-only change is focused and reviewable.



- Diff was inspected and summarized.
- Commit message is clear.
- Pull request evidence or equivalent text evidence is included.
- `.gitignore` was checked.
- M0 readiness note names repository access and team communication status.

Incomplete:

- Work was committed directly without evidence of a branch or review step.
- Diff includes unrelated changes.
- Pull request evidence is missing.
- `.gitignore` check is absent.
- M0 readiness note is missing.
- Work is not synced to GitHub.

## Definition of Done

You are done when the Canvas submission links to a synced repository with pull request evidence, a documentation change summary, and a Git audit note in the Assignment 3 submission folder.

# Assignment 4: Polyglot Parity

## Purpose

This assignment strengthens R first, then translates a small descriptive task to Python/pandas and checks whether both languages produce the same answer. The goal is careful reading of code in both R and Python, not Python mastery.

## Learning Objectives

- Build a trusted R `dplyr` summary before translating it.
- Refactor repeated R logic into a small function.
- Translate a simple summary to Python/pandas with AI support if useful.
- Run parity checks for row counts, group labels, and rounded statistics.
- Keep notebook outputs and dependency notes clean for review.

## Inputs

- Week 5 page: `weeks/week05-polyglot-r-deepening/index.qmd`
- R deepening page: `weeks/week05-polyglot-r-deepening/r-deepening-parity.qmd`
- Dataset: `examples/nhanes-equity/data/nhanes_equity_v6.csv`
- Parity template: `assignments/assignment04-polyglot/parity-table-template.csv`
- Submission folder: `assignments/assignment04-polyglot/submission/`

## Tasks

1. Create or use the folder `assignments/assignment04-polyglot/submission/`.
2. Create `polyglot-parity.qmd`.
3. In R, load the NHANES CSV with a relative path.
4. Produce a descriptive BMI summary by `IncomeGroup` and `Gender`.
5. Refactor the R summary into one small reusable function.
6. Create `translation.ipynb` and translate the same summary to Python/pandas.
7. Create `helpers.py` with at least one reusable Python helper called from the notebook.
8. Complete a parity table comparing row counts, filtered row counts, grouping labels, and rounded summary values.
9. Write a dependency note naming the R and Python packages used. If an instructor-approved Python package is added, also update the root `.devcontainer/requirements.txt`; do not create a second manifest in the submission folder.
10. Clear bulky notebook outputs before committing.
11. Render the Quarto file to HTML.
12. Commit and sync the assignment folder.

## AI-Use Expectations

AI may help translate R to Python, explain tracebacks, or suggest helper functions. You must verify that the translated code answers the same question, uses the same filters, handles missingness the same way, and returns matching grouped results after rounding. Include `ai-use-note.md`.

## Reproducibility Requirements

- R and Python files must run in GitHub Codespaces.
- Paths must be relative to the repository.
- The notebook must not contain large stale outputs.
- Dependencies must be documented in `dependency-note.md`.

- Differences between R and Python results must be explained before changing code.

## What To Submit

Place all submission files in:

```
1 assignments/assignment04-polyglot/submission/
```

Submit these files:

- `polyglot-parity.qmd`: R analysis, R function, parity discussion, and rendered-output instructions.
- `polyglot-parity.html`: rendered Quarto output.
- `translation.ipynb`: Python/pandas translation notebook.
- `helpers.py`: reusable Python helper script called from the notebook.
- `parity-table.csv`: completed parity table.
- `dependency-note.md`: Python/R dependency note.
- `ai-use-note.md`: what AI helped translate or debug and how you verified the result.

## Submission Route

Submit your GitHub repository link on Canvas after committing and syncing the required files.

## Grading Checklist

Canvas applies a 7-point rubric. This assignment is part of the best-4-of-6 set for Assignments 4-9:

| Criterion                                                  | Points |
|------------------------------------------------------------|--------|
| R summary is correct, clear, and uses a relative data path | 1      |
| R function reproduces the summary logic                    | 1      |

| Criterion                                                  | Points |
|------------------------------------------------------------|--------|
| Python translation matches the same analytic question      | 1      |
| Parity table checks row counts, groups, and rounded values | 1.5    |
| Dependency and notebook hygiene are documented             | 1      |
| AI-use note explains generation, changes, and verification | 1      |
| Files render/run from the submission folder and are synced | 0.5    |

## Definition of Done

You are done when the R summary, Python translation, and parity table agree or any differences are clearly explained, and the required files are committed in the Assignment 4 submission folder.

# Assignment 5: Visualization Audit

## Purpose

This assignment turns a descriptive summary into an honest visual explanation. You will create a `ggplot2` visualization, audit a flawed plot, revise your own design, and write a caption that states the statistic, data source, exclusions, and limitations.

## Learning Objectives

- Create a clear `ggplot2` visualization from the NHANES classroom dataset.
- Identify misleading scale, title, grouping, missingness, and caption choices.
- Revise a plot so the visual design matches the evidence.
- Explain the descriptive and unweighted limits of the figure.
- Connect visualization readiness to the M1 project proposal.

## Inputs

- Dataset: `examples/nhanes-equity/data/nhanes_equity_v6.csv`
- Worked example: `weeks/week06-visualization/viz03-worked-example.qmd`
- Studio activity: `weeks/week06-visualization/viz06-studio-visual-audit.qmd`
- Walkthrough and reference: `weeks/week06-visualization/viz07-assignment-a5-and-reference.qmd`
- Flawed plot starter in your personal workspace:  
`assignments/assignment05-visualization/flawed-plot-code.R`

- Visual-audit template in your personal workspace:  
`assignments/assignment05-visualization/visual-audit-note-template.md`
- M1 connection: `milestones/m1-proposal/README.md`

## Tasks

1. Create or use the folder `assignments/assignment05-visualization/submission/`.
2. Review the Week 6 worked example.
3. Choose one descriptive comparison from the NHANES CSV snapshot.
4. Draft an initial plot using R and `ggplot2`.
5. Audit the flawed plot starter and your own draft for scale choices, labels, missing context, aggregation, and over-claiming.
6. Revise your plot so it supports a descriptive interpretation.
7. Export the corrected plot as `corrected-plot.png`.
8. Write a caption that includes data source, statistic, exclusions, and limitation.
9. Add a `plotly` preview [optional — include only if relevant to your project]: only if it helps you inspect the figure; the required submission is still the static plot.
10. Write a short M1 link note naming how the visualization standard affects your project dataset, audience, or feasibility.

## AI-Use Expectations

AI may help draft `ggplot2` code, suggest labels, or identify possible visual problems. You must verify the code against real column names, check that the scale does not exaggerate the result, and remove causal or population-level claims that the plot cannot support. Include `ai-use-note.md`.

## Reproducibility Requirements

- The Quarto file must render in GitHub Codespaces.

- The dataset path must be relative.
- The plot image must be exported by code or a documented render step.
- Captions must state that the analysis is descriptive and unweighted when using the classroom NHANES snapshot.

## What To Submit

Place all submission files in:

```
1 assignments/assignment05-visualization/submission/
```

Submit these files:

- `visualization-audit.qmd` [required]: rerunnable Quarto file that loads the CSV, creates the final plot, and explains the revision.
- `visualization-audit.html` [required]: rendered output.
- `corrected-plot.png` [required]: exported final plot.
- `visual-audit-note.md` [required]: at least three issues checked or corrected.
- `caption.md` [required] [brief — 1-3 focused bullets or sentences are enough]: final figure caption.
- `m1-link-note.md` [required] [brief — 1-3 focused bullets or sentences are enough]: two or three sentences connecting the visualization standard to M1.
- `ai-use-note.md` [required]: what AI helped with and how you checked it, or a sentence saying AI was not used.

## Submission Route

Submit your GitHub repository link on Canvas after committing and syncing the required files.



## Grading Checklist

Canvas applies a 7-point rubric. This assignment is part of the best-4-of-6 set for Assignments 4-9:

| Criterion                                                                     | Points |
|-------------------------------------------------------------------------------|--------|
| Plot is generated by rerunnable <code>ggplot2</code> code with relative paths | 1.5    |
| Visual audit identifies at least three meaningful issues                      | 1      |
| Corrected plot improves scale, labeling, grouping, or caption clarity         | 1.5    |
| Caption states statistic, source, exclusions, and limitations                 | 1      |
| Interpretation stays descriptive and non-causal                               | 0.75   |
| AI-use note and M1 link note are specific                                     | 0.75   |
| Required files are committed and synced                                       | 0.5    |

## Definition of Done

You are done when a peer can render `visualization-audit.qmd`, see the corrected plot, understand the main comparison from the caption, and identify how you checked the plot for misleading design.

# Assignment 6: EDA and AI Audit

## Purpose

This assignment builds a transparent preliminary analysis. You will define a cohort, check missingness, create a Table 1-style descriptive summary, audit planted errors, and revise a provenance/stewardship note for reuse in M2.

## Learning Objectives

- Define an analysis cohort before summarizing results.
- Report missingness for key variables.
- Create a Table 1-style summary with N, mean, and SD.
- Detect planted correctness, reproducibility, interpretation, and stewardship errors.
- Write descriptive methods text that avoids causal overclaiming.

## Inputs

- Dataset: `examples/nhanes-equity/data/nhanes_equity_v6.csv`
- Worked example: `weeks/week07-eda-ai-audit/eda03-worked-example.qmd`
- Studio activity: `weeks/week07-eda-ai-audit/eda06-studio-planted-errors.qmd`
- Walkthrough and reference: `weeks/week07-eda-ai-audit/eda07-assignment-a6-and-reference.qmd`
- Planted-error starter in your personal workspace:  
`assignments/assignment06-eda-ai-audit/planted-error-starter.qmd`
- Audit template in your personal workspace: `assignments/assignment06-eda-ai-audit/audit-note-temp`
- M2 connection: `milestones/m2-preliminary-analysis/README.md`

## Tasks

1. Create or use the folder `assignments/assignment06-eda-ai-audit/submission/`.
2. Load the NHANES CSV snapshot with a relative path.
3. Define and document an analysis cohort using age 20-80.
4. Create a missingness table for key variables.
5. Produce a Table 1-style summary by `IncomeGroup`.
6. Include N, BMI mean (SD), and age mean (SD).
7. Write a short methods note stating that the analysis is descriptive, unweighted, and non-causal.
8. Audit the planted-error starter and list at least five issues.
9. Categorize each issue as correctness, reproducibility, interpretation, or stewardship.
10. Revise the provenance/stewardship statement so it names the data source, classroom-use context, and one privacy or stewardship caution.

## AI-Use Expectations

AI may help draft code, explain an error, or improve the organization of the audit note. You must verify cohort filters, missingness handling, labels, summary statistics, and interpretation against the code and output. Include `ai-use-note.md`.

`ai-use-note.md` is the graded AI artifact; the inline `# AI prompt: / # Verified:` comments in your code are the required working practice that feeds it — copy your comment pairs into the note. The template and a filled example are on the [assignments index](#).

## Reproducibility Requirements

- The EDA note must render in GitHub Codespaces.
- Paths must be relative.
- `table1.csv` must be generated by code.
- Missingness handling must be stated before interpreting summaries.

- The methods note must avoid causal and population-level claims unsupported by the unweighted classroom example.

## What To Submit

Place all submission files in:

```
1 assignments/assignment06-eda-ai-audit/submission/
```

Submit these files:

- `eda-note.qmd`: rerunnable Quarto file with cohort definition, missingness table, Table 1, and methods note.
- `eda-note.html`: rendered output.
- `table1.csv`: exported Table 1 or equivalent descriptive summary.
- `audit-note.md`: at least five planted issues with category, why it matters, and corrected approach.
- `provenance-stewardship-note.md`: short revised note suitable for reuse in M2.
- `ai-use-note.md`: what AI helped with and how you checked it, or a sentence saying AI was not used.

## Submission Route

Submit your GitHub repository link on Canvas after committing and syncing the required files.

## Grading Checklist

Canvas applies a 7-point rubric. This assignment is part of the best-4-of-6 set for Assignments 4-9:

| Criterion                                                                                                                                    | Points |
|----------------------------------------------------------------------------------------------------------------------------------------------|--------|
| Cohort uses the required age 20-80 filter and exclusions are documented before results                                                       | 1      |
| Missingness table is complete for key variables                                                                                              | 1      |
| Table 1 includes N, BMI mean (SD), and age mean (SD) by group                                                                                | 1.5    |
| Methods note is descriptive, unweighted, and non-causal                                                                                      | 1      |
| Audit note identifies at least five planted issues across the four categories: correctness, reproducibility, interpretation, and stewardship | 1.25   |
| Provenance/stewardship and AI-use notes are specific                                                                                         | 0.75   |
| Required files render, export, commit, and sync                                                                                              | 0.5    |

The provenance/stewardship note and the AI-use note share one rubric row, but they are two separate required files with different jobs: the stewardship note is about the data and its responsible use, while the AI-use note is about how you used and verified AI. Write each one fully — a strong AI-use note does not cover for a missing stewardship note, or the reverse.

## Definition of Done

You are done when `eda-note.qmd` renders, `table1.csv` is re-generated by code, and the audit note explains at least five planted errors and corrected approaches.

# Assignment 7: Dashboard KT Prototype

## Purpose

This assignment introduces dashboard-style Knowledge Translation. You will run or render a small dashboard product, identify the audience and question, document one control or interaction, improve one audience-facing element, test the product, and state a limitation.

**What you are not doing in A7:** you are not building a full Shiny app, deploying a live dashboard, changing the dataset, or adding new analysis. You are making a small documentation or audience-facing text improvement to the starter dashboard. A live dashboard is not required unless the assignment says so. A static preview, screenshot, or rendered HTML page is acceptable when it clearly shows the KT product and passes the privacy check.

## Learning Objectives

- Explain how a dashboard differs from a static report.
- Identify an audience-specific KT question.
- Document one control, filter, or dashboard-style interaction.
- Connect one visualization to a plain-language interpretation.
- Test a dashboard product in Codespaces and record the result.

## Inputs

- Week 8 index: `weeks/week08-dashboard-kt/index.qmd`
- Required core artifact in your personal workspace: `assignments/assignment07-dashboard/dashboard-starter.qmd`
- Walkthrough and reference: `weeks/week08-dashboard-kt/dash07-assignment-a7-and-reference.qmd`
- Shiny exemplar (instructor demonstration only; not an A7 submission pathway): `examples/nhanes-equity/app/app.R`
- Cached data in your personal workspace: `examples/nhanes-equity/data/nhanes_equity_v6.rds`
- CSV snapshot in your personal workspace: `examples/nhanes-equity/data/nhanes_equity_v6.csv`
- Adaptation template: `assignments/assignment07-dashboard/dashboard-adaptation-template.md`

## Tasks

1. Create or use the folder `assignments/assignment07-dashboard/submission/`.
2. Copy `assignments/assignment07-dashboard/dashboard-starter.qmd` to `assignments/assignment07-dashboard/submission/dashboard.qmd`. Adapt and render this copy from the repository root. The Shiny exemplar is a demonstration and is not an approved A7 submission pathway without prior instructor approval.
3. Choose one audience, such as a public-health analyst, student project team, or briefing group.
4. Write one audience-specific dashboard question.
5. Identify one control, filter, or dashboard-style interaction and the visualization it affects.
6. Identify one label, caption, help text, interpretation note, or README sentence that could be clearer for that audience.
7. Make one documentation-only or UI-text-only improvement. Do not change analytic logic for this assignment.
8. Re-render the starter dashboard with a different control value and record the test result.
9. Write one limitation statement that belongs in or beside the dashboard.
10. Complete the privacy-safe publishing check before sharing screenshots or rendered outputs.



## AI-Use Expectations

AI may help improve wording for labels, captions, or limitation statements. You must verify that the revised language still matches the dashboard output and does not imply causality, population inference, or access to row-level data. Include `ai-use-note.md`.

## Reproducibility Requirements

- The dashboard or starter dashboard must run or render in GitHub Codespaces.
- Paths must be relative.
- The testing log must name the command or render path used.
- Screenshots must avoid exposing row-level data.
- The submitted adaptation must not require private credentials or local-only files.

## What To Submit

Place all submission files in:

```
1 assignments/assignment07-dashboard/submission/
```

Submit these files:

- `dashboard-adaptation-note.md` [required]: audience, question, control/interaction, visualization, changed element, and KT rationale.
- `dashboard.qmd` [required]: the adapted Quarto source used to create the preview.
- `before-after-text.md` [required] [brief — 1-3 focused bullets or sentences are enough]: original text and revised text.
- `testing-log.md` [required] [brief — 1-3 focused bullets or sentences are enough]: command or render path used, date tested, and result. A few lines is enough.

- **limitation-note.md** [required] [brief — 1-3 focused bullets or sentences are enough]: one limitation or caution suitable for the dashboard. One clear sentence is enough.
- **privacy-check.md** [required] [brief — 1-3 focused bullets or sentences are enough]: evidence that no row-level data, small-cell overclaim, or causal overclaim is shared. Three short sentences are enough.
- **dashboard-preview.png** [required]: screenshot or exported preview of the affected dashboard area. This is evidence, not a designed product — a plain screenshot is fine.
- **reproducibility-note.md** [required] [brief — 1-3 focused bullets or sentences are enough]: how the dashboard was run or rendered in Codespaces. One short paragraph is enough.
- **ai-use-note.md** [required]: what AI helped with and how you checked it, or a sentence saying AI was not used.

## Submission Route

Submit your GitHub repository link on Canvas after committing and syncing the required files.

## Grading Checklist

Canvas applies a 7-point rubric. This assignment is part of the best-4-of-6 set for Assignments 4-9:

| Criterion                                                                | Points |
|--------------------------------------------------------------------------|--------|
| Audience and dashboard question are specific                             | 1      |
| Control/filter or interaction and visualization are described accurately | 1      |

| Criterion                                                                                  | Points |
|--------------------------------------------------------------------------------------------|--------|
| Documentation-only or UI-text-only change improves clarity without changing analytic logic | 1.25   |
| Interpretation and limitation are descriptive and non-causal                               | 1      |
| Testing and reproducibility notes show a successful local check                            | 1      |
| Privacy-safe publishing check is complete                                                  | 0.75   |
| Dashboard source, preview, AI-use note, and required files are committed and synced        | 1      |

## Definition of Done

You are done when the submitted `dashboard.qmd`, preview, and notes let a reviewer identify the audience, control, visualization, text improvement, test result, and limitation and reproduce the preview if needed.

# Assignment 8: Scientific Communication

## Purpose

This assignment turns one result from your analysis or dashboard into a short audience-specific communication product. You will create a small slide deck or slide outline, write a plain-language summary, cite the data source, and use peer feedback to improve the claim.

## Learning Objectives

- Translate one technical result into a clear audience-facing takeaway.
- Build a short Quarto revealjs deck or structured slide outline.
- Use one figure or table from prior work as evidence.
- State a limitation and source note beside the claim.
- Give and apply peer feedback on clarity, evidence, and KT value.

## Inputs

- Week 9 page: `weeks/week09-science-communication/index.qmd`
- Peer feedback page: `weeks/week09-science-communication/comm04-peer-feedback.qmd`
- Renderable slide starter in your personal workspace: `assignments/assignment08-communication/starter-slides.qmd`
- Slide-outline starter in your personal workspace: `assignments/assignment08-communication/slide-outline-template.qmd`

- Peer-feedback template in your personal workspace:  
`assignments/assignment08-communication/peer-feedback-template.md`
- Prior work from Assignment 5, 6, or 7
- Submission folder: `assignments/assignment08-communication/submission/`

## Tasks

1. Create or use the folder `assignments/assignment08-communication/submission/`.
2. Choose one figure, table, or dashboard finding from prior work.
3. Define the audience in one sentence.
4. Write one plain-language takeaway that a non-technical reader can understand.
5. Create either `slides.qmd` as a revealjs deck or `slide-outline.qmd` as a structured slide outline. Copy the corresponding starter from `assignments/assignment08-communication/` in your personal workspace into its `submission/` folder and rename it before editing. Do not edit the original starter.
6. Include 3-5 slides or slide sections: context, evidence, takeaway, limitation, and next step.
7. Add one source note or citation for the dataset.
8. Complete the in-studio partner review early enough to make at least one A8 revision before submission. Also complete the Canvas Discussion exchange by the standard syllabus deadline — Monday after the Week 9 class, 4:00 PM. Canvas feedback may inform the presentation deck or final-portfolio revisions; it is not the feedback source required before the initial A8 submission.
9. Revise at least one sentence or slide based on feedback.
10. Document any AI assistance used for simplification, slide wording, or feedback synthesis.

## AI-Use Expectations

AI may help simplify jargon, suggest slide structure, or tighten wording. You must verify that the simplified text still matches

the evidence, citation, and limitation. AI must not invent findings, sources, or audience needs. Include `ai-use-note.md`.

## Reproducibility Requirements

- The deck or outline must render in GitHub Codespaces.
- Any figure or table must be traceable to prior code or a submitted output.
- Source notes and citations must be visible.
- Paths must be relative.
- Peer feedback must be specific enough to act on.

## What To Submit

Place all submission files in:

```
1 assignments/assignment08-communication/submission/
```

Submit these files:

- `slides.qmd` or `slide-outline.qmd` [required]: source file for the communication product — submit one of the two.
- `slides.html` or `slide-outline.html` [required]: rendered output. Commit this required HTML with its source.
- `plain-language-summary.md` [required]: 150-250 word audience-facing summary.
- `peer-feedback.md` [required]: feedback received and one revision made. Expected length: the completed feedback form plus one or two sentences describing the revision you made — do not write an essay.
- `source-note.md` [required] [brief — 1-3 focused bullets or sentences are enough]: dataset/source note and citation information.
- `ai-use-note.md` [required]: what AI helped with and how you checked it, or a sentence saying AI was not used.

## Submission Route

Submit your GitHub repository link on Canvas after committing and syncing the required files.

## Grading Checklist

Canvas applies a 7-point rubric. This assignment is part of the best-4-of-6 set for Assignments 4-9. The full revealjs deck and the structured slide outline are graded identically — choosing the outline does not lower your possible score.

| Criterion                                                   | Points |
|-------------------------------------------------------------|--------|
| Audience and KT purpose are specific                        | 1      |
| Slide deck or outline has a clear 3-5 part story            | 1.25   |
| Figure/table evidence is traceable and accurately described | 1      |
| Plain-language summary is clear and avoids overclaiming     | 1      |
| Source note and limitation are visible                      | 1      |
| Peer feedback is specific and revision is documented        | 1      |
| AI-use note and render/sync requirements are complete       | 0.75   |

## Definition of Done

You are done when a peer can read the deck or outline, identify the main takeaway, see the evidence and limitation, and understand what changed after feedback.

If you submit an outline for A8, it receives the same A8 score as a deck. The narrated presentation deck is a later stage and requires `slides.qmd`, so convert the revised A8 story into a renderable deck for the portfolio rather than treating the A8 outline as final slides.

# Assignment 9: Reproducible Report

## Purpose

This assignment expands a project finding into a short reproducible Quarto report. The report should regenerate a table and figure from code, cite the data source, state limitations, and prepare the work for M3 Project Update review. Write the main report for a non-technical decision-maker audience; technical detail such as code choices and sensitivity notes belongs in a technical appendix.

## Learning Objectives

- Write a Quarto report that combines narrative, code, tables, figures, and citations.
- Generate at least one table and one figure from code.
- Use BibTeX citations and a visible source note.
- Include limitations and a grounding audit for claims.
- Prepare a preview or publishing check suitable for M3 review.

## Inputs

- Week 10 page: `weeks/week10-reporting-publishing/index.qmd`
- Report template in your personal workspace: `assignments/assignment09-reporting/report-template.qmd`
- Citations how-to: [Week 10 citations and grounding page](#)  
— start here if you have not used `references.bib` before
- Bibliography starter in your personal workspace: `assignments/assignment09-reporting/references.bib`



- M3 connection: `milestones/m3-project-update/README.md`
- Publishing checklist template: `assignments/assignment09-reporting/publishing-checklist-template`
- Grounding audit template: `assignments/assignment09-reporting/grounding-audit-template.md`

## Tasks

1. Create or use the folder `assignments/assignment09-reporting/submission/`.
2. Copy `report-template.qmd` and `references.bib` from `assignments/assignment09-reporting/` into its `submission/` folder. Rename the report copy `report.qmd`, then edit the copies rather than the starters.
3. Define one descriptive project question. If your project dataset is not ready, use the NHANES classroom dataset and explain that this is a practice report.
4. Load the data with a relative path.
5. Generate at least one table and one figure from code.
6. Add citations using a local `references.bib` file or the course bibliography workflow.
7. Write a limitations section that avoids causal overclaiming.
8. Complete a grounding audit: each claim should point to code output, a citation, or a stated limitation.
9. Complete a publishing check: GitHub Pages preview if available, or a private rendered HTML preview in the repository.
10. Write an M3 update note explaining what is ready for peer review and what feedback would be useful.
11. Before committing, search your submission files for absolute or local paths (for example `C:/` or `/Users/`) and replace any you find with relative paths.

## AI-Use Expectations

AI may help outline the report, debug Quarto, improve wording, or identify unclear claims. You must verify every generated number, citation, and source claim against the rendered output and bibliography. Include `ai-use-note.md`.

## Reproducibility Requirements

- `report.qmd` must render in GitHub Codespaces.
- Tables and figures must be generated by code.
- Paths must be relative.
- Bibliography entries must resolve.
- Publishing preview must not require private local files.
- The report must complement the dashboard-style product rather than merely duplicate it.

## What To Submit

Place all submission files in:

```
1 assignments/assignment09-reporting/submission/
```

Submit these files:

- `report.qmd`: rerunnable Quarto report.
- `report.html`: rendered report committed with its required source.
- `references.bib`: bibliography entries used by the report.
- `outputs/`: generated table or figure files if the report writes external outputs.
- `publishing-checklist.md`: GitHub Pages or private preview check.
- `grounding-audit.md`: claim-by-claim check against code output, citation, or limitation.
- `m3-update-note.md`: what is ready for M3 review and what feedback is requested. At minimum, name one part of the report that is ready for review and ask one or two specific feedback questions — for example, “Is the cohort definition clear?” or “Does the limitations section cover the missing-data issue?” The [Week 10 A9/M3 page](#) lists more example questions.
- `ai-use-note.md`: what AI helped with and how you checked it, or a sentence saying AI was not used.

## Submission Route

Submit your GitHub repository link on Canvas after committing and syncing the required files.

## Grading Checklist

Canvas applies a 7-point rubric. This assignment is part of the best-4-of-6 set for Assignments 4-9:

| Criterion                                                       | Points |
|-----------------------------------------------------------------|--------|
| Report renders in Codespaces with relative paths                | 1      |
| Includes one code-generated table and one code-generated figure | 1.25   |
| Citations resolve through BibTeX workflow                       | 1      |
| Limitations are accurate and visible                            | 1      |
| Grounding audit checks key claims against evidence              | 1      |
| Publishing checklist and M3 update note are complete            | 1      |
| AI-use note and required files are committed and synced         | 0.75   |

The Task 3 fallback is fully acceptable for grading: a practice report built on the NHANES classroom dataset is marked with this same rubric, as long as the report states that it is a practice report.

## Definition of Done

You are done when `report.qmd` renders to `report.html`, citations resolve, outputs are generated by code, and the M3 update note tells reviewers exactly what feedback you need.

## Milestones

# Milestones

## Worked example

New to milestones? Skim the [NHANES worked example](#) — one complete illustrative project from M0 to the Final Portfolio.

## Looking for due dates?

The [Assignment Schedule](#) lists every milestone alongside the weekly assignments in date order.

Milestones move the term project from team setup to final portfolio. They are short checkpoints, not separate side projects. Each one should make the next stage easier to complete and easier for another person to review.

## Submission Contract

For every milestone:

- commit and sync the required files before submitting
- submit the GitHub repository link on Canvas
- include the latest commit hash in the Canvas comment box
- use relative paths
- document any package or data access requirements
- disclose and audit AI use

## Milestone Map

| Milestone       | Timing                        | Focus                                          | Link                                     |
|-----------------|-------------------------------|------------------------------------------------|------------------------------------------|
| M0              | Week 4                        | Group formation and repository access          | <a href="#">M0: Group Formation</a>      |
| M1              | Week 6                        | Project proposal and data intake               | <a href="#">M1: Project Proposal</a>     |
| M2              | Week 7                        | Preliminary analysis and Table 1-style summary | <a href="#">M2: Preliminary Analysis</a> |
| M3              | Week 10                       | Draft report and dashboard-style update        | <a href="#">M3: Project Update</a>       |
| M4              | Week 11                       | Peer review and reproducibility feedback       | <a href="#">M4: Peer Review</a>          |
| Presentation    | With Final Portfolio (Dec 11) | Narrated slide deck (online)                   | <a href="#">Presentation</a>             |
| Final Portfolio | Final deadline                | Complete project record                        | <a href="#">Final Portfolio</a>          |

## Project Points and Score Ownership

Project scores use direct points that match the course weights: M1–M4 are 5 points each, the Presentation is 10 points, and the Final Portfolio is 35 points. Staff record a score directly from zero to the stated maximum using the existing checklist as the evidence guide; there is no separate performance-band conversion, contribution multiplier, or adjustment formula.

Shared project artifacts receive one group score. Only a component that a brief already identifies as individually authored

receives an individual score. The current M1–M4, Presentation, and Final Portfolio deliverables are shared project artifacts, so each currently receives one group score.

## **Reproducibility Expectations**

Milestone work should be reviewable in GitHub Codespaces or a documented local setup. Keep data access clear, use relative paths, and avoid manual edits to generated outputs unless the edit is explicitly documented.

## **AI-Use Expectations**

AI tools may help with drafting, code review, debugging, and clarity checks. Students remain responsible for correctness. Each milestone includes an AI-use note so reviewers can see what was AI-assisted and what was verified by the team.

## **Definition Of Done**

A milestone is ready when the required files are present, the repository is synced, the Canvas submission points to the correct commit, and another student could understand what changed without private explanation.



# M0: Group Formation

 See a worked example

The [M0 submission in the NHANES worked example](#) shows one illustrative version of this milestone. Use it for structure and depth — your project must be your own.

## Purpose

Project groups are 3–4 students. M0 makes sure every project team has a working collaboration setup before analytic work begins. By the end of this milestone, each group should know who is on the team, where the shared repository lives, how communication will happen, and how the team will handle reproducibility and AI use.

Your shared repository starts as a fork of the group project template linked from Canvas — do not build it as a blank repository. The fork steps are walked through on the [Week 4 group repository page](#).

## Due Timing

Submit by the Week 4 deadline posted in Canvas.

 How M0 is graded

M0 is marked **complete/incomplete**. There is no point score; this milestone is meant to unblock the project work that follows.

## Required Deliverables

Create a `submission/` folder in this milestone directory:

```
1 milestones/m0-group-formation/submission/
```

Add these files. All five are required; tags follow [How To Read Submission Lists](#).

- `group-formation.md` [required]
- `repo-access-evidence.md` [required]
- `communication-plan.md` [required]
- `codespace-check.md` [required]
- `ai-use-note.md` [required]

## What To Submit

Submit your GitHub repository link on Canvas after committing and syncing the files above. In the Canvas comment box, include the latest commit hash for the submission.

## File Guidance

Every bullet below is a required field, not an example. One short line per field is enough.

`group-formation.md` should include:

- group name
- student names
- GitHub usernames
- one-sentence project area or dataset interest
- primary repository owner

`repo-access-evidence.md` should include:

- shared repository link
- confirmation that all group members can view the repo

- confirmation that all group members can open the repo in Codespaces or a local equivalent

`communication-plan.md` should include:

- primary communication channel
- expected response window
- weekly check-in plan
- how the group will document decisions

`codespace-check.md` should include:

- who tested the repository
- what command or render was tested
- whether the test passed
- any setup issue and the planned fix

`ai-use-note.md` should include:

- whether AI tools were used to draft the group plan
- what the group verified independently
- what the group will not outsource to AI

## Reproducibility Expectations

Use relative paths and keep project files inside the shared repository. If a group member needs a package, script, or external file to reproduce work, document it in the repository README before the next milestone.

## AI-Use Expectations

AI tools may help draft communication norms or check clarity. The team remains responsible for verifying repository access, Codespaces setup, project scope, and all submitted claims.

## Grading Checklist

This milestone is marked complete when:

- all group members are listed with GitHub usernames
- shared repository access is documented
- communication norms are specific enough to use
- Codespaces or local setup has been tested
- AI-use expectations are acknowledged
- files are committed, synced, and linked through Canvas

## Definition Of Done

M0 is done when every group member can reach the shared repository, knows the team communication plan, and can explain how the project will be kept reproducible from this point forward.

# M1: Project Proposal

 See a worked example

The [M1 submission in the NHANES worked example](#) shows one illustrative version of this milestone. Use it for structure and depth — your project must be your own.

## Purpose

M1 turns a broad project idea into a scoped, feasible, and ethically grounded analytic plan. The proposal should make the audience, data source, question, reproducibility plan, and first visualization direction clear enough for feedback.

## Due Timing

Submit by the Week 6 deadline posted in Canvas. This milestone prepares the group for Week 7 preliminary analysis and Week 8 dashboard prototyping.

## Points and Score Ownership

M1 is worth **5 course points** and receives one group score. The checklist below guides the direct 0–5 score; no contribution multiplier or performance-band conversion is used.

The five points are allocated across these broad criteria:

| Criterion                                                                | Points   |
|--------------------------------------------------------------------------|----------|
| Proposal scope, audience, KT purpose, and questions                      | 1        |
| Data provenance, access, privacy, stewardship, and relevant equity risks | 1        |
| Initial visual direction and risk of misinterpretation                   | 1        |
| Feasibility and reproducibility plan                                     | 1        |
| AI-use audit and complete, synced submission evidence                    | 1        |
| <b>Total</b>                                                             | <b>5</b> |

These are broad judgments, not collections of micro-deductions. A substantively complete, internally consistent proposal can earn full credit without research-grade polish.

## Required Deliverables

Create or update:

```
1 milestones/m1-proposal/submission/
```

Include these files. All five are required; tags follow [How To Read Submission Lists](#).

- `proposal.md` [required]
- `data-intake-card.md` [required]
- `initial-visual-plan.md` [required]
- `reproducibility-plan.md` [required]
- `ai-use-note.md` [required]

## What To Submit

Submit your GitHub repository link on Canvas after committing and syncing the files above. In the Canvas comment box, include the latest commit hash and the folder path for the milestone.

## Proposal Guidance

Every field below is [required] unless it carries its own tag.

`proposal.md` should include:

- project title
- group members
- audience and KT purpose
- one primary question
- one secondary question
- expected data source
- planned analytic output
- known limitations

KT purpose means the practical communication goal of your project: who needs to understand the result, what decision or conversation it supports, and what caution they need.

The secondary question is not the primary question in different words. It should narrow or extend the primary one. For example, primary: “How does average systolic blood pressure differ across income groups in NHANES adults?” Secondary: “Does that difference look similar for men and women?”

`data-intake-card.md` should summarize:

- data source and steward
- access method
- file format
- unit of analysis
- key variables
- privacy and stewardship considerations

- Indigenous Data Sovereignty or CARE/OCAP awareness [when relevant — required if your data involve Indigenous Peoples, communities, or services]

CARE/OCAP are principles and frameworks for responsible governance of Indigenous data. If your dataset includes Indigenous identifiers, communities, territories, or services, link back to the [Week 1 guidance](#) before proceeding.

`initial-visual-plan.md` should include:

- one planned figure
- what the figure should help the audience decide or understand
- one risk of misinterpretation

`reproducibility-plan.md` should include:

- expected repository folders
- data access plan
- package/dependency notes
- how another student could rerun the work

## Reproducibility Expectations

Use relative paths. Do not commit restricted, private, or identifiable data. If public data are too large for GitHub, commit a small sample or documented access script and explain how the full file is obtained.

## AI-Use Expectations

AI may help brainstorm wording, search terms, or code structure. Students must verify data-source claims, privacy considerations, citations, and feasibility. Include an `ai-use-note.md` describing what AI helped with and what was checked manually.



## Full-Credit Preparation Checklist

A substantively complete submission should show:

- the question is specific and feasible
- the audience and KT purpose are named
- data provenance and access are documented
- privacy, stewardship, and equity risks are acknowledged
- at least one visualization direction is described
- reproducibility plan is concrete
- AI use is disclosed and audited

## Definition Of Done

M1 is done when another group can read the proposal and understand what you will analyze, why it matters, what data you will use, and how the work will be rerun.

## M2: Preliminary Analysis

 See a worked example

The [M2 submission in the NHANES worked example](#) shows one illustrative version of this milestone. Use it for structure and depth — your project must be your own.

### Purpose

M2 checks that the project can move from proposal to evidence. The milestone should show a documented cohort or analytic sample, a missingness check, a descriptive summary, and a careful interpretation that avoids overclaiming.

### Due Timing

Submit by the Week 7 deadline posted in Canvas. This milestone prepares the group for Week 8 dashboard prototyping and Week 9 communication work.

### Points and Score Ownership

M2 is worth **5 course points** and receives one group score. The checklist below guides the direct 0–5 score; no contribution multiplier or performance-band conversion is used.

The five points are allocated across these broad criteria:

| Criterion                                                | Points   |
|----------------------------------------------------------|----------|
| Cohort definition and inclusion/exclusion decisions      | 1        |
| Missingness check and Table 1-style descriptive evidence | 1        |
| Reproducible source, render, and code-generated outputs  | 1        |
| Methods note, interpretation, and claim boundary         | 1        |
| AI-use audit and complete, synced submission evidence    | 1        |
| <b>Total</b>                                             | <b>5</b> |

These are broad judgments, not collections of micro-deductions. A clear, rerunnable descriptive analysis can earn full credit without advanced methods or research-grade polish.

## Required Deliverables

Create or update:

```
1 milestones/m2-preliminary-analysis/submission/
```

Include these files. All six are required — none are optional — but three (`table1.csv`, `missingness-summary.csv`, `methods-note.md`) are small outputs or short notes produced alongside the analysis, not separate projects. Tags follow [How To Read Submission Lists](#).

- `preliminary-analysis.qmd` [required]
- `preliminary-analysis.html` [required]
- `table1.csv` [required]
- `missingness-summary.csv` [required]
- `methods-note.md` [required]
- `ai-use-note.md` [required]

Committing `preliminary-analysis.html` here is deliberate: every rendered file explicitly required by an assessment is committed with its source.

## What To Submit

Submit your GitHub repository link on Canvas after committing and syncing the files above. In the Canvas comment box, include the latest commit hash and the folder path for the milestone.

## Analysis Requirements

Your preliminary analysis should include:

- a clear analytic sample or cohort definition
- a short explanation of inclusion and exclusion decisions
- a missingness check for key variables
- a Table 1-style descriptive summary
- at least one figure or table generated by code
- a methods note that states what the summary can and cannot claim

“Table 1-style” is health-research shorthand for the first table of a paper: one row per characteristic of your analytic sample (for example age, sex, income group), with counts, percentages, and means or medians. Follow the [Week 7 worked example](#) for the expected format.

Your group may build on one member’s Assignment 6 work. If you do, add one attribution sentence to `methods-note.md`, for example: “The preliminary analysis builds on Jordan’s A6 `eda-note.qmd`.” Export the missingness table by code as `missingness-summary.csv`; this is an M2-only export and is not an additional A6 submission file.

## Reproducibility Expectations

The `.qmd` file must render in GitHub Codespaces or a documented local environment. Use relative paths, document packages, and avoid manual edits to generated tables or figures.

## AI-Use Expectations

AI may help draft code, suggest summaries, or review language.

In your `ai-use-note.md`, name at least three checks you performed, such as checking row counts, variable definitions, missingness, grouping labels, and whether AI changed the meaning of the result. The [Week 3 AI audit checklist](#) and the [Week 7 audit categories](#) list more checks to choose from.

## Full-Credit Preparation Checklist

A substantively complete submission should show:

- cohort definition is explicit
- missingness is reported
- Table 1-style output includes counts and descriptive statistics
- code-generated output is reproducible
- interpretation is descriptive and non-causal unless justified
- AI-generated help is disclosed and audited
- files are committed, synced, and submitted through Canvas

## Definition Of Done

M2 is done when a peer can rerun the analysis, see the same descriptive outputs, and understand the limits of the claims being made.

## M3: Project Update

 See a worked example

The [M3 submission in the NHANES worked example](#) shows one illustrative version of this milestone. Use it for structure and depth — your project must be your own.

### Purpose

M3 turns the preliminary analysis into a project draft that is ready for feedback. The update should show the current evidence, the intended KT product, and the main decisions that still need peer or instructor critique.

### Due Timing

Submit by the Week 10 deadline posted in Canvas. This milestone supports report drafting, publishing checks, and the Week 11 reproducibility review.

### Points and Score Ownership

M3 is worth **5 course points** and receives one group score. The checklist below guides the direct 0–5 score; no contribution multiplier or performance-band conversion is used.

The five points are allocated across these broad criteria:

| Criterion                                                                  | Points   |
|----------------------------------------------------------------------------|----------|
| Project update framing,<br>current finding, and<br>decision needing review | 1        |
| Rendered, evidence-based<br>report draft                                   | 1        |
| Dashboard/KT preview,<br>audience fit, and caution                         | 1        |
| Targeted feedback<br>request and<br>reproducibility guidance               | 1        |
| AI-use audit and<br>complete, synced<br>submission evidence                | 1        |
| <b>Total</b>                                                               | <b>5</b> |

These are broad judgments, not collections of micro-deductions. M3 is a draft checkpoint: substantively complete work can earn full credit while still identifying choices and revisions for later stages.

## Required Deliverables

Create or update:

1 `milestones/m3-project-update/submission/`

Include these files. Seven files looks like a lot, but only `report-draft.qmd` involves substantial work, and it grows out of your M2 analysis; the `.md` files are short notes. Tags follow [How To Read Submission Lists](#).

- `project-update.md` [required]
- `report-draft.qmd` [required]
- `report-draft.html` [required]
- `dashboard-preview.md` [required] [brief — 1-3 focused bullets or sentences are enough]
- `reproducibility-note.md` [required]

- `feedback-request.md` [required]
- `ai-use-note.md` [required]

Commit `report-draft.html` with its source. The course-wide rule is that every rendered artifact explicitly listed by an assessment is part of the submission.

## What To Submit

Submit your GitHub repository link on Canvas after committing and syncing the files above. In the Canvas comment box, include the latest commit hash, the folder path, and a rendered report link [optional — include only if relevant to your project]. A link only exists if you published or shared a rendered preview; the committed `report-draft.html` is enough on its own.

## Update Requirements

`project-update.md` should include:

- current project title and audience
- refined research or KT question
- data source and analytic sample
- one finding that appears stable
- one finding or choice that needs review
- next steps before peer review

`dashboard-preview.md` is a lightweight preview file. Include a link, screenshot filename, or one short note describing where the current dashboard/KT draft can be found. It is not a full report.

`dashboard-preview.md` should include:

- link or screenshot path for the dashboard-style product
- intended audience
- one interaction, filter, or decision-support feature
- one limitation or privacy caution

`feedback-request.md` should ask for targeted feedback on:



- correctness
- interpretation
- visual clarity
- reproducibility
- audience fit

## Reproducibility Expectations

The report draft must render in GitHub Codespaces or a documented local environment. Relative paths are required. Generated figures and tables should come from code, not manual copying.

## AI-Use Expectations

AI may help revise prose, identify unclear sections, or suggest checks. Students must verify citations, numeric results, interpretations, and any code suggested by AI. The `ai-use-note.md` should name the main AI-assisted tasks and the verification steps.

## Full-Credit Preparation Checklist

A substantively complete submission should show:

- report draft renders
- current findings are supported by code-generated evidence
- dashboard-style product is visible or clearly previewed
- feedback request is specific
- reproducibility note explains how to rerun the work
- AI use is disclosed and audited

## Definition Of Done

M3 is done when another student can open the repo, render the draft report, review the dashboard preview, and provide useful feedback without guessing what the project is trying to do.

## M4: Peer Review

 See a worked example

The [M4 submission in the NHANES worked example](#) shows one illustrative version of this milestone. Use it for structure and depth — your project must be your own.

### Purpose

M4 gives each team structured feedback on correctness, reproducibility, interpretation, and audience fit before final submission. The review should be specific, respectful, and useful enough that the receiving group can act on it.

### Due Timing

Submit by the Week 11 deadline posted in Canvas. Peer review supports the final presentation and final portfolio revision.

### Points and Score Ownership

M4 is worth **5 course points** and receives one group score. The checklist below guides the direct 0–5 score; no contribution multiplier or performance-band conversion is used.

The five points are allocated across these broad criteria:

| Criterion                                                                    | Points   |
|------------------------------------------------------------------------------|----------|
| Specific, prioritized peer review                                            | 1        |
| Documented reproducibility attempt and stopping point                        | 1        |
| Auditability check and blocking-risk assessment                              | 1        |
| Audience-focused KT feedback                                                 | 1        |
| Reviewer reflection, AI disclosure, and complete, synced submission evidence | 1        |
| <b>Total</b>                                                                 | <b>5</b> |

M4 grades the reviewing team’s work, not the quality of the peer project it received. A documented failed rerun can earn full credit when the attempt, stopping point, evidence reviewed, and useful feedback are clear. The criteria are broad judgments, not collections of micro-deductions.

## Required Deliverables

Create or update:

```
1 milestones/m4-peer-review/submission/
```

Include these files. All five are required; tags follow [How To Read Submission Lists](#).

- `peer-review.md` [required]
- `reproducibility-check.md` [required]
- `auditability-check.md` [required]
- `kt-feedback.md` [required]
- `ai-use-note.md` [required]

## What To Submit

Submit your GitHub repository link on Canvas after committing and syncing the files above. In the Canvas comment box, include the reviewed group name and the latest commit hash for your submission.

! Easy to miss: name the group you reviewed

Your Canvas comment must include the reviewed group's name and your latest commit hash. Without the group name, graders cannot match your review to a project.

## Review Requirements

`peer-review.md` should include:

- reviewed project title
- two strengths
- three prioritized revision suggestions
- one question for the group
- one issue that would block final reproducibility if unfixed
- a short reviewer reflection answering: what changed in your own thinking after reviewing this project; which limitation, bias, or stewardship issue deserves the most attention; and one review habit you will apply to your own project

The reflection belongs at the end of `peer-review.md`; do not create a separate `reviewer-reflection.md` file.

`reproducibility-check.md` should document:

- whether the report rendered
- whether paths were relative
- whether data access was clear
- whether figures and tables were generated from code
- any error messages or setup blockers

`auditability-check.md` should document:

- whether data provenance is clear
- whether AI use is disclosed
- whether claims are supported by evidence
- whether limitations are stated

`kt-feedback.md` should focus on:

- audience fit
- plain-language clarity
- visual clarity
- actionability
- privacy-safe communication

## Reproducibility Expectations

Run the reviewed project in GitHub Codespaces when possible. If a full rerun is not possible, document exactly where the process stopped and what evidence you reviewed instead.

If you cannot rerun the reviewed project in Codespaces, that is acceptable only if you document what you tried, where it failed, and what information the project owner would need to fix it. A documented failed rerun does not block completing M4 — an undocumented one does.

## AI-Use Expectations

AI may help organize feedback or identify unclear writing. Reviewers must independently inspect the files, outputs, and claims.

### Do not let AI invent evidence

Do not use AI to invent results or infer evidence that is not in the reviewed repository. Every claim in your review must point to something you actually saw in the files.

## Full-Credit Preparation Checklist

A substantively complete submission should show:

- feedback is specific and actionable
- reproducibility check is documented
- auditability and KT feedback are included
- at least one blocking risk is named or the absence of blockers is justified
- AI use is disclosed
- files are committed, synced, and submitted through Canvas

## Definition Of Done

M4 is done when the reviewed group can use your comments to improve the final portfolio without needing a follow-up meeting to understand the feedback.

# Presentation (Final Portfolio Component)

 See a worked example

The [the Presentation submission in the NHANES worked example](#) shows one illustrative version of this milestone. Use it for structure and depth — your project must be your own.

## Purpose

The presentation is the knowledge-translation slide version of the final project, submitted as a **narrated 5-7 slide deck** and scored as the Presentation component (10 points) of the [Final Portfolio](#). It should explain the audience, question, data, one or two main findings, limitations, and the practical meaning of the work without overclaiming.

## Due Timing

The narrated deck is submitted online with the Final Portfolio on **Dec 11**; there is no in-class presentation session. Record narration over your 5-7 slides (a narrated `slides.html/slides.pdf` export, or a short screen recording) and commit it with the slide source.



## Points and Score Ownership

The presentation is worth **10 course points** and receives one group score. The checklist below guides the direct 0–10 score; no contribution multiplier or performance-band conversion is used.

The ten points are allocated across five broad criteria:

| Criterion                                            | Points    |
|------------------------------------------------------|-----------|
| Audience, purpose, and story                         | 2         |
| Source, evidence, and interpretation                 | 2         |
| Limitations, privacy, and stewardship                | 2         |
| Render, readability, narration, and pacing           | 2         |
| Feedback, revision, reproducibility, and AI evidence | 2         |
| <b>Total</b>                                         | <b>10</b> |

These categories do not reward decorative polish or a particular presentation tool. A plain, readable, evidence-based presentation can earn full credit. One underlying problem is assessed under the category that most directly owns it rather than being automatically deducted in several categories.

## Required Deliverables

Create or update:

```
1 milestones/m5-presentation/submission/
```

Include these files. All are required; tags follow [How To Read Submission Lists](#).

- `slides.qmd` [required]
- `slides.html` or `slides.pdf` [required] — commit this explicitly required rendered file with `slides.qmd`
- narrated recording [required] — a narrated export of the slides or a short screen recording; if narration is embedded in `slides.html/slides.pdf`, note that in `presentation-checklist.md`

- `source-and-limitations-note.md` [required]
- `presentation-checklist.md` [required]
- `peer-feedback-received.md` [required] [brief — 1-3 focused bullets or sentences are enough]
- `ai-use-note.md` [required]

## What To Submit

Submit your GitHub repository link on Canvas after committing and syncing the files above. In the Canvas comment box, include the latest commit hash, the slide path, and any rendered slide link.

## Slide Requirements

The presentation requires a renderable `slides.qmd`. If your A8 submission used the equally valid outline pathway, convert that revised story into the Week 9 slide starter now; this is the intended A8-to-presentation progression, not a new communication task.

Use 5 to 7 slides:

1. Title, team, audience, and KT purpose
2. Question and why it matters
3. Data source, analytic sample, and stewardship note
4. Main figure or table
5. Interpretation and limitation
6. Practical implication or next step
7. Backup or Q&A slide [optional — include only if relevant to your project; a 5- or 6-slide deck with no backup slide loses nothing]

## Source And Limitations Note

`source-and-limitations-note.md` should include:

- data source and citation

- key inclusion or exclusion choices
- whether results are descriptive, predictive, or causal
- one privacy or stewardship caution
- one limitation that should be stated aloud

## Peer Feedback Received

`peer-feedback-received.md` should summarize the feedback your group received, identify the source of the feedback, and state what you changed or chose not to change before finalizing the narrated deck. The main source is the M4 peer review your group received; peer feedback from the Week 9 exchange counts too if you used it. You are summarizing existing feedback, not writing a new response document.

## Reproducibility Expectations

Slides should render in GitHub Codespaces or a documented local environment. Figures and tables should come from the project repository whenever possible. Screenshots may be used for dashboards, but the source page or app path must be documented.

## AI-Use Expectations

AI may help improve slide wording, draft speaker notes, or identify jargon. Students must verify all numbers, citations, screenshots, and claims. The `ai-use-note.md` should state what AI helped with and what was checked manually.

## Presentation Preparation Checklist

The bullets below unpack the five scored categories above; they are preparation checks, not additional scored rows.

- audience and KT purpose are clear

- question is focused
- data source and limitations are transparent
- main visual is readable on screen
- interpretation is accurate and non-causal unless justified
- privacy and stewardship risks are acknowledged
- narration fits the target length
- likely questions are anticipated in the deck or a backup slide

## **Definition Of Done**

The presentation is done when the narrated slides are committed, rendered, and submitted with the Final Portfolio, and play without relying on local-only files.

# Final Portfolio

 See a worked example

The [the Final Portfolio submission in the NHANES worked example](#) shows one illustrative version of this milestone. Use it for structure and depth — your project must be your own.

## Purpose

The final portfolio is the complete project record: report, code, data documentation, dashboard-style KT product, presentation materials, reproducibility evidence, and AI-use audit. It should let another person understand what was done, why it matters, and how to rerun or inspect the work.

## Due Timing

Submit by the final portfolio deadline posted in Canvas. The syllabus deadline is the course final submission deadline unless Canvas states a more specific time.

## Points and Score Ownership

The Final Portfolio is worth **35 course points** and receives one group score. The existing portfolio and reproducibility checklists guide the direct 0–35 score; no contribution multiplier, complicated adjustment formula, or performance-band conversion is used.

The 35 points are allocated across five broad categories:

| Criterion                                                             | Points    |
|-----------------------------------------------------------------------|-----------|
| Final report and evidence                                             | 10        |
| Reproducibility and traceability                                      | 10        |
| KT product and presentation alignment                                 | 5         |
| Provenance, stewardship, and limitations                              | 5         |
| README, repository access, AI audit, risk list, and submission record | 5         |
| <b>Total</b>                                                          | <b>35</b> |

These categories are broad judgments, not containers for dozens of tiny deductions. The portfolio rewards coherent evidence and a workflow another person can understand and inspect. It does not reward statistical sophistication, public deployment, Shiny, publication-quality prose, or decorative polish. One underlying problem is assessed under the category that most directly owns it unless it causes genuinely separate required failures.

The **narrated presentation deck** is a required part of the portfolio and is scored separately as the **Presentation** component (10 points); see [Presentation](#). It is submitted online with the portfolio on Dec 11 — there is no in-class presentation.

## Required Deliverables

Create or update:

```
1 milestones/final-portfolio/submission/
```

Include these files:

- `final-portfolio-checklist.md`
- `repository-access-note.md`

- `final-risk-list.md`
- `ai-use-note.md`

Your project repository should also include:

- final report source and rendered output
- dashboard-style KT product or static preview
- data provenance and stewardship documentation
- reproducibility instructions
- narrated presentation slide deck (5-7 slides, recorded narration)
- clear README for the whole project

A live dashboard is not required unless the assignment says so. A static preview, screenshot, or rendered HTML page is acceptable when it clearly shows the KT product and passes the privacy check.

The three KT-product options, from heaviest to lightest:

- **Live dashboard** — an interactive page or app a reviewer can open and click (for example, a demo-only Shiny app). Never required.
- **Static preview** — a rendered, non-interactive HTML page committed to the repository that shows the KT product's current state.
- **Screenshot** — an image file of the product committed to the repository, with a note saying which file or app it shows.

Any of the three is acceptable for the final portfolio.

## Repository Access Note

Keep `repository-access-note.md` short. It must state:

- repository URL and whether it is public or private;
- latest submitted commit hash;
- confirmation that course staff can open the repository;
- if private, who granted staff access and the date access was checked;

- if public, confirmation that the safe-publishing check was completed;
- relative paths to the final report and dashboard/KT product.

This note records access facts only; it is not a new reflection or project summary.

Revising earlier work is allowed and encouraged: the final portfolio is the right place to apply M4 feedback and improve assignments or milestone files you already submitted. Only the files listed above are prescribed. Beyond them, organize the repository however serves your project best, as long as the README explains the structure.

## What To Submit

Submit your GitHub repository link on Canvas after committing and syncing all final materials. In the Canvas comment box, include:

- latest commit hash
- final report path
- dashboard or KT product path
- final portfolio checklist path
- rendered public link if the repository is public and safe to publish

## Public And Private Repository Guidance

Use a public repository only when the data and outputs are allowed to be public. Do not publish restricted, private, identifiable, or culturally sensitive data. If the repository must stay private, keep it private, grant course staff access, and submit the private repository link on Canvas.

Public repositories should avoid raw restricted data and should use documented public data, simulated data, or aggregate outputs that are safe to share. Private repositories still require clear documentation and reproducibility instructions for course



staff. See [Repository Sharing Guidance](#) before changing repository visibility.

## Reproducibility Checklist

Use the [Final Portfolio Checklist](#) to confirm:

- repository opens from a fresh clone or Codespace
- README explains the project, audience, and file structure
- all paths are relative
- required packages are documented
- report renders from source
- every rendered artifact explicitly required for the portfolio is committed with its source
- dashboard or KT product can be opened or rerun, or its static preview is committed
- generated figures and tables can be traced to code
- data provenance and access rules are documented
- privacy and stewardship risks are addressed
- AI use is disclosed and audited
- final commit is synced before Canvas submission

## What A Strong Project README Looks Like

Weak: “Final project for SPPH 381H. See files.”

Strong:

```
1 # Income and Blood Pressure in NHANES Adults
2
3 Audience: community health planners (non-technical).
4 Question: how does average systolic blood pressure differ across income groups?
5
6 - report/      - final report source (report.qmd) and rendered report.html
7 - dashboard/  - KT product static preview (preview.html) and screenshot
8 - data/       - documented access script; no raw restricted data committed
9 - milestones/ - milestone submissions, including the final portfolio checklist
10
```

```
11 To rerun: open the repo in Codespaces, then render report/report.qmd.  
12 Required packages are listed in reproducibility-note.md.
```

The strong version answers the four reader questions: what the project is, who it is for, where each artifact lives, and how to rerun it.

## AI-Use Expectations

AI may help polish writing, check code, create revision plans, or test explanations. Students must verify all outputs, citations, code, numeric results, and interpretations. The final `ai-use-note.md` should describe major AI uses across the project and the checks used to protect correctness.

## Grading Checklist

The final portfolio is ready for assessment when:

- report is complete and evidence-based
- dashboard-style KT product matches the intended audience
- data provenance and limitations are transparent
- repository is organized and reproducible
- final outputs are committed and synced
- narrated presentation deck aligns with the final project
- AI-use audit is complete
- repository access route is clear

## Definition Of Done

The final portfolio is done when course staff can open the submitted repository, locate the final artifacts, rerun or inspect the work, and understand the project without asking for missing files.

# Final Portfolio Checklist

Use this checklist before submitting the final repository link on Canvas.

## Repository Access

- ☐ Repository link opens for course staff.
- ☐ Private repository access has been granted to course staff when needed.
- ☐ Latest commit is synced to GitHub.
- ☐ Canvas submission includes the latest commit hash.
- ☐ `repository-access-note.md` records visibility, staff-access status, the checked date, latest commit hash, and final artifact paths.

## Project README

- ☐ README names the project audience and purpose.
- ☐ README explains the folder structure.
- ☐ README explains how to render the report.
- ☐ README explains how to open or run the KT product.
- ☐ README names any required packages or environment setup.

## Data And Stewardship

- ☐ Data source is cited.
- ☐ Access method is documented.
- ☐ Data restrictions are stated.
- ☐ Privacy and stewardship risks are acknowledged.

- ☐ Public repository contains only shareable data and outputs.

## **Reproducibility**

- ☐ Paths are relative.
- ☐ Report renders from source.
- ☐ Figures and tables are generated by code.
- ☐ Dashboard-style product opens or runs, or a static preview (screenshot or rendered HTML page) is committed in its place.
- ☐ Dependency notes are current.
- ☐ Known failures are recorded in the risk list.

## **Communication**

- ☐ Final report states the question, data, methods, findings, and limitations.
- ☐ KT product is matched to the intended audience.
- ☐ Presentation materials align with the final findings.
- ☐ Claims are descriptive, predictive, or causal only when justified.

## **AI Use**

- ☐ AI-use note lists major AI-assisted tasks.
- ☐ Numeric outputs and citations were verified manually.
- ☐ AI-generated code was tested.
- ☐ AI-generated prose was reviewed for accuracy and tone.

## **Definition Of Done**

- ☐ Final repository link submitted on Canvas.
- ☐ Commit hash included in Canvas.
- ☐ Final report path included in Canvas.

☐ Dashboard or KT product path included in Canvas.

# Repository Sharing Guidance

Use this guidance when deciding whether the final project repository can be public.

## Public Repository

A public repository is appropriate when:

- data are public and permitted for redistribution, or raw data are not committed
- outputs do not reveal private, identifiable, restricted, or culturally sensitive information
- data source terms allow public sharing
- the README explains where data came from and how to rerun the project

Public repositories are useful for portfolios, but publication is never more important than privacy, stewardship, or data-use rules.

## Private Repository

Use a private repository when:

- the data source is restricted
- the dataset includes sensitive or potentially identifiable records
- a data steward has not approved public sharing
- the group is unsure whether public release is allowed

For a private repository, grant course staff access and submit the private GitHub repository link on Canvas. The project still needs reproducibility instructions for course staff.

Record the chosen route in `repository-access-note.md` using the field list in the [Final Portfolio brief](#). The note lets the grader verify the submitted link and artifact paths without guessing; it does not require a justification for keeping the repository private.

## Safe Publishing Check

Before making a repository public, confirm:

- no restricted raw data are committed
- no credentials, tokens, or private URLs are committed
- no individual-level sensitive examples appear in outputs
- figures and tables are aggregate or otherwise safe to share
- README states the data source and sharing limits

## **Worked Example (NHANES)**



## Worked Example (NHANES)

## Worked Example (NHANES)

 Illustrative example — do not copy

This is **one** worked example on the class NHANES dataset, written by a *fictional* group, to show the expected **structure and depth** of a good submission. Your project must use **your own** question, data, audience, and analysis — do not copy this text or these files. This example lives in the public book for learning; graders assess your original work.

To make the milestone briefs concrete, this section shows **one complete worked project** — the fictional *Cedar Equity Lab* group studying income and BMI patterns in the class NHANES dataset — developed all the way from group formation to the final portfolio.

Read it for the **shape and quality** of good work: how a question is scoped, how a Table 1 and missingness summary are presented, how limitations are stated, how AI use is documented, and how a peer review is written. Then build **your own** project on **your own** question and data.

### The milestones

- [M0 — Group Formation](#)
- [M1 — Project Proposal](#)
- [M2 — Preliminary Analysis](#)
- [M3 — Project Update](#)
- [M4 — Peer Review](#)
- [Presentation — Narrated Deck](#)
- [Final Portfolio](#)

## **Example: M0 — Group Formation**

## Example: M0 — Group Formation

 Illustrative example — do not copy

This is **one** worked example on the class NHANES dataset, written by a *fictional* group, to show the expected **structure and depth** of a good submission. Your project must use **your own** question, data, audience, and analysis — do not copy this text or these files. This example lives in the public book for learning; graders assess your original work.

Below is the *Cedar Equity Lab* group's `m0-group-formation` submission from the worked NHANES example. See the [M0 brief](#) for what is required.

---

*File: `ai-use-note.md`*

### AI-use note

AI was used to suggest headings for the communication plan. The group read and edited each norm, then independently tested repository and Codespace access. The group will not outsource source verification, interpretation of results, privacy decisions, or final approval of submitted claims to AI.

---

*File: `codespace-check.md`*

## Codespace check

- Tested by: Maya Chen, Jordan Singh, and Alex Martin
  - Test: opened the group Codespace, ran `quarto check`, and confirmed that R, Quarto, and the tidyverse were available.
  - Result: passed for all three members.
  - Setup issue and plan: no current issue; package or path problems will be recorded in the README before the next milestone.
- 

*File: communication-plan.md*

## Communication plan

- Primary channel: the group's private course chat.
  - Response window: within 24 hours on weekdays and 48 hours on weekends.
  - Weekly check-in: 20 minutes after Tuesday class.
  - Decision record: one member adds dated decisions to the repository README or the relevant milestone note; another member confirms the change in the pull request.
- 

*File: group-formation.md*

## Group formation

- Group name: Cedar Equity Lab
- Student names: Maya Chen, Jordan Singh, Alex Martin
- GitHub usernames: `mayac-hds`, `jsingh-data`, `alexm-publichealth`
- Project area: Descriptive income and BMI patterns in the prepared NHANES classroom dataset.

- Primary repository owner: Maya Chen (`mayac-hds`)
- 

*File: `repo-access-evidence.md`*

## Repository access evidence

- Shared repository: `https://github.com/example-course/cedar-nhanes-equity`
- All three group members confirmed that they could open the repository.
- All three group members confirmed that they could open it in Codespaces.

The URL above is a representative, non-live value in this static instructor sample. A real M0 submission must contain a working group repository URL.

## **Example: M1 — Project Proposal**

# Example: M1 — Project Proposal

 Illustrative example — do not copy

This is **one** worked example on the class NHANES dataset, written by a *fictional* group, to show the expected **structure and depth** of a good submission. Your project must use **your own** question, data, audience, and analysis — do not copy this text or these files. This example lives in the public book for learning; graders assess your original work.

Below is the *Cedar Equity Lab* group's `m1-proposal` submission from the worked NHANES example. See the [M1 brief](#) for what is required.

---

*File: `ai-use-note.md`*

## AI-use note

AI helped shorten the audience and KT-purpose paragraph. The group checked the dataset dimensions in R, verified the source and income-group derivation in the course files, and removed language that implied causality or national representation. The questions and feasibility decision were approved by all three group members.

---

*File: `data-intake-card.md`*



## Data intake card

- **Dataset:** NHANES Health Equity classroom dataset
- **Source and steward:** CDC/National Center for Health Statistics, National Health and Nutrition Examination Survey
- **Access:** Prepared CSV committed at `data/nhanes_equity_v6.csv`; source documentation at <https://wwwn.cdc.gov/nchs/nhanes/>; checked 2026-10-01
- **Terms:** Public-use NHANES files; cite CDC/NCHS documentation
- **Geography and time:** United States; cycles represented from 1999-2000 through 2017-2018 and 2021-2023
- **Format and level:** CSV; row-level public-use-derived classroom records
- **Unit of analysis:** One examination participant record within a survey cycle
- **Key variables:** Age in years; BMI in  $\text{kg}/\text{m}^2$ ; IncomeGroup derived from the family income-to-poverty ratio; Cycle as the NHANES cycle label
- **Privacy/security:** Use only aggregate outputs, avoid small-cell claims, and do not upload row-level records to external AI tools
- **Stewardship:** Describe group comparisons without stigma and keep the prepared, unweighted classroom boundary beside results
- **Indigenous Data Sovereignty relevance:** No specific Indigenous community, service, territory, or identifier is part of the proposed comparison. We will reassess CARE/OCAP relevance if the project scope changes.
- **Known limitations:** Population inference requires the correct NHANES survey design; the prepared file is not a causal dataset
- **KT framing:** The data can identify a descriptive pattern worth discussing while teaching the audience what the comparison cannot establish
- **Citation:** CDC/NCHS, National Health and Nutrition Examination Survey, <https://wwwn.cdc.gov/nchs/nhanes/>, accessed 2026-10-01

AI could misstate the IncomeGroup cut-points or describe the

group as an individual income measure. We checked the variable creation in the course case-study build script and will describe it as a derived category.

---

*File: `initial-visual-plan.md`*

## Initial visual plan

- Planned figure: a line plot of unweighted mean BMI across NHANES cycles, with one line for each income group and visible cycle and BMI labels.
- Audience purpose: help briefing staff see whether the small overall income-group difference is similar across cycles or concentrated in a few periods.
- Misinterpretation risk: connected lines may be read as a causal trend or a national estimate. The title and caption will say that the summary is descriptive, unweighted, and based on the prepared classroom records.

---

*File: `proposal.md`*

## Project proposal

### Project and team

**Title:** Income and BMI Patterns in NHANES Adults

**Group:** Cedar Equity Lab — Maya Chen, Jordan Singh, and Alex Martin

## Audience and KT purpose

Our audience is municipal public-health staff preparing an internal health-equity briefing. The project will give them a simple descriptive view of income-group differences in the prepared dataset, while keeping the unweighted and non-causal limits visible. It is intended to support a discussion about what deserves closer survey-aware analysis, not an immediate policy decision.

## Questions

**Primary question:** How does mean BMI differ across income groups among adults age 20-80 in the prepared NHANES Health Equity classroom dataset?

**Secondary question:** How does mean BMI vary across available NHANES cycles for one selected income group?

## Data and planned output

We will use the cached CSV derived from CDC/NCHS NHANES public-use files. The planned outputs are a Table 1-style summary, one cycle-by-income figure, a short Quarto report, and a small Quarto dashboard-style page with one editable income group.

## Known limitations

The cached file is a prepared classroom dataset. The proposed summaries are unweighted, will exclude some missing values after reporting them, and cannot support causal or national-population claims. A full NHANES analysis would require survey weights, strata, and primary sampling units.

---

*File: reproducibility-plan.md*

## Reproducibility plan

Expected folders are `data/`, `report/`, `dashboard/`, and `milestones/`. The cached public-use-derived CSV will be committed once under `data/` and treated as read-only. R source files will load `tidyverse` and `knitr` explicitly and use paths relative to the project root.

Another student should be able to open the course Codespace, start at the repository root, and run `quarto render` on the named source. Generated tables, figures, and HTML files will be regenerated from code rather than edited by hand. Any added dependency will be recorded before use.

Feasibility check: the CSV loaded successfully in the course environment with 105,626 rows and 18 columns. R is the project language; a Python version is not needed for the proposed questions.

## **Example: M2 — Preliminary Analysis**

## Example: M2 — Preliminary Analysis

 Illustrative example — do not copy

This is **one** worked example on the class NHANES dataset, written by a *fictional* group, to show the expected **structure and depth** of a good submission. Your project must use **your own** question, data, audience, and analysis — do not copy this text or these files. This example lives in the public book for learning; graders assess your original work.

Below is the *Cedar Equity Lab* group's `m2-preliminary-analysis` submission from the worked NHANES example. See the [M2 brief](#) for what is required.

---

*File: `ai-use-note.md`*

### AI-use note

AI suggested a tidyverse outline for the grouped summary and a shorter first draft of the interpretation. We independently checked:

1. the prepared-file and adult-cohort row counts;
2. missingness before complete-case exclusions;
3. the income-group labels and the N, means, and standard deviations in the CSV export;
4. that the rendered figure matches the grouped data; and

5. that the final prose says descriptive and unweighted and makes no causal or national-population claim.

---

*File: `methods-note.md`*

## Methods note

We used the prepared NHANES Health Equity classroom CSV and restricted the cohort to ages 20-80, inclusive. We reported missingness for the key variables before selecting complete records for age, BMI, and income group. The grouped summary reports N, BMI mean (SD), and age mean (SD) by `IncomeGroup`; the figure shows unweighted mean BMI by income group and cycle. These results describe the prepared classroom records and cannot support causal or national-population claims.

The preliminary analysis builds on Jordan's A6 `eda-note.qmd`; the group rechecked the cohort, denominators, exports, and wording before reuse.

---

*File: `missingness-summary.csv` — [view the data table](#)*

---

*File: `preliminary-analysis.html` — [view the rendered output](#)*

---

*File: `table1.csv` — [view the data table](#)*

## **Example: M3 — Project Update**



# Example: M3 — Project Update

 Illustrative example — do not copy

This is **one** worked example on the class NHANES dataset, written by a *fictional* group, to show the expected **structure and depth** of a good submission. Your project must use **your own** question, data, audience, and analysis — do not copy this text or these files. This example lives in the public book for learning; graders assess your original work.

Below is the *Cedar Equity Lab* group's `m3-project-update` submission from the worked NHANES example. See the [M3 brief](#) for what is required.

---

*File: `ai-use-note.md`*

## AI-use note

AI helped propose a shorter ordering for the draft report and flagged two sentences as potentially causal. The group checked every displayed number against the rendered table, confirmed the CDC/NCHS citation in `references.bib`, reran the report and dashboard, and rewrote the interpretation to say descriptive, unweighted, and modest.

---

*File: `dashboard-preview.md`*

## Dashboard preview

- Current source: `dashboard/dashboard.qmd`
  - Rendered preview: `dashboard/dashboard.html`
  - Intended audience: municipal public-health staff preparing an internal health-equity briefing
  - Feature: change the single `selected_income` value and re-render to update the table, title, and cycle plot
  - Limitation/privacy caution: the page shows aggregate, unweighted classroom summaries only; it does not expose row-level data or support causal or national-population claims
- 

*File: `feedback-request.md`*

## Feedback request

Please focus on these questions:

- **Correctness:** Do the complete-case counts in the report agree with the table and figure?
  - **Interpretation:** Is the difference described as modest and descriptive, without implying causality or national representation?
  - **Visual clarity:** Does the cycle figure make the small group difference understandable without exaggerating it?
  - **Reproducibility:** Are the root render commands and data path clear enough to follow in a fresh Codespace?
  - **Audience fit:** Would a municipal public-health briefing team understand what question this result should prompt next?
- 

*File: `project-update.md`*

## Project update

- **Current title:** Income and BMI Patterns in NHANES Adults
- **Audience:** Municipal public-health staff preparing an internal health-equity briefing
- **Refined primary question:** How does mean BMI differ across income groups among adults age 20-80 in the prepared classroom dataset?
- **Data and sample:** Cached CDC/NCHS NHANES-derived classroom CSV; 57,137 adult rows before missing BMI and income-group exclusions
- **Stable finding:** Mean BMI is about 28.6 in the high-income group, 29.4 in the low-income group, and 29.3 in the middle-income group in the complete adult records.
- **Choice needing review:** Does the cycle-by-income figure help the audience understand the small difference without making it look more important than it is?
- **Next steps:** Apply peer feedback, align the dashboard wording with the report, complete a fresh-environment render, and turn the story into M5 slides.

---

*File: `report-draft.html` — [view the rendered output](#)*

---

*File: `reproducibility-note.md`*

## Reproducibility note

From the project root in the course Codespace, run `quarto render milestones/m3-project-update/submission/report-draft.qmd` and `quarto render dashboard/dashboard.qmd`. Both sources load the committed CSV at `data/nhanes_equity_v6.csv` using a relative path and require only `tidyverse` and `knitr`. The current renders completed without a manual data or path step.

## **Example: M4 — Peer Review**

## Example: M4 — Peer Review

 Illustrative example — do not copy

This is **one** worked example on the class NHANES dataset, written by a *fictional* group, to show the expected **structure and depth** of a good submission. Your project must use **your own** question, data, audience, and analysis — do not copy this text or these files. This example lives in the public book for learning; graders assess your original work.

Below is the *Cedar Equity Lab* group's `m4-peer-review` submission from the worked NHANES example. See the [M4 brief](#) for what is required.

---

*File: `ai-use-note.md`*

### AI-use note

AI was not used to inspect the peer files or generate evidence. After the group completed the render and file checks, AI was used once to suggest a shorter heading order for the review. We compared the reorganized note with the files we actually opened and removed one generic suggestion that was not supported by the snapshot.

---

*File: `auditability-check.md`*

## Auditability check

- **Data provenance:** Partly clear. The project names a fictional community health survey extract but gives no steward, version, access route, or variable definition.
  - **AI disclosure:** Present and specific to shortening the limitation. The note says the group checked the causal wording.
  - **Claim support:** Not yet sufficient. The 62% and 54% values cannot be traced to code or data in the snapshot.
  - **Limitations:** Visible and appropriately descriptive. The denominator and missing-response rule still need to be stated beside the result.
- 

*File: kt-feedback.md*

## KT feedback

- **Audience fit:** The one-page comparison card is a reasonable format for city planning staff, and the near/far wording is easier to understand than a technical distance variable name.
  - **Plain language:** The main sentence is clear. Replace “reported good or very good health more often” with the two percentages beside it so the size of the difference stays visible.
  - **Visual clarity:** We could not assess the actual page because there is no screenshot or render. A plain static preview would be enough for review.
  - **Actionability:** End with a modest next step, such as checking age and mobility patterns before using the comparison in planning discussion.
  - **Privacy:** Keep the page aggregate. Do not display respondent rows, exact addresses, or small neighbourhood cells in the preview.
-

*File: peer-review.md*

## Peer review

**Reviewed project:** Walking Access and Self-Rated Health, Northside Active Living Team

### Two strengths

1. The question and audience are focused. A city planning team can understand why the near/far comparison might be useful for a first discussion.
2. The draft does not claim that walking-route access causes better health. It names age, mobility, neighbourhood conditions, and missing responses as possible limitations.

### Three prioritized revision suggestions

1. **First, make the result reproducible.** Add the classroom data access path and the code that produces 62% and 54%. The current report renders, but the table is manually typed, so we could not check the denominators or missing responses.
2. **Add an openable KT preview.** A plain screenshot or rendered static page is enough. The current text describes a selector, but a reviewer cannot see the proposed labels, visual hierarchy, or limitation placement.
3. **Clarify the denominator beside each percentage.** State whether missing self-rated-health responses were excluded and whether 430 and 382 are the denominators used for 62% and 54%. This will make the result easier to audit and harder to overread.

### Question for the group

Will the final comparison keep the two distance categories supplied in the classroom extract, or will your team create a new

cut-point? If it changes, please document why before updating the percentages.

## Blocking reproducibility risk

The absent data access step and calculation code would block final reproducibility. A reviewer can render the prose-only QMD but cannot regenerate or verify the central result.

## Reviewer reflection

- Reviewing this project reminded us that a document can render successfully while its analysis is still not reproducible.
- Missing self-rated-health responses and differences in age or mobility need the most attention because they could change how the near/far comparison is interpreted.
- For our own project, we will open the README first, run the stated command, and trace the main number to code before commenting on presentation quality.

---

*File: `reproducibility-check.md`*

## Reproducibility check

- **Report render:** Passed with `quarto render report-draft.qmd` from the fictional M3 snapshot folder.
- **Paths:** No absolute path appears in the QMD, but there is also no data path or analysis script to test.
- **Data access:** Not clear. The source is described only as a fictional classroom survey extract and no file/access route is supplied.
- **Figures and tables from code:** No. The central table is typed in Markdown and the draft contains no code that calculates the percentages.



- **KT product:** Could not be opened. `dashboard-preview.md` describes the planned page but does not point to a screenshot, render, or source.
- **Stopping point:** We completed the report render, then stopped the analytic rerun because the data and calculation code were unavailable. The project owner needs to add both and document the root run command.

## **Example: Presentation — Narrated Deck**

# Example: Presentation — Narrated Deck

 Illustrative example — do not copy

This is **one** worked example on the class NHANES dataset, written by a *fictional* group, to show the expected **structure and depth** of a good submission. Your project must use **your own** question, data, audience, and analysis — do not copy this text or these files. This example lives in the public book for learning; graders assess your original work.

Below is the *Cedar Equity Lab* group's **m5-presentation** submission from the worked NHANES example. See the [Presentation brief](#) for what is required.

---

*File: ai-use-note.md*

## AI-use note

AI suggested shorter slide titles and identified jargon in the first draft. The group checked the 57,137 cohort count, missingness values, three displayed means, figure labels, and source note against the final report and rendered slides. We rejected a suggested phrase that implied income predicted BMI.

---

*File: peer-feedback-received.md*

## Peer feedback received on the NHANES project

This is inward-facing feedback on our NHANES project. It is separate from the outward-facing M4 review we wrote about the Northside Active Living project.

- **Source:** River Health Team’s review of our M3 report and KT preview.
- **Feedback received:** Say explicitly that the overall difference is small; use “Average BMI (kg/m<sup>2</sup>)” rather than “Mean BMI” in the audience-facing dashboard; and keep the unweighted limitation beside the figure.
- **Changes made:** We added “modest/small” to the report and slides, expanded the dashboard label and units, and placed the descriptive/unweighted caution in the dashboard figure caption and limitation section.
- **Suggestion not adopted:** The reviewers suggested a second dashboard control for age range. We kept one income-group control because the current audience question is answered without added complexity.

---

*File: presentation-checklist.md*

## Presentation checklist

- ☒ Six-slide story fits a seven-minute practice run.
- ☒ Audience, question, and KT purpose are visible near the start.
- ☒ Data source, adult cohort, and missingness are stated.
- ☒ Main figure title, units, source boundary, and limitation are readable.
- ☒ Interpretation remains descriptive and non-causal.
- ☒ Privacy and stewardship caution is included in the source note.
- ☒ Slides render from the project root without local-only files.

- ☒ Final slide gives a practical next analytic question.
- ☒ Backup notes include source, cohort, and AI-use details.

---

*File: `slides.html` — [view the rendered output](#)*

---

*File: `source-and-limitations-note.md`*

## Source and limitations note

- **Source:** Prepared classroom snapshot derived from CDC/NCHS National Health and Nutrition Examination Survey public-use files; <https://wwwn.cdc.gov/nchs/nhanes/>
- **Inclusion/exclusion:** Ages 20-80 inclusive; grouped results use non-missing age, BMI, and income group; cycle figure also requires a non-missing cycle
- **Claim type:** Descriptive and unweighted, not predictive or causal
- **Privacy/stewardship caution:** Show aggregate results only, avoid stigmatizing income groups, and do not upload row-level records to AI tools
- **Limitation to say aloud:** The analysis does not use the NHANES survey design and is not a national estimate

## **Example: Final Portfolio**

# Example: Final Portfolio

 Illustrative example — do not copy

This is **one** worked example on the class NHANES dataset, written by a *fictional* group, to show the expected **structure and depth** of a good submission. Your project must use **your own** question, data, audience, and analysis — do not copy this text or these files. This example lives in the public book for learning; graders assess your original work.

Below is the *Cedar Equity Lab* group's `final-portfolio` submission from the worked NHANES example. See the [Final Portfolio brief](#) for what is required.

---

*File: `ai-use-note.md`*

## Final project AI-use audit

### Major uses

- Shortened the audience and KT-purpose language in M1.
- Suggested an initial tidyverse outline for the M2 grouped summary.
- Helped reorder the M3 report and identify potentially causal wording.
- Suggested shorter slide titles and flagged jargon before M5.
- Helped organize, but not perform, the outward-facing M4 review.

## Checks completed by the group

- Recomputed the adult cohort, missingness counts, complete-case Ns, means, and standard deviations from the committed CSV.
- Confirmed that report, dashboard, and slide figures come from code and agree with the displayed results.
- Checked the CDC/NCHS source citation and the prepared-classroom provenance statement.
- Tested all three final render commands from the project root.
- Reviewed AI-assisted prose for causal, population-level, stigmatizing, or unsupported language.
- Did not upload row-level data to any AI tool.

The group, not the AI tool, approved every submitted number, citation, code chunk, interpretation, and sharing decision.

---

*File: `final-portfolio-checklist.md`*

## Final Portfolio Checklist

### Repository access

- ☒ Repository route is recorded for course staff.
- ☒ Representative latest commit hash is recorded.
- ☒ Final artifact paths are listed in `repository-access-note.md`.
- ☒ Safe-publishing check is documented.

### Project README

- ☒ README names the audience, purpose, questions, and main finding.
- ☒ README explains the folder structure.
- ☒ README gives the three core render commands.
- ☒ README names the required R packages and data boundary.



## Data and stewardship

- ☒ CDC/NCHS NHANES is cited.
- ☒ The cached access method and classroom-use boundary are documented.
- ☒ Privacy, aggregate-output, and stewardship cautions are stated.
- ☒ The model contains no credentials or private data.

## Reproducibility

- ☒ Paths are relative to the project root.
- ☒ Final report renders from source.
- ☒ Final dashboard-style page renders from source.
- ☒ Final slides render from source.
- ☒ Tables and figures are generated by code.
- ☒ Dependencies are limited to `tidyverse` and `knitr` in the supported course environment.
- ☒ Known analytic and communication risks are recorded.

## Communication and AI use

- ☒ Report states the question, data, methods, findings, and limitations.
- ☒ KT page and slides use the same audience and claim boundary.
- ☒ Main findings agree across the report, KT page, and slides.
- ☒ Major AI-assisted tasks and manual checks are recorded.

## Submission facts

- ☒ Representative repository link and commit hash are included.
- ☒ Final report path is included.
- ☒ Dashboard/KT product path is included.
- ☒ Final checklist path is included.

---

*File: final-risk-list.md*

## Final risk list

No known render or repository-organization blocker remains after the final test. The remaining risks are interpretation limits rather than hidden failures:

- The prepared classroom analysis is unweighted and is not a national NHANES estimate.
- Complete-case grouped results exclude records with missing BMI or income group; missingness is reported but not modelled.
- Income-group comparisons can be stigmatizing if the modest descriptive difference is presented without context.
- A future change to the cached data would require re-rendering the report, dashboard, and slides together.

---

*File: repository-access-note.md*

## Repository access note

- Repository URL: `https://github.com/example-course/cedar-nhanes-equity`
- Visibility: private
- Latest submitted commit: `a3f92bc`
- Staff access: representative course-staff access was granted by Maya Chen and checked on 2026-12-10
- Final report: `report/report.qmd` and `report/report.html`
- Dashboard/KT product: `dashboard/dashboard.qmd` and `dashboard/dashboard.html`
- Presentation: `milestones/m5-presentation/submission/slides.qmd` and `slides.html`

The URL, hash, and access date are clearly representative values in this static instructor sample. A real student submission must provide live, verifiable facts.

---

*Project report:* `report.html` — [view it](#)

---

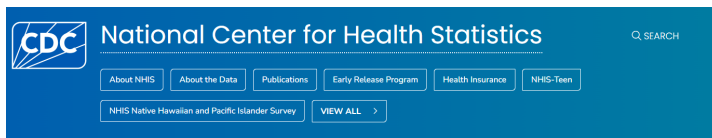
*Project dashboard:* `dashboard.html` — [view it](#)

*Optional interactive version — Shiny app (demonstration only):* the course includes a runnable **Shiny** dashboard built on the same NHANES data. A live app needs a server, so it runs in a GitHub Codespace or locally — not on this static site. View or download the source to try it yourself: [examples/nhanes-equity/app/app.R](#) (on GitHub). A live app is **optional**; a static dashboard like the one above is the required deliverable.

## **Reference: Data Sources**

# NHIS

The National Health Interview Survey (NHIS) is a cross-sectional household interview survey conducted by the US Centers for Disease Control (CDC). The survey collects information on the health status, health care access, and health behaviors of the US population.



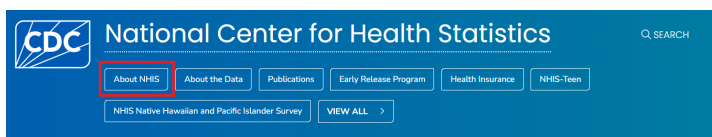
## National Health Interview Survey



[CDC website for NHIS](#)

## About NHIS

You can find more information about the NHIS survey by clicking on [About NHIS](#):



## About NHIS

For Everyone  
NOVEMBER 20, 2024

On the **About NHIS** page, you will find information about what the survey collects, data and documentation, the instructions on how to use the data, and information on related NHIS surveys.

### Overview

The National Health Interview Survey (NHIS) monitors the health of people across the country—the U.S. population—by collecting and analyzing data on a broad range of health topics. NHIS focuses on the health of U.S. children and adults.

CDC's [National Center for Health Statistics](#) (NCHS) conducts NHIS. It is the nation's largest and oldest national health survey. NHIS began collecting data in 1957.

Each year, NHIS collects data from about 27,000 adults through confidential, face-to-face interviews throughout the year. Many of these adults also provide health information about

#### ON THIS PAGE

##### Overview

[What's collected](#)

[Data and documentation](#)

[How the data are used](#)

[Related NHIS surveys](#)

You can click on each of the links or scroll down to find information on each of these topics one by one. For example, if we click on the **How the data are used** link, we will find information as follows:

### How the data are used

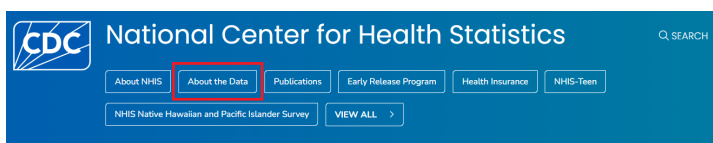
NCHS staff use NHIS data to produce official national health estimates. One of NHIS's key strengths is its ability to produce estimates for many subgroups of the U.S. population. NHIS can do this because it collects data from such a large group of people.

NHIS health statistics provide health information by—

- Demographic characteristics, like age and marital status
- Geographic location, like region and metro area size
- Social and economic circumstances, like employment status and poverty level

## About the Data

If you click on **About the Data** link, you will find information about the NHIS questionnaires, datasets, and documentation:



Under the **Overview** tab, you will find information about different NHIS cycles and the questionnaires, datasets, and documentation available in each cycle. As an example, let's click on the **2023 NHIS Questionnaires, Datasets, and Documentation** link to see the details for the 2023 NHIS:

[NHIS Data, Questionnaires and Related Documentation](#)

## Overview

Resources on this page provide information about the [National Health Interview Survey](#) (NHIS) data collection, its participants, and its protocols. It also provides analytic notes and references. They will help you understand, analyze, and use NHIS data.

Find survey instruments, data, and other resources by year at—

- [2025 NHIS Questionnaires, Datasets, and Documentation](#)
- [2024 NHIS Questionnaires, Datasets, and Documentation](#)
- [2023 NHIS Questionnaires, Datasets, and Documentation](#)
- [2022 NHIS Questionnaires, Datasets, and Documentation](#)
- [2021 NHIS Questionnaires, Datasets, and Documentation](#)
- [2020 NHIS Questionnaires, Datasets, and Documentation](#)
- [2019 NHIS Questionnaires, Datasets, and Documentation](#)

## ON THIS PAGE

### Overview

[Before 2019](#)

[Data linked to NHIS data](#)

[Other NHIS survey data](#)

[Related surveys](#)

[Related pages](#)

## NHIS Questionnaires, Datasets, and Documentation

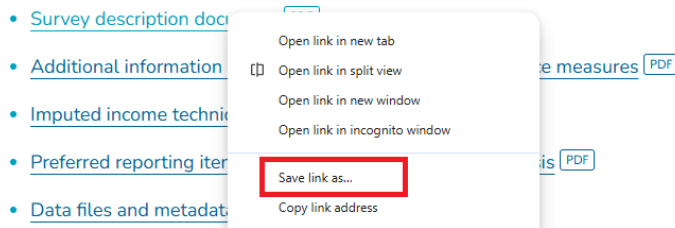
To view the description of the 2023 NHIS, click on the **Survey description document** link:

## Using our data

- [Survey description document](#) [PDF](#)

This will open the PDF of the 2023 NHIS survey description. Alternatively, you can right-click on the link, click on **Save link as...** and save the PDF file on your computer for future reference.

## Using our data



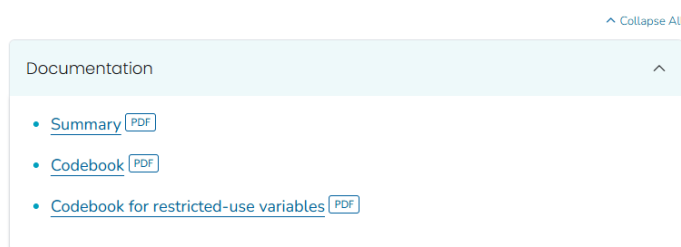
Similarly, you can open and save the questionnaires and related materials under the **Questionnaires and survey materials** tab.

In the recent NHIS, data are available in 5 categories: interview data for adults, interview data for children, data for parent-child pairs, imputed income, and paradata. You can click each link for documentation to see a summary of the data and the codebook.

Data are available in 5 categories:

- Sample adult interview
- Sample child interview
- Parent-child pair data
- Imputed income
- Paradata

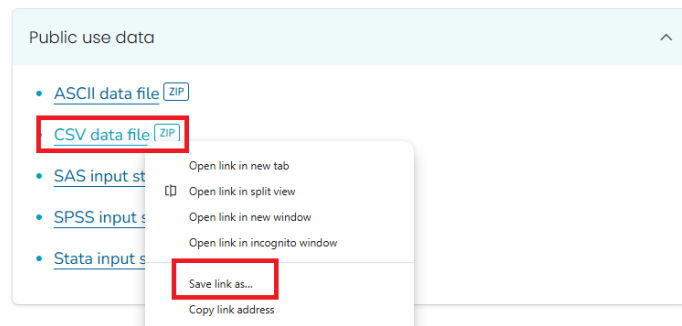
## Sample adult interview



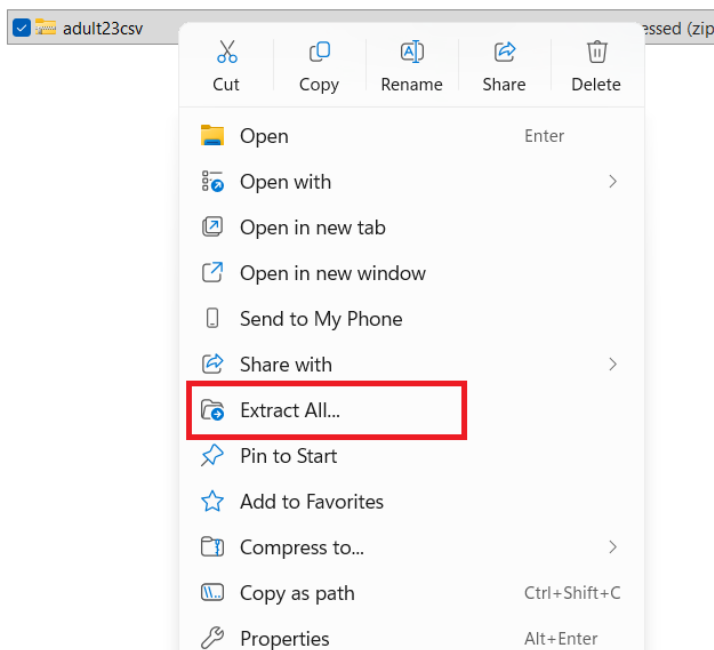
## NHIS Data

The data files are stored on the NHIS website in different formats. You can save the dataset in any statistical package that supports these file formats, e.g., ASCII, CSV, SAS. Let us save the `adult` dataset in the CSV format, which is in a zip file.





The next step is to unzip the file to extract the CSV. Depending on your computer, you can unzip the file with or without software. For example, if you are using Windows, you can right-click the zip file and select **Extract All...** to unzip it.



Once unzipped, you will find two files: the CSV file for the dataset and a readme file with some instructions. Below is the example for the `adult` dataset:

|                                                                                           |                       |           |
|-------------------------------------------------------------------------------------------|-----------------------|-----------|
|  adult23 | Microsoft Excel Co... | 28,738 KB |
|  readme  | Text Document         | 3 KB      |

To open the datafile in Excel, - you can double click on the CSV file, or - right click on the CSV file and click on **Open**.

Later, you will learn how to open the data file in different statistical software. For Excel, it would take a few seconds/minutes because of the large size of the dataset. Once you open the data file, you will see the data in a tabular format with rows and columns.

| URBRRL | RATCAT_A | INCTCFLG_A | IMPINCFLG_A | LANGSPECR_A | LANGSOC_A | LANGDOC_A | LANGMED_A | LANGHM_A | PPSU | PSTRAT |
|--------|----------|------------|-------------|-------------|-----------|-----------|-----------|----------|------|--------|
| 3      | 4        | 0          | 0           |             |           |           |           |          | 2    | 103    |
| 4      | 8        | 0          | 0           |             |           |           |           |          | 2    | 122    |
| 4      | 14       | 0          | 0           |             |           |           |           |          | 2    | 122    |
| 4      | 10       | 0          | 0           |             |           |           |           |          | 2    | 122    |
| 4      | 5        | 0          | 0           |             |           |           |           |          | 2    | 122    |
| 4      | 14       | 0          | 0           |             |           |           |           |          | 2    | 122    |
| 3      | 12       | 0          | 0           |             |           |           |           |          | 2    | 103    |

- Each row represents a person.
- Each column represents a variable.
- The first row contains the variable names. The remaining rows contain the data for each person.

## Use of the codebook

To understand the variables in the dataset, you can refer to the codebook. The codebook provides information about the variable names, labels, and values. For example, we want to understand the variable `SEX_A`:

| FQ      | FR    | FS     | FT    | FU       | FV       |
|---------|-------|--------|-------|----------|----------|
| SRVY_YR | SEX_A | AGEP_A | AGE65 | ASTATNEV | MENTHOLC |
| 2023    | 1     | 67     |       | 1        |          |
| 2023    | 1     | 73     |       | 1        |          |
| 2023    | 1     | 48     |       | 1        |          |
| 2023    | 2     | 42     |       | 1        |          |
| 2023    | 2     | 50     |       | 1        |          |
| 2023    | 2     | 46     |       | 1        |          |
| 2023    | 2     | 36     |       | 1        |          |
| 2023    | 1     | 44     |       | 1        |          |

We can look for `SEX_A` in the codebook, which will provide us with the variable label and the values for that variable (e.g., 1 for Male, 2 for Female, and so on).

17 / 653 | - 117% + | | | | | | |

---

2023 NATIONAL HEALTH INTERVIEW SURVEY (NHIS)  
Codebook for Sample Adult File (Version: 24 June 2024)  
PUBLIC USE

---

**Variable Information:**

Variable: **SEX\_A**

Module: Roster

Section: HHC : Household Composition

File(s): Adult

Data Type: Numeric

Length: 1

Question Text:

Description: Sex of Sample Adult

Recode: Yes

Universe: HHSTAT\_A=1

Universe Description: Sample adults 18+

Sources:

Question ID:

Keywords:

Notes:

Evaluation Report:

**Unweighted frequencies:**

| SEX_A | Sex of Sample Adult | Frequency | Percent |
|-------|---------------------|-----------|---------|
| 1     | Male                | 13457     | 45.58   |
| 2     | Female              | 16059     | 54.40   |
| 7     | Refused             | 4         | 0.01    |
| 8     | Not Ascertained     | 0         | 0.00    |
| 9     | Don't Know          | 2         | 0.01    |

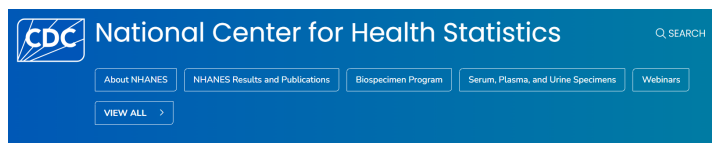
Frequency Missing:

There are so many variables in the dataset, and it is not possible to understand all of them at once. You can refer to the codebook to understand the variables that you are interested in for your analysis. For example, the `WTFA_A` variable is the sample adult weight variable used to calculate weighted estimates

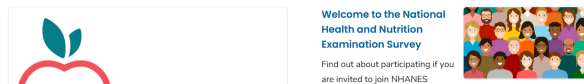
for the US adult population. Similarly, the `HYPEV_A` variable is the hypertension ever variable, indicating whether a doctor or other health professional has ever told the person they have hypertension. Again, you can refer to the codebook to understand the variables that you are interested in.

# NHANES

The National Health and Nutrition Examination Survey (NHANES) is a cross-sectional survey conducted by the US Centers for Disease Control (CDC). The survey collects information on the health and nutritional status of the US population. The survey includes interviews, physical examinations, and laboratory tests.



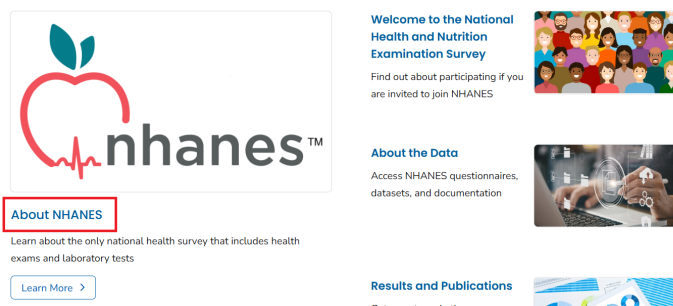
## National Health and Nutrition Examination Survey



[CDC website for NHANES](#)

## About NHANES

You can find more information about the NHANES survey by clicking on [About NHANES](#):



## About NHANES

For Everyone  
DECEMBER 18, 2024

### AT A GLANCE

- NHANES is the National Health and Nutrition Examination Survey.
- NHANES is a national survey that measures the health and nutrition of adults and children in the United States.
- NHANES is the only national health survey that includes health exams and laboratory tests for participants of all ages.



On the **About NHANES** page, you will find information about what the survey collects, data and documentation, the instructions on how to use the data, and information on related NHANES surveys.

## Overview

The National Health and Nutrition Examination Survey (NHANES) collects data about the health of adults and children in the United States. We also collect data about what participants eat, drink, and take as supplements to determine how many nutrients are in their diet. These dietary interviews and blood tests help us measure the nutritional status of U.S. adults and children.

CDC's [National Center for Health Statistics](#) (NCHS) conducts NHANES. NHANES is the only national health survey that includes health exams, laboratory tests, and dietary interviews for participants of all ages.

### ON THIS PAGE

- Overview
- What's collected
- Data and documentation
- How the data are used
- Program director
- Contact NHANES

You can click on each of the links or scroll down to find information on each of these topics one by one. For example, if we click on the **Data and documentation** link, we will find information as follows:

## Data and documentation

NCHS [publishes reports](#) featuring analysis of NHANES data. Selected estimates also are available as part of [interactive data visualizations](#) that provide tables and charts.

### Data files

[NHANES data files](#) and [related documentation](#) are publicly available to download and analyze. The related documentation can help researchers understand and use NHANES data. All potentially identifiable information has been removed to ensure the confidentiality of participants and their households.

[NHANES restricted data files](#) are available through the NCHS Research Data Center (RDC) for a fee. To access restricted NHANES data, researchers must submit requests to the

### RELATED PAGES

- What NHANES Covers and How It Works
- Who Participates In NHANES
- How People Use NHANES Data
- What NHANES Data Have Achieved
- Ethics Review Board Approval

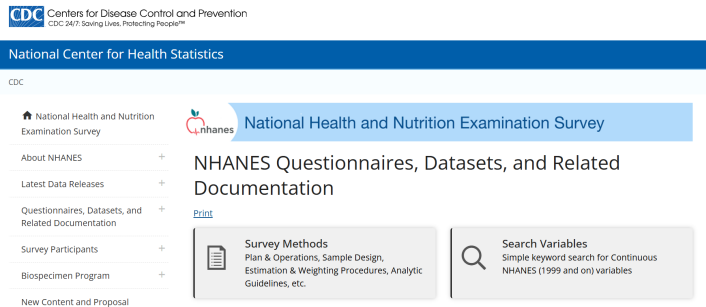
[VIEW ALL](#)  
National Health and Nutrition  
Examination Survey

## Survey Methods and Analytic Guidelines

If you click on **NHANES data files and related documentation** link, you will find information about the NHANES questionnaires, datasets, and documentation:

## Data files

[NHANES data files and related documentation](#) are publicly available to download and analyze. The related documentation can help researchers understand and use NHANES data. All potentially identifiable information has been removed to ensure the confidentiality of participants and their households.

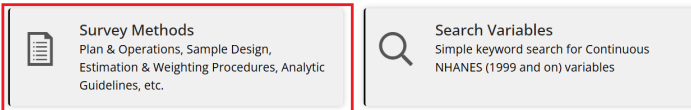


You can click **Survey Methods** to find information on survey methods and analytic guidelines. The **Survey Methods and Analytic Guidelines** page provides information about the plan and operation of the NHANES survey, the sampling design, methods for calculating weights and variance estimation, and guidelines for analyzing NHANES data, along with links to related resources.

[NHANES data files and related documentation](#)

## NHANES Questionnaires, Datasets, and Related Documentation

[Print](#)



## Search Variables

If you click on **Search Variables** link, you will find a search tool to search for variables in the NHANES datasets (1999 and on).

## NHANES Questionnaires, Datasets, and Related Documentation

[Print](#)

**Survey Methods**  
Plan & Operations, Sample Design,  
Estimation & Weighting Procedures, Analytic  
Guidelines, etc.

**Search Variables**  
Simple keyword search for Continuous  
NHANES (1999 and on) variables

You can search for variables by entering keywords in the search box. For example, if you enter **diabetes** in the search box, you will find a list of variables related to diabetes in the NHANES datasets.

### Search Variables

[Print](#)

Search for words and "quoted phrases" in selected fields (Data File Name, SAS Label, Variable Description, Variable Name) of Continuous NHANES (1999 and on) variables that have published documentation. *Multiply imputed Dual Energy X-Ray Absorptiometry variables are not included.*

Refer to the [Survey Content Brochure](#) for the full content of Continuous NHANES.

|                         |                                            |
|-------------------------|--------------------------------------------|
| Search Term             | <input type="text" value="diabetes"/>      |
| Fields to Search        | <input type="text" value="All"/>           |
| Sort By                 | <input type="text" value="Variable Name"/> |
| Limited Access          | <input type="text" value="Exclude"/>       |
| Release Cycle           | <input type="text" value="All"/>           |
| Search Result Page Size | <input type="text" value="50"/>            |

Search

Number of variables found: 459

| Variable Name | SAS Label                            | Variable Description                                                                                                                            | Data File Name   | Component Link                          |
|---------------|--------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|------------------|-----------------------------------------|
| DID040        | Age when first told you had diabetes | How old (was SP/were you) when a doctor or other health professional first told (you/him/her) that (you/he/she) had diabetes or sugar diabetes? | Diabetes (DIQ_D) | <a href="#">2005-2006 Questionnaire</a> |
| DID040        | Age when first told you had diabetes | How old (was SP/were you) when a doctor or other health professional first told (you/him/her) that (you/he/she) had diabetes or sugar diabetes? | Diabetes (DIQ_E) | <a href="#">2007-2008 Questionnaire</a> |
| DID040        | Age when first told you had diabetes | How old (was SP/were you) when a doctor or other health professional first told (you/him/her) that (you/he/she) had diabetes or sugar diabetes? | Diabetes (DIQ_F) | <a href="#">2009-2010 Questionnaire</a> |
| DID040        | Age when first told you had          | How old (was SP/were you) when a doctor or other health professional first told (you/him/her) that                                              | Diabetes (DIQ_G) | <a href="#">2011-2012 Questionnaire</a> |

You can restrict your search to a specific NHANES cycle by selecting the cycle from the drop-down menu. For example, if you select the **2017–2018** cycle, you will find a list of variables related to diabetes in the 2017-2018 NHANES dataset.



|                         |           |
|-------------------------|-----------|
| Limited Access          | Exclude   |
| Release Cycle           | 2017-2018 |
| Search Result Page Size | 50        |

Search

Number of variables found: 58

| Variable Name | SAS Label                                | Variable Description                                                                                    | Data File Name              | Component Link                          |
|---------------|------------------------------------------|---------------------------------------------------------------------------------------------------------|-----------------------------|-----------------------------------------|
| DIQ175C       | Age                                      | Why (Do you/Does SP) think (you are/he is/she is) at risk for diabetes or prediabetes? [Anything else?] | Diabetes (DIQ_J)            | <a href="#">2017-2018 Questionnaire</a> |
| RHQ163        | Age told you had diabetes while pregnant | How old (were you/was SP) when (you were/she was) first told (you/she) had diabetes during a pregnancy? | Reproductive Health (RHQ_J) | <a href="#">2017-2018 Questionnaire</a> |

## NHANES Questionnaires, Datasets, and Documentation

Let's explore the questionnaires, datasets, and documentation for the 2017-2018 NHANES cycle. You need to click on **NHANES 2017-2018** under the **Continuous NHANES** tab:

### NHANES Questionnaires, Datasets, and Related Documentation

[Print](#)



**Survey Methods**  
Plan & Operations, Sample Design, Estimation & Weighting Procedures, Analytic Guidelines, etc.



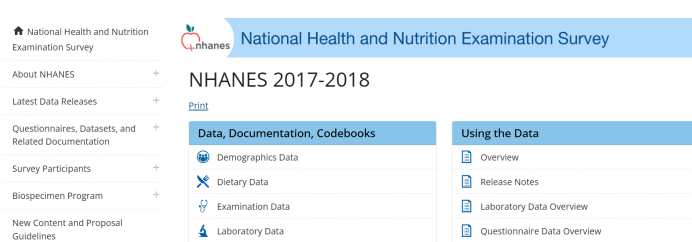
**Search Variables**  
Simple keyword search for Continuous NHANES (1999 and on) variables

[» Guidelines for High Quality Analyses of NHANES Data](#)

#### Continuous NHANES

|                  |                        |                                          |                  |
|------------------|------------------------|------------------------------------------|------------------|
| NHANES 2025-2026 | NHANES 08/2021-08/2023 | NHANES 2017-March 2020 Pre-Pandemic Data |                  |
| NHANES 2019-2020 | NHANES 2017-2018       | NHANES 2015-2016                         | NHANES 2013-2014 |

This will open the page for the 2017-2018 NHANES cycle, where you can find information on the survey description, questionnaires, datasets, and documentation.



There are six categories of data available for the 2017-2018 NHANES cycle: Demographics, Dietary, Examination, Laboratory, Questionnaire, and Limited Access data. You can click each category to find the datasets and documentation available in that category. For example, if you click on **Demographics Data**, you will find a list of datasets and documentation related to demographics data in the 2017-2018 NHANES cycle.

Data are available in 6 categories:

- Demographics
- Dietary
- Examination
- Laboratory
- Questionnaire
- Limited Access

## NHANES 2017-2018

[Print](#)

| Data, Documentation, Codebooks | Using the Data                         |
|--------------------------------|----------------------------------------|
| Demographics Data              | Overview                               |
| Dietary Data                   | Release Notes                          |
| Examination Data               | Laboratory Data Overview               |
| Laboratory Data                | Questionnaire Data Overview            |
| Questionnaire Data             | Examination Data Overview              |
| Limited Access Data            | Survey Methods and Analytic Guidelines |

## 2017-2018 Demographics Data - Continuous NHANES

[Print](#)

- [NHANES 2017-2018 Demographics Variable List](#)
- [Questionnaire Instruments](#)
- [Exam Procedure Manuals](#)
- [SAS Universal Viewer](#)
- [Data User Agreement](#)

| Data File Name                           | Doc File                   | Data File                                  | Date Published |
|------------------------------------------|----------------------------|--------------------------------------------|----------------|
| Demographic Variables and Sample Weights | <a href="#">DEMO_1.Doc</a> | <a href="#">DEMO_1 Data [XPT - 3.3 MB]</a> | February 2020  |

There is a **Doc File** available for each dataset that provides information about the dataset, including the variables it contains, their coding, and any special considerations for using it. It is important to read the **Doc** file before using the dataset to understand the data structure and how to use it properly. The **Data File** contains the dataset data in **.XPT** format, which is a SAS Transport file, not a CSV.

## 2017-2018 Demographics Data - Continuous NHANES

[Print](#)

- [NHANES 2017-2018 Demographics Variable List](#)
- [Questionnaire Instruments](#)
- [Exam Procedure Manuals](#)
- [SAS Universal Viewer](#)
- [Data User Agreement](#)

| Data File Name                           | Doc File                   | Data File                                  | Date Published |
|------------------------------------------|----------------------------|--------------------------------------------|----------------|
| Demographic Variables and Sample Weights | <a href="#">DEMO_J Doc</a> | <a href="#">DEMO_J Data [XPT - 3.3 MB]</a> | February 2020  |

## Data

You need to click on the [DEMO\\_J Data \[XPT - 3.3 MB\]](#) link to download the dataset. Note that you cannot easily open an .XPT file in Excel or Google Sheets. In the real world, data isn't always in a friendly format. For this week, let's download the `NHANES_Demo_Converted.csv` file from our Canvas page to look at the rows. Later, we will use an AI co-pilot to write R code that reads .XPT files automatically.

Let's open the `NHANES_Demo_Converted.csv` file in Excel and see the first few rows:

|    | A     | B        | C        | D        | E        | F        | G        |
|----|-------|----------|----------|----------|----------|----------|----------|
| 1  | SEQN  | SDDSRVYR | RIDSTATR | RIAGENDR | RIDAGEYR | RIDAGEMN | RIDRETH1 |
| 2  | 93703 | 10       | 2        | 2        | 2        | NA       | 5        |
| 3  | 93704 | 10       | 2        | 1        | 2        | NA       | 3        |
| 4  | 93705 | 10       | 2        | 2        | 66       | NA       | 4        |
| 5  | 93706 | 10       | 2        | 1        | 18       | NA       | 5        |
| 6  | 93707 | 10       | 2        | 1        | 13       | NA       | 5        |
| 7  | 93708 | 10       | 2        | 2        | 66       | NA       | 5        |
| 8  | 93709 | 10       | 2        | 2        | 75       | NA       | 4        |
| 9  | 93710 | 10       | 2        | 2        | 0        | 11       | 3        |
| 10 | 93711 | 10       | 2        | 1        | 56       | NA       | 5        |
| 11 | 93712 | 10       | 2        | 1        | 18       | NA       | 1        |
| 12 | 93713 | 10       | 2        | 1        | 67       | NA       | 3        |
| 13 | 93714 | 10       | 2        | 2        | 54       | NA       | 4        |
| 14 | 93715 | 10       | 2        | 1        | 71       | NA       | 5        |

- Each row represents a person.
- Each column represents a variable.
- The first row contains the variable names. The remaining rows contain the data for each person.

For example, `SEQN` is the unique identifier for each person in the dataset, and `RIAGENDR` is the variable for gender.

## Use of the Codebook

To understand the variables in the dataset, you can refer to the codebook. The codebook provides information about the variable names, labels, and values. The Doc File explained earlier contains the codebook for the dataset. You need to click on the DEMO\_J Doc link to open the codebook on a new page. Let's we want to understand the variables SEQN and RIAGENDR:

### National Health and Nutrition Examination Survey

2017-2018 Data Documentation, Codebook, and Frequencies

Demographic Variables and Sample Weights (DEMO\_J)

Data File: DEMO\_J.xpt

First Published: February 2020

Last Revised: NA

#### Component Description

The demographics file provides individual, family, and household-level information on the following topics:

- Survey participant's household interview and examination status;

#### TABLE OF CONTENTS

- Component Description
- Eligible Sample
- Interview Setting and Mode of Administration
- Quality Assurance & Quality Control
- Data Processing and Editing
- Analytic Notes
- References
- Codebook
  - SEQN - Respondent sequence number
  - SDDSRVYR - Data release cycle
  - RIDSTATR - Interview/Examination status
  - RIAGENDR - Gender
  - RIDAGEYR - Age in years at screening
  - RIDAGEMN - Age in months at screening - 0 to 24 mos

You need to click on the **SEQN - Respondent sequence number** link or scroll down to find the information about the SEQN variable.

#### SEQN - Respondent sequence number

**Variable Name:** SEQN  
**SAS Label:** Respondent sequence number  
**English Text:** Respondent sequence number.  
**Target:** Both males and females 0 YEARS - 150 YEARS

#### SDDSRVYR - Data release cycle

**Variable Name:** SDDSRVYR  
**SAS Label:** Data release cycle  
**English Text:** Data release cycle

#### TABLE OF CONTENTS

- Component Description
- Eligible Sample
- Interview Setting and Mode of Administration
- Quality Assurance & Quality Control
- Data Processing and Editing
- Analytic Notes
- References
- Codebook
  - SEQN - Respondent sequence number
  - SDDSRVYR - Data release cycle
  - RIDSTATR - Interview/Examination status
  - RIAGENDR - Gender

Similarly, you can click on the **RIAGENDR - Gender** link or scroll down to find the information about the RIAGENDR variable.

#### RIAGENDR - Gender

**Variable Name:** RIAGENDR  
**SAS Label:** Gender  
**English Text:** Gender of the participant.  
**Target:** Both males and females 0 YEARS - 150 YEARS

| Code or Value | Value Description | Count | Cumulative | Skip to Item |
|---------------|-------------------|-------|------------|--------------|
| 1             | Male              | 4557  | 4557       |              |
| 2             | Female            | 4697  | 9254       |              |
| .             | Missing           | 0     | 9254       |              |

#### RIDAGEYR - Age in years at screening

**Variable Name:** RIDAGEYR

#### TABLE OF CONTENTS

- Component Description
- Eligible Sample
- Interview Setting and Mode of Administration
- Quality Assurance & Quality Control
- Data Processing and Editing
- Analytic Notes
- References
- Codebook
  - SEQN - Respondent sequence number
  - SDDSRVYR - Data release cycle
  - RIDSTATR - Interview/Examination status
  - RIAGENDR - Gender
  - RIDAGEYR - Age in years at screening
  - RIDAGEMN - Age in months at screening - 0 to 24 mos

As you can see, the **RIAGENDR** variable has two values: 1 for Male and 2 for Female, without any missing values. You can also see the frequency of each value in the dataset. These numbers/frequencies should exactly match the frequencies in your dataset. You can use the codebook to understand other variables in the dataset. Similarly, you can use the other data files (e.g., Dietary) and their codebook to understand the variables in those datasets. Unlike the demographic file, which uses only one dataset and one codebook, the other files use multiple datasets and codebooks. As an example, if you click on the **Questionnaire Data** link, you will find multiple datasets and codebooks for questionnaire data:

## 2017-2018 Questionnaire Data - Continuous NHANES

[Print](#)

- [NHANES 2017-2018 Questionnaire Variable List](#)
- [Questionnaire Instruments](#)
- [2017-2018 Questionnaire Data Overview](#)
- [SAS Universal Viewer](#)
- [Data User Agreement](#)

| Data File Name                    | Doc File                     | Data File                                   | Date Published |
|-----------------------------------|------------------------------|---------------------------------------------|----------------|
| Acculturation                     | <a href="#">ACO.J.Doc</a>    | <a href="#">ACO.J.Data [XPT - 396.3 KB]</a> | February 2020  |
| Alcohol Use                       | <a href="#">ALO.J.Doc</a>    | <a href="#">ALO.J.Data [XPT - 434.4 KB]</a> | December 2020  |
| Audiometry                        | <a href="#">AUO.J.Doc</a>    | <a href="#">AUO.J.Data [XPT - 3.9 MB]</a>   | July 2020      |
| Blood Pressure & Cholesterol      | <a href="#">BPO.J.Doc</a>    | <a href="#">BPO.J.Data [XPT - 531.8 KB]</a> | February 2020  |
| Cardiovascular Health             | <a href="#">CDO.J.Doc</a>    | <a href="#">CDO.J.Data [XPT - 518.7 KB]</a> | February 2020  |
| Consumer Behavior                 | <a href="#">CBO.J.Doc</a>    | <a href="#">CBO.J.Data [XPT - 435.4 KB]</a> | April 2021     |
| Consumer Behavior Phone Follow-up | <a href="#">CBOPFA.J.Doc</a> | <a href="#">CBOPFA.J.Data [XPT - 2.7]</a>   | June 2021      |

# BRFSS

The Behavioral Risk Factor Surveillance System (BRFSS) is a cross-sectional survey conducted by the US Centers for Disease Control (CDC). The survey collects information on health-related risk behaviors, chronic health conditions, and use of preventive services among US adults. BRFSS completes more than 400,000 adult interviews each year, making it the largest continuously conducted health survey system in the world.



[CDC website for BRFSS](#)

## About BRFSS

You can find more information about the BRFSS survey by clicking on [BRFSS FAQs](#):



On the BRFSS FAQs page, you will find information about the BRFSS survey, including what the survey collects, how the survey is conducted, and how to use the data.

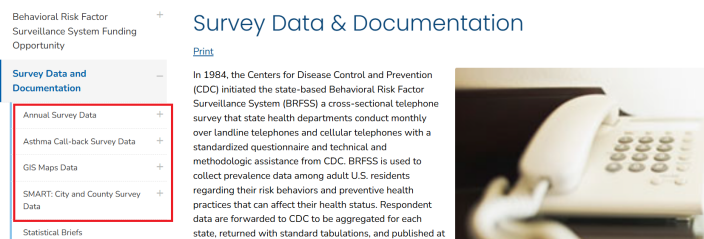
## Survey Data & Documentation

If you click on Survey Data & Documentation link, you will find information about the BRFSS questionnaires, datasets, and documentation:



There are four types of data files available for download: 1) annual data, 2) asthma call-back data, 3) GIS map data, and 4) SMART data. The annual data files include information on health-related risk behaviors, chronic health conditions, and use of preventive services among US adults. The asthma call-back data files include information on asthma-related health outcomes and healthcare utilization among adults with asthma. The GIS map data files include information on the geographic distribution of the data. The SMART (Selected Metropolitan/Micropolitan Area Risk Trends) data provide prevalence rates for selected conditions and behaviors for cities and their surrounding counties.

## BRFSS Survey Data & Documentation



For the annual data, you can click on the **Annual Survey Data** link to find information about the annual data files, including how to access the data, how to use the data, and how to analyze the data. There are two types of annual data files available for download: 1) the raw data files and 2) the prevalence and trends database.

## Annual Survey Data

There are separate links for each year of data, and you can click on each link to find information about the data for that year. For example, if you click on the **2023 Annual Survey Data** link, you will find information about the 2023 annual data files, including how to access the data, how to use the data, and how to analyze the data.

### Annual Survey Data



## 2023 BRFSS Survey Data and Documentation

[Print](#)

The 2023 BRFSS data continue to reflect the changes initially made in 2011 for weighting methodology (raking) and adding cell-phone-only respondents. The aggregate BRFSS combined landline and cell phone data set is built from the landline and cell phone data submitted for 2023 and includes data from 48 states, the District of Columbia, Guam, Puerto Rico, and the US Virgin Islands. During 2023, Kentucky and Pennsylvania were unable to collect enough data to meet the minimum requirements to be included in this public data set. The data set has been modified to comply with President Trump's executive orders. There are some missing values which may appear to be inconsistencies in the data based on a respondents' answers to questions that were removed.

### 2023 Survey Data Information

[2023 BRFSS Overview CDC](#)  [PDF – 213 KB]

The 2023 BRFSS Survey Data and Documentation page provides information about three resources: 1) data information such as the codebook and sampling weights, 2) data files in ASCII and SAS XPT transport formats, and 3) SAS resources. To download/save the data files, click the specific format you want. For example, if you click on the **2023 BRFSS Data (SAS Transport Format)** link, the annual survey data for 2023 will be downloaded in a ZIP file that contains the dataset in a .XPT format.

### Data Files

There are 433,323 records for 2023. More information on participation is available in the [states conducting surveillance, by year table](#). The data files are provided in ASCII and SAS Transport formats. The September update includes updated race/ethnicity calculated variables for Mississippi. The February update includes changes to the data set which has been modified to comply with President Trump's executive orders. There are some missing values which may appear to be inconsistencies in the data based on a respondents' answers to questions that were removed.

[2023 BRFSS Data \(ASCII\)](#)  [ZIP – 41.5 MB]

February, 2025

This file for the combined landline and cell phone data set is in ASCII format. It has a fixed record length of 2111 positions.

[2023 BRFSS Data \(SAS Transport Format\)](#) 

[ZIP – 64.3 MB]

February, 2025

This file for the combined landline and cell phone data set was exported from SAS V9.4 in the XPT transport

You need to unzip the file to extract the data file in .XPT format. However, opening the data file in Excel or Google Sheets is a bit tricky. The SAS XPT (or ASCII) data file needs to be converted/formatted appropriately before it can be opened in Excel or Google Sheets. Many statistical software packages, such as R, SAS, or STATA, can read the data file in .XPT format. Below is the data file, where the column names are the variable names, and the rows are the observations (i.e., survey respondents):

|    | STATE FIPS CODE | FILE MONTH | INTERVIEW DATE | INTERVIEW MONTH | INTERVIEW DAY | INTERVIEW YEAR | FINAL DISPOSITION | ANNUAL SEQUENCE NUMBER | PRIMARY SAMPLING UNIT | CORRECT TELEPHONE NUMBER? | PRIVATE RESIDENCE? | DO YOU LIVE IN COLLEGE HOUSING? | RESIDENT OF STATE | CELLULAR TELEPHONE | ARE YOU 18 YEARS OF AGE OR OLDER? |
|----|-----------------|------------|----------------|-----------------|---------------|----------------|-------------------|------------------------|-----------------------|---------------------------|--------------------|---------------------------------|-------------------|--------------------|-----------------------------------|
| 1  | 1               | 1          | 1 03012023     | 03              | 01            | 2023           | 1100              | 2023000001             | 2023000001            | 1                         | 1                  | -                               | 1                 | 2                  |                                   |
| 2  | 1               | 1          | 1 01062023     | 01              | 06            | 2023           | 1100              | 2023000002             | 2023000002            | 1                         | 1                  | -                               | 1                 | 2                  |                                   |
| 3  | 1               | 1          | 1 03082023     | 03              | 08            | 2023           | 1100              | 2023000003             | 2023000003            | 1                         | 1                  | -                               | 1                 | 2                  |                                   |
| 4  | 1               | 1          | 1 03062023     | 03              | 06            | 2023           | 1100              | 2023000004             | 2023000004            | 1                         | 1                  | -                               | 1                 | 2                  |                                   |
| 5  | 1               | 1          | 1 01062023     | 01              | 06            | 2023           | 1100              | 2023000005             | 2023000005            | 1                         | 1                  | -                               | 1                 | 2                  |                                   |
| 6  | 1               | 1          | 1 01092023     | 01              | 09            | 2023           | 1100              | 2023000006             | 2023000006            | 1                         | 1                  | -                               | 1                 | 2                  |                                   |
| 7  | 1               | 1          | 1 0212023      | 03              | 21            | 2023           | 1100              | 2023000007             | 2023000007            | 1                         | 1                  | -                               | 1                 | 2                  |                                   |
| 8  | 1               | 1          | 1 01062023     | 01              | 06            | 2023           | 1100              | 2023000008             | 2023000008            | 1                         | 1                  | -                               | 1                 | 2                  |                                   |
| 9  | 1               | 1          | 1 01152023     | 03              | 15            | 2023           | 1100              | 2023000009             | 2023000009            | 1                         | 1                  | -                               | 1                 | 2                  |                                   |
| 10 | 1               | 1          | 1 01082023     | 01              | 08            | 2023           | 1100              | 2023000010             | 2023000010            | 1                         | 1                  | -                               | 1                 | 2                  |                                   |
| 11 | 1               | 1          | 1 03082023     | 03              | 08            | 2023           | 1100              | 2023000011             | 2023000011            | 1                         | 1                  | -                               | 1                 | 2                  |                                   |
| 12 | 1               | 1          | 1 01082023     | 01              | 08            | 2023           | 1100              | 2023000012             | 2023000012            | 1                         | 1                  | -                               | 1                 | 2                  |                                   |
| 13 | 1               | 1          | 1 03012023     | 03              | 01            | 2023           | 1100              | 2023000013             | 2023000013            | 1                         | 1                  | -                               | 1                 | 2                  |                                   |
| 14 | 1               | 1          | 1 01062023     | 01              | 06            | 2023           | 1200              | 2023000014             | 2023000014            | 1                         | 1                  | -                               | 1                 | 2                  |                                   |
| 15 | 1               | 1          | 1 0312023      | 03              | 21            | 2023           | 1100              | 2023000015             | 2023000015            | 1                         | 1                  | -                               | 1                 | 2                  |                                   |
| 16 | 1               | 1          | 1 0112023      | 03              | 13            | 2023           | 1100              | 2023000016             | 2023000016            | 1                         | 1                  | -                               | 1                 | 2                  |                                   |

To understand the variables in the dataset, you can refer to the codebook:

### 2023 Survey Data Information

[2023 BRFSS Overview CDC](#) [PDF – 213 KB]

Provides information on the background, design, data collection and processing, and the statistical and analytical issues for the combined landline and cell phone data set.

[2023 BRFSS Codebook CDC](#) [ZIP – 3 MB]

Codebook for the file showing variable name, location, and frequency of values for all reporting areas combined for the combined landline and cell phone data set.

You need to unzip the file to extract the codebook in HTML format. The codebook provides information about the variable names, labels, and values. For example, if you want to understand the variable **Drinking and Driving**, you can find the variable name, label, and values in the codebook:

| Label: Drinking and Driving<br>Section Name: Calculated Variables<br>Module Number: 15<br>Question Number: 3<br>Column: 2110<br>Type of Variable: Num<br>SAS Variable Name: _DRNKDRV<br>Question Prologue:<br>Question: Drinking and Driving (Reported having driven at least once when perhaps had too much to drink) |                                                |           |            |                     |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------|-----------|------------|---------------------|
| Value                                                                                                                                                                                                                                                                                                                  | Value Label                                    | Frequency | Percentage | Weighted Percentage |
| 1                                                                                                                                                                                                                                                                                                                      | Have driven after having too much to drink     | 6,192     | 1.43       | 1.36                |
| 2                                                                                                                                                                                                                                                                                                                      | Have not driven after having too much to drink | 205,541   | 47.43      | 45.81               |
| 9                                                                                                                                                                                                                                                                                                                      | Don't know/Not Sure/Refused/Missing            | 221,590   | 51.14      | 52.83               |

You can similarly download the asthma call-back, GIS map, and SMART datasets. To understand the variables in those datasets, you can refer to the codebook for each dataset.

[Asthma Call-back Data](#)

[GIS Data](#)

[SMART Data](#)

## Prevalence and Trends Database

The BRFSS prevalence and trends tool allows users to view and download prevalence estimates through charts, graphs, and maps. You can explore the estimates by selecting the topic or location. For example, if you select by **topic**, e.g., for alcohol consumption for Alabama for the year 2024, you need to select the options from the dropdown menu and click on the **View Results** button to see the results:

**CDC** BRFSS Prevalence & Trends Data Search

Per a court order, HHS is required to restore this website to its version as of 12:00 AM on January 29, 2025. Information on this page may be modified and/or removed in the future subject to the terms of the court's order and implemented consistent with applicable law. Any information on this page promoting gender ideology is extremely inaccurate and disconnected from truth. The Trump Administration rejects gender ideology due to the harms and divisiveness it causes. This page does not reflect reality and therefore the Administration and this Department reject it.

[Print](#)

The enhanced BRFSS Prevalence & Trends Tool allows you to explore prevalence data by location or topic.

- Just like the original tool, this version allows you to view and download prevalence estimates through charts, graphs, and maps.
- In addition, this version also provides access to prevalence estimates from the BRFSS core data at the state level as well as data from Selected Metropolitan/Micropolitan Area Risk Trends (SMART).

These prevalence estimates have been updated to include both crude and age-adjusted prevalence.

The tool will continue to be updated as new functions and data become available.



**Explore by Location** **Explore by Topic**

Explore bar and trendline graphs for survey questions by selecting a state, territories or Metropolitan/Micropolitan Statistical Areas (MMSAs).

| Dataset    | Location         | Class               | Topic          | Year |
|------------|------------------|---------------------|----------------|------|
| Nationwide | States - Alabama | Alcohol Consumption | Binge Drinking | 2024 |

Question: Binge drinkers (males having five or more drinks on one occasion, females having four or more drinks on one occasion) [variable calculated from one or more BRFSS questions]

**View Results** [Clear Filters](#)

Prevalence and Trends Database

The results will show the (i) prevalence estimates for alcohol consumption in Alabama for the year 2024, and (ii) the trend of alcohol consumption in Alabama over time.

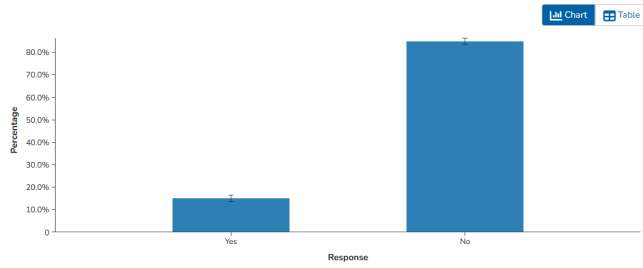
[View Results](#) [Clear Filters](#)

Binge drinkers (males having five or more drinks on one occasion, females having four or more drinks on one occasion) (variable calculated from one or more BRFSS questions)

*Alcohol Consumption: Binge Drinking*

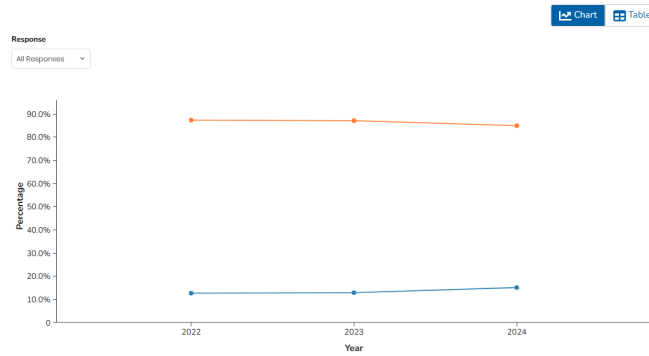
View By:  Data Type:

2024, Alabama



The ordering of subcategory rows in the table view may vary across responses.

All Available Years, Alabama



# CCS

The Canadian Cannabis Survey (CCS) is an annual survey conducted by Health Canada to gather information on cannabis use among people aged 16 years and older living in all Canadian provinces and territories. The survey collects data on various aspects of cannabis consumption, including their knowledge, attitudes and behaviours towards cannabis use.



Government of Canada  
Gouvernement du Canada

Franglais

Search Canada.ca

MENU

[Canada.ca](#) > [Open Government](#) > [Search](#)

### Canadian Cannabis Survey (CCS) 2024 Public Use Microdata File (PUMF)

Have your say

\*\*\*\*\*  
[Rate this dataset](#)  
[Provide feedback](#)

The 2024 Canadian Cannabis Survey (CCS) recruited people aged 16 years and older living in all Canadian provinces and territories. Respondents completed a survey on patterns of cannabis use for medical and non-medical purposes, as well as their knowledge, attitudes and behaviours towards cannabis use. A total of 11,666 respondents completed online surveys between April 4 and July 2, 2024. Summary results are available here: <https://www.canada.ca/en/health-canada/services/drugs-medication/cannabis/research-data/canadian-cannabis-survey-2024-summary.html>

Additional

The Office Cannabis Science and Surveillance releases the PUMF for CCS 2024 to allow

More than 11,000 respondents participated in the 2024 survey. The study provides valuable insights into cannabis use patterns, the cannabis market, and issues of public safety. The survey results are published on Health Canada’s website:

[Health Canada website for CCS](#)

593



Government of Canada
Gouvernement du Canada

Franglais

MENU

[Canada.ca](#) > [Open Government](#) > [Search](#)

## Canadian Cannabis Survey (CCS) 2024 Public Use Microdata File (PUMF)

Have your say

Rate this dataset

Provide feedback

The 2024 Canadian Cannabis Survey (CCS) recruited people aged 16 years and older living in all Canadian provinces and territories. Respondents completed a survey on patterns of cannabis use for medical and non-medical purposes, as well as their knowledge, attitudes and behaviours towards cannabis use. A total of 11,666 respondents completed online surveys between April 4 and July 2, 2024. Summary results are available here:

<https://www.canada.ca/en/health-canada/services/drugs-medication/cannabis/research-data/canadian-cannabis-survey-2024-summary.html>

Additional

The Office Cannabis Science and Surveillance releases the PUMF for CCS 2024 to allow



Government of Canada
Gouvernement du Canada

Franglais

MENU

[Canada.ca](#) > [Health](#) > [Drug and health products](#) > [Drugs and medication](#) > [Cannabis](#) > [Cannabis research and data](#)

## Canadian Cannabis Survey 2024: Summary

On this page

- [Introduction](#)
- [Results by theme](#)
- [Methods](#)
- [Considerations](#)

## Open Licence

The data from the Canadian Cannabis Survey is available under the Open Government Licence - Canada (OGL-Canada). This licence allows users to freely use, modify, and distribute the data, provided that they attribute the source of the data and do not use it for commercial purposes. The OGL-Canada promotes transparency and encourages the use of government data for research, analysis, and public benefit. You can follow the [Open Government Licence - Canada](#) link for more information on the terms and conditions of the licence.

[Open Licence](#)

Additional Information

**Keywords:**  
cannabis | drugs

**Subject:**  
Health and Safety

**Maintenance and Update Frequency:**

[data/cannabis-survey-2024-summary.html](#)

The Office Cannabis Science and Surveillance releases the PUMF for CCS 2024 to allow researchers, provincial governments and others to conduct their own analyses of this important data.

**Publisher - Current Organization Name:** Health Canada

**Publisher - Organization Name at Publication:** Office of Cannabis Science and Surveillance, Strategic Policy Directorate, Controlled Substances and Cannabis Branch, Health Canada

**Licence:** [Open Government Licence - Canada](#)

## Open Government Licence - Canada

You are encouraged to use the Information that is available under this licence with only a few conditions.

### Using Information under this licence

- Use of any Information indicates your acceptance of the terms below.
- The Information Provider grants you a worldwide, royalty-free, perpetual, non-exclusive licence to use the Information, including for commercial purposes, subject to the terms below.

## Data and Resources

The **Data and Resources** tab on the Health Canada website provides access to the public use data, user guides, and the data dictionary.

Data and Resources

Search resources...

Q

CCS

Canadian Cannabis Survey (CCS) 2024

English

dataset

CSV

Explore

CCS

Canadian Cannabis Survey (CCS) 2024 PUMF User Guide

English

guide

DOCX

Explore

CCS

Canadian Cannabis Survey (CCS) 2024 PUMF User Guide

French

guide

DOCX

Explore

CCS

Public Use Microdata Files (PUMF) Derived Variable Specifications

English

specification

DOCX

Explore

CCS

Public Use Microdata Files (PUMF) Derived Variable Specifications

French

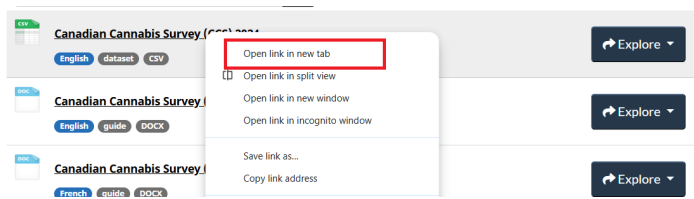
specification

DOCX

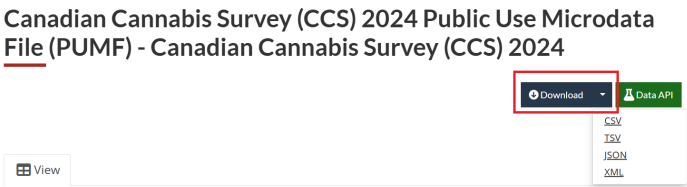
Explore

The CCS 2024 data is available to download in various formats, including CSV, TSV, JSON, and XML. The data can be accessed through the Health Canada website, where you can choose the format that best suits your needs. You need to

open the data link (say, in a new tab in your browser) and select the desired format to download the data.



Under the **Download** tab, you can find the data in different formats. For example, you can download the data in CSV format by clicking on the **CSV** link:



After downloading, you can open the data in Excel, Google Sheets, or any statistical software. The data is structured in a tabular format, with each row representing an individual respondent and each column representing different variables:

|    | A      | B    | C             | D     | E   | F               | G              | H    | I          | J               | K             | L          | M            |
|----|--------|------|---------------|-------|-----|-----------------|----------------|------|------------|-----------------|---------------|------------|--------------|
| 1  | seq_id | wave | sample_weight | age65 | sex | sexuality_recod | community_size | prov | race_white | health_physical | health_mental | education3 | income_recod |
| 2  | 1      | 7    | 2622.409      | 2     | 2   | 1               | 1              | 7    | 1          | 2               | 2             | 3          | 3            |
| 3  | 2      | 7    | 1517.9545     | 4     | 1   | 1               | -7             | 5    | 1          | 2               | 3             | 3          | 2            |
| 4  | 3      | 7    | 1547.9545     | 1     | 1   | 1               | 1              | 1    | 0          | 2               | 4             | 1          | 1            |
| 5  | 4      | 7    | 3095          | 1     | 2   | 1               | 2              | 7    | 1          | 4               | 3             | 1          | 2            |
| 6  | 5      | 7    | 2475.0746     | 4     | 1   | 2               | 2              | 2    | 0          | 2               | 4             | 3          | 4            |
| 7  | 6      | 7    | 2206.2006     | 6     | 1   | 1               | 2              | 2    | 1          | 1               | 1             | 2          | 3            |
| 8  | 7      | 7    | 4033.9425     | 4     | 1   | 1               | 1              | 7    | 0          | 4               | 3             | 1          | 2            |
| 9  | 8      | 7    | 1933.1395     | 1     | 2   | 1               | 2              | 9    | 1          | 1               | 1             | 1          | 3            |
| 10 | 9      | 7    | 4273.64       | 3     | 1   | 2               | 9              | 0    | 4          | 3               | 3             | -8         |              |
| 11 | 10     | 7    | 1512.9487     | 5     | 1   | -8              | -8             | 6    | -8         | 3               | 2             | 1          | -8           |
| 12 | 11     | 7    | 2438.4091     | 4     | 2   | 1               | 2              | 1    | 0          | 3               | 4             | 2          | 3            |
| 13 | 12     | 7    | 1860.8133     | 1     | 1   | 1               | 2              | 5    | 1          | 3               | 4             | 1          | 1            |

The **Canadian Cannabis Survey (CCS) 2024 PUMF User Guide** gives detailed information on the survey methodology, data collection process, and how to use the public use microdata file (PUMF). You can also find the variable names and their descriptions in this file.

[2024 Public Use Microdata File](#)



## Data and Resources

Search resources...

- Canadian Cannabis Survey (CCS) 2024  
English dataset CSV Explore
- Canadian Cannabis Survey (CCS) 2024 PUMF User Guide**  
English guide DOCK Explore
- Canadian Cannabis Survey (CCS) 2024 PUMF User Guide  
French guide DOCK Explore
- Public Use Microdata Files (PUMF) Derived Variable Specifications  
English specification DOCK Explore
- Public Use Microdata Files (PUMF) Derived Variable Specifications  
French specification DOCK Explore

Let's find a variable (`access_ease_legal`) in the Excel file and check its description in the data dictionary. You can use the Find function in Excel (Ctrl + F) to search for the variable name:

|   | cansirisk_cantli | risk_cand | risk_cane | access_ease_legal | access_eewarn_see | warn_crec | warn_kno | warr |
|---|------------------|-----------|-----------|-------------------|-------------------|-----------|----------|------|
| 3 | 3                | 3         | 3         | 3                 | 5                 | 4         | -7       | -7   |
| 2 | 2                | 2         | 2         | 4                 | 2                 | 4         | -7       | -7   |
| 1 | 1                | 1         | 1         | 4                 | 3                 | 1         | 1        | 1    |
| 4 | 4                | 4         | 4         | 4                 | 3                 | 4         | -7       | -7   |
| 2 | 3                | 5         | 5         | 4                 | 5                 | 1         | 4        | 4    |
| 5 | 5                | 5         | 5         | 4                 | 5                 | 4         | -7       | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 3         | 3        | 2    |
| 4 | 4                | 4         | 4         | 1                 | 3                 | 4         | -7       | -7   |
| 3 | 3                | 3         | 2         | 4                 | 3                 | 4         | -7       | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 | 4                | 4         | 4         | 4                 | 4                 | 5         | 7        | -7   |
| 3 | 3                | 3         | 3         | 4                 | 4                 | 5         | 7        | -7   |
| 2 | 2                | 2         | 2         | 4                 | 4                 | 5         | 7        | -7   |
| 1 | 1                | 1         | 1         | 4                 | 4                 | 5         | 7        | -7   |
| 4 |                  |           |           |                   |                   |           |          |      |

|                                                        |                                                                                                                                                                                                                                               |
|--------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>access_ease_legal</b>                               | How easy or difficult is it for you to get legal cannabis?                                                                                                                                                                                    |
| <i>[variable added in 2024 to replace access_ease]</i> | <input type="radio"/> 1 Very difficult<br><input type="radio"/> 2 Fairly difficult<br><input type="radio"/> 3 Fairly easy<br><input type="radio"/> 4 Very easy<br><input type="radio"/> 5 I don't know<br><input type="checkbox"/> -7 Missing |

In this dataset, -7 generally represents missing. Always check the data dictionary for the specific coding of missing values, “don’t know,” or “refused” responses, as they can vary between datasets.

# CADS

The Canadian Alcohol and Drugs Survey (CADS) is a national survey conducted by Statistics Canada that collects data on the use of substances, such as alcohol, cannabis and other drugs. The study was conducted by Statistics Canada in 2019 with the cooperation and support of Health Canada. The target population for the survey was all persons aged 15 years and over living in Canada, excluding residents of the Yukon, Northwest Territories, and Nunavut, full-time residents of institutions, and residents of Native reserves.



The screenshot shows the top of a Government of Canada website page. At the top left is the Canadian flag and the text "Government of Canada / Gouvernement du Canada". To the right is a search bar with the text "Search Canada.ca" and a magnifying glass icon. Below the header is a "MENU" button with a downward arrow. Under the menu is a breadcrumb trail: "Canada.ca > Open Government > Search". The main heading is "Canadian Alcohol and Drugs Survey: Public Use Microdata File". Below this heading is a box with the text "Have your say" and two links: "Rate this dataset" and "Provide feedback". To the right of this box is a paragraph of text: "This public use microdata file (PUMF) is from the Canadian Alcohol and Drugs Survey and includes information about the use of substances, including alcohol, cannabis and other drugs. This product is for users who prefer to do their own analysis by focusing on specific subgroups in the population or by cross-classifying variables that were not the focus of our release in The Daily." Below this paragraph is a box with the text "Publisher - Current Organization Name: Statistics Canada".

[CADS 2019 dataset on the CADS website](#)

## Open Licence

Similar to the [Canadian Cannabis Survey](#), the CADS is also licensed under the [Open Government Licence - Canada](#). This means that the data can be freely used, modified, and shared by anyone for any purpose, as long as they provide proper attribution to the source of the data.

## Canadian Alcohol and Drugs Survey: Public Use Microdata File

**Have your say**  
[Rate this dataset](#)  
[Provide feedback](#)

**Additional Information**

This public use microdata file (PUMF) is from the Canadian Alcohol and Drugs Survey and includes information about the use of substances, including alcohol, cannabis and other drugs. This product is for users who prefer to do their own analysis by focusing on specific subgroups in the population or by cross-classifying variables that were not the focus of our release in The Daily.

**Publisher - Current Organization Name:** Statistics Canada

**Licence:** [Open Government Licence - Canada](#)

 Government of Canada  
Gouvernement du Canada

[Canada.ca](#) > [Open Government](#) > [About Open Government](#)

### Open Government Licence - Canada

You are encouraged to use the Information that is available under this licence with only a few conditions.

#### Using Information under this licence

- Use of any Information indicates your acceptance of the terms below.

## Data and Resources

The **Data and Resources** tab provides access to the public use data, user guides, and the data dictionary.

### Data and Resources

 **Canadian Alcohol and Drugs Survey: Public Use Microdata File**  
[English](#) [French](#) [dataset](#) [CSV](#)

[Explore](#)

 **Canadian Alcohol and Drugs Survey: Public Use Microdata File**  
[English](#) [French](#) [dataset](#) [SAS](#)

[Explore](#)

 **Canadian Alcohol and Drugs Survey: Public Use Microdata File**  
[English](#) [website](#) [HTML](#)

[Explore](#)

 **Canadian Alcohol and Drugs Survey: Public Use Microdata File**  
[French](#) [website](#) [HTML](#)

[Explore](#)

The public use data file is available in a ZIP format. You need to open the **Canadian Alcohol and Drugs Survey: Public Use Microdata File** link, and click on **Go to resource** to download the ZIP file.

## Data and Resources

Search resources... 



Canadian Alcohol and Drugs Survey: Public Use Microdata File

English French dataset CSV

Explore



Canadian Alcohol and Drugs Survey: Public Use Microdata File

English French dataset SAS

Explore



Canadian Alcohol and Drugs Survey: Public Use Microdata File

English website HTML

Explore



Canadian Alcohol and Drugs Survey: Public Use Microdata File

French website HTML

Explore

[Canada.ca](#) > [Open Government](#) > [Search](#) > [Canadian Alcohol and Drugs Survey: Public Use Microdata File](#)

### Canadian Alcohol and Drugs Survey: Public Use Microdata File - Canadian Alcohol and Drugs Survey: Public Use Microdata File

 Go to resource

The ZIP file contains the following files: an instruction on how to cite CADS 2019, data documentation, and the dataset in CSV format.

|                                                                                                                                 |                    |                    |        |
|---------------------------------------------------------------------------------------------------------------------------------|--------------------|--------------------|--------|
|  How to cite CADS2019_Comment citer ECAD2019 | 27-Jul-21 10:42 AM | Adobe Acrobat D... | 388 KB |
|  Documentation                               | 05-Aug-21 6:56 AM  | File folder        |        |
|  Data_donnees                                | 05-Aug-21 6:46 AM  | File folder        |        |

## Data Documentation

The data documentation folder includes three sub-folders: Codebook\_Livre des codes, Guide, and Questionnaires.

|                                                                                                              |                   |             |
|--------------------------------------------------------------------------------------------------------------|-------------------|-------------|
|  Codebook_Livre des codes | 05-Aug-21 6:56 AM | File folder |
|  Guide                    | 05-Aug-21 6:56 AM | File folder |
|  Questionnaires           | 05-Aug-21 6:56 AM | File folder |

The **Guide** folder contains user guides (in English and French) that provide an overview of the survey, including the methodology, data collection process, data processing, data quality, and sampling weights.

|                                                                                                |                |                   |                    |          |
|------------------------------------------------------------------------------------------------|----------------|-------------------|--------------------|----------|
|  ECAD2019_GuF | <b>French</b>  | 14-Jul-21 6:47 AM | Adobe Acrobat D... | 1,653 KB |
|  CADS2019_UgE | <b>English</b> | 14-Jul-21 6:45 AM | Adobe Acrobat D... | 1,306 KB |

The **Codebook\_Livre des codes** folder contains codebooks (in English and French) that provide detailed information about the variables in the dataset, including variable names, labels, and values. There are two types of codebooks: those with and those without variable frequencies. But both codebooks provide the same information about the variables in the dataset.

**Codebook including frequencies for variables**

|                                                                                                               |                |                    |                    |          |
|---------------------------------------------------------------------------------------------------------------|----------------|--------------------|--------------------|----------|
|  ECAD2019_FMGD_lvCd          | <b>French</b>  | 22-Dec-20 11:50 AM | Adobe Acrobat D... | 1,948 KB |
|  CADS2019_PUMF_Cdbk          |                | 22-Dec-20 11:41 AM | Adobe Acrobat D... | 1,925 KB |
|  ECAD2019_FMGD_lvCd_zerofreq |                | 22-Dec-20 11:24 AM | Adobe Acrobat D... | 1,885 KB |
|  CADS2019_PUMF_Cdbk_zerofreq | <b>English</b> | 20 11:16 AM        | Adobe Acrobat D... | 1,863 KB |

**Codebook excluding frequencies for variables**

The **Questionnaires** folder contains the survey questionnaires (in English and French) that were used to collect the data.

## Data

The **Data\_donnees** folder contains the dataset in CSV format. You can open the dataset in Excel or any statistical software to explore the data. The row represents the respondent, and the column represents the variable:

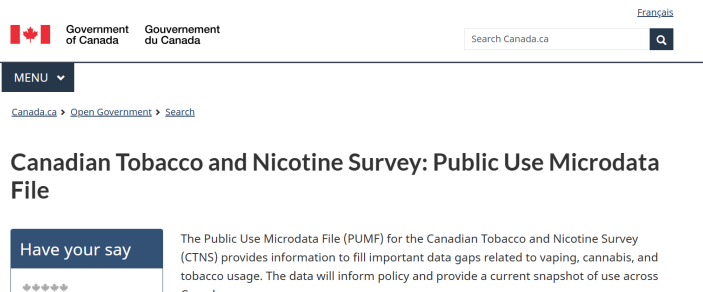
| CADS                                                                              |        |         |         |      |         |          |     |        |         |        |        |        |        |         |       |   |
|-----------------------------------------------------------------------------------|--------|---------|---------|------|---------|----------|-----|--------|---------|--------|--------|--------|--------|---------|-------|---|
| File Home Insert Draw Page Layout Formulas Data Review View Automate Help Acrobat |        |         |         |      |         |          |     |        |         |        |        |        |        |         |       |   |
| A1 PUMFID                                                                         |        |         |         |      |         |          |     |        |         |        |        |        |        |         |       |   |
|                                                                                   | A      | B       | C       | D    | E       | F        | G   | H      | I       | J      | K      | L      | M      | N       | O     | P |
|                                                                                   | PUMFID | REFYEAR | REFMONT | PROV | DVURBAN | MHHTYPEB | SEX | DV_AGE | HHLI_01 | HDSIZE | HWB_05 | HWB_10 | ALC_05 | ALC_10P | ALC_J |   |
| 1                                                                                 | 1000   | 2019    | 12      | 46   | 2       | 3        | 1   | 8      | 1       | 2      | 2      | 2      | 2      | 1       | 16    |   |
| 2                                                                                 | 1001   | 2019    | 9       | 46   | 2       | 6        | 1   | 5      | 1       | 4      | 1      | 1      | 1      | 1       | 17    |   |
| 3                                                                                 | 1002   | 2019    | 6       | 35   | 1       | 3        | 2   | 8      | 1       | 2      | 3      | 2      | 2      | 1       | 18    |   |
| 4                                                                                 | 1003   | 2019    | 8       | 11   | 1       | 3        | 1   | 8      | 1       | 3      | 2      | 2      | 2      | 1       | 18    |   |
| 5                                                                                 | 1004   | 2019    | 6       | 12   | 1       | 3        | 1   | 9      | 1       | 3      | 2      | 3      | 1      | 16      |       |   |
| 6                                                                                 | 1005   | 2019    | 10      | 12   | 1       | 3        | 1   | 8      | 1       | 2      | 2      | 2      | 1      | 1       | 99    |   |
| 7                                                                                 | 1006   | 2019    | 10      | 35   | 1       | 3        | 1   | 6      | 1       | 1      | 1      | 1      | 1      | 1       | 18    |   |
| 8                                                                                 | 1007   | 2019    | 12      | 48   | 1       | 3        | 1   | 7      | 1       | 2      | 4      | 4      | 1      | 1       | 16    |   |
| 9                                                                                 | 1008   | 2019    | 10      | 46   | 1       | 6        | 1   | 9      | 1       | 3      | 1      | 1      | 1      | 1       | 17    |   |
| 10                                                                                | 1009   | 2019    | 6       | 47   | 1       | 3        | 2   | 9      | 1       | 2      | 2      | 2      | 2      | 1       | 16    |   |
| 11                                                                                | 1010   | 2019    | 11      | 24   | 1       | 8        | 2   | 7      | 1       | 2      | 3      | 2      | 1      | 1       | 16    |   |

You can refer to the codebook to understand the variables that you are interested in for your analysis. For example, if you are interested in the variable DVURBAN, you can refer to the codebook to find out that the variable (e.g., for the English version of the codebook with frequencies):

|                   |                                                                                                                    |           |                    |           |    |
|-------------------|--------------------------------------------------------------------------------------------------------------------|-----------|--------------------|-----------|----|
| Variable Name:    | DVURBAN                                                                                                            | Length:   | 1.0                | Position: | 14 |
| Question Name:    |                                                                                                                    |           |                    |           |    |
| Concept:          | Characteristics of community where the respondent lives                                                            |           |                    |           |    |
| Question Text:    |                                                                                                                    |           |                    |           |    |
| Universe:         | All respondents                                                                                                    |           |                    |           |    |
| Note:             | The category 'Population centre' is historically comparable to 'Urban' which was used in previous years for CTUMS. |           |                    |           |    |
| Source:           |                                                                                                                    |           |                    |           |    |
| Answer Categories | Code                                                                                                               | Frequency | Weighted Frequency | %         |    |
| Population centre | 1                                                                                                                  | 7,208     | 21,321,106         | 68.8      |    |
| Rural             | 2                                                                                                                  | 2,438     | 5,122,120          | 16.5      |    |
| Not stated        | 9                                                                                                                  | 647       | 4,552,224          | 14.7      |    |
| Total             |                                                                                                                    | 10,293    | 30,995,450         | 100.0     |    |

# CTNS

The Canadian Tobacco and Nicotine Survey (CTNS) is a cross-sectional survey that collects information on tobacco and nicotine use among Canadians aged 15 years and older. The survey is conducted by Statistics Canada on behalf of Health Canada and provides valuable insights into the prevalence of cigarette smoking, vaping, cannabis, and alcohol use.



[CTNS 2022 dataset for researchers](#)

## Open Licence

The CTNS is licensed under the [Open Government Licence - Canada](#). This means that the data can be freely used, modified, and shared by anyone for any purpose, as long as they provide proper attribution to the source of the data.

## Data and Resources

The **Data and Resources** tab provides access to the public use data, user guides, and the data dictionary.



## Data and Resources

Search resources...

|                                                                                                                                                   |         |
|---------------------------------------------------------------------------------------------------------------------------------------------------|---------|
|  Canadian Tobacco and Nicotine Survey: Public Use Microdata File | Explore |
| English website HTML                                                                                                                              |         |
|  Canadian Tobacco and Nicotine Survey: Public Use Microdata File | Explore |
| French website HTML                                                                                                                               |         |
|  Canadian Tobacco and Nicotine Survey: Public Use Microdata File | Explore |
| English French dataset CSV                                                                                                                        |         |
|  Canadian Tobacco and Nicotine Survey: Public Use Microdata File | Explore |
| English French dataset SAS                                                                                                                        |         |

The public use data file is available in a ZIP format. You can click on the Canadian Tobacco and Nicotine Survey: Public Use Microdata File link, followed by Go to resource link to download the ZIP file.

## Data and Resources

Search resources...

|                                                                                                                                                     |         |
|-----------------------------------------------------------------------------------------------------------------------------------------------------|---------|
|  Canadian Tobacco and Nicotine Survey: Public Use Microdata File   | Explore |
| English website HTML                                                                                                                                |         |
|  Canadian Tobacco and Nicotine Survey: Public Use Microdata File  | Explore |
| French website HTML                                                                                                                                 |         |
|  Canadian Tobacco and Nicotine Survey: Public Use Microdata File | Explore |
| English French dataset CSV                                                                                                                          |         |
|  Canadian Tobacco and Nicotine Survey: Public Use Microdata File | Explore |
| English French dataset SAS                                                                                                                          |         |

[Canada.ca](#) > [Open Government](#) > [Search](#) > [Canadian Tobacco and Nicotine Survey: Public Use Microdata File](#)

### Canadian Tobacco and Nicotine Survey: Public Use Microdata File - Canadian Tobacco and Nicotine Survey: Public Use Microdata File

Go to resource

Similar to the [Canadian Alcohol and Drugs Survey \(CADS\)](#) tutorial, the ZIP file for CTNS contains the following files: citation instructions, data documentation, and the dataset in CSV format. The data documentation, questionnaire, and codebook are also the same as those for CADS. The only difference is that the data documentation and questionnaire for CTNS are labelled CTNS rather than CADS.

# CSADS

The Canadian Student Alcohol and Drugs Survey (CSADS) is a national survey that collects data on the alcohol and drug use patterns of Canadian students in grades 7 to 12. The survey is conducted every two years and provides valuable insights into the prevalence and trends of substance use among Canadian youth.

## Canadian Student Alcohol and Drugs Survey (CSADS): Public Use Microdata File

Have your say

★★★★★

[Rate this dataset](#)

[Provide feedback](#)

Every two years, Health Canada conducts the Canadian Student Alcohol and Drugs Survey (CSADS), a cross-sectional survey of students in grades 7 to 12 across the Canadian provinces.

With tens of thousands of participants each cycle, it is one of the largest youth surveys in the world. For over 30 years, the survey—previously known as the Youth Smoking Survey (YSS; pre-2014) and the Canadian Student Tobacco, Alcohol and Drugs Survey (CSTADS;

[CSADS Canada website files CTNS](#)

## Open Licence

The CTNS is licensed under the [Open Government Licence - Canada](#). This means that the data can be freely used, modified, and shared by anyone for any purpose, as long as they provide proper attribution to the source of the data.

## Data and Resources

The **Data and Resources** tab provides access to the public use data, user guides, the data dictionary, and information on bootstrap weights.

## Data and Resources

Search resources...

|                                                                                   |                                                   |                         |
|-----------------------------------------------------------------------------------|---------------------------------------------------|-------------------------|
|  | <b>2021-2022 CSTADS: PUMF — User Guide</b>        | <a href="#">Explore</a> |
|  | <b>2021-2022 ECTADE : FMGD — Mode d'emploi</b>    | <a href="#">Explore</a> |
|  | <b>2021-2022 CSTADS: PUMF — Data</b>              | <a href="#">Explore</a> |
|  | <b>2021-2022 CSTADS: PUMF — Bootstrap Weights</b> | <a href="#">Explore</a> |
|  | <b>2023-2024 CSADS: PUMF — User Guide</b>         | <a href="#">Explore</a> |
|  | <b>2023-2024 ECADE : FMGD — Mode d'emploi</b>     | <a href="#">Explore</a> |
|  | <b>2023-2024 CSADS: PUMF — Data</b>               | <a href="#">Explore</a> |
|  | <b>2023-2024 CSADS: PUMF — Bootstrap Weights</b>  | <a href="#">Explore</a> |

To download the 2021–2022 CSTADS, you can click on the [2021-2022 CSTADS: PUMF – Dataset in CSV format](#) link. The data is available to download in various formats. Under the **Download** tab, you can find the formats. For example, you can download the data in CSV format by clicking on the [CSV](#) link:

**MENU**

[Canada.ca](#) > [Open Government](#) > [Search](#) > [Canadian Student Alcohol and Drugs Survey \(CSADS\): Public Use Microdata File](#)

### Canadian Student Alcohol and Drugs Survey (CSADS): Public Use Microdata File - 2021-2022 CSTADS: PUMF — Data

[Download](#)
[Data API](#)

[View](#)

[CSV](#)  
[TSV](#)  
[JSON](#)  
[XML](#)

After downloading, you can open the data in Excel, Google Sheets, or any statistical software. The data is structured in a tabular format, with each row representing an individual respondent and each column representing different variables:

|    | A     | B      | C     | D        | E       | F     | G        | H       | I        | J      | K      | L      | M      | N      | O      |
|----|-------|--------|-------|----------|---------|-------|----------|---------|----------|--------|--------|--------|--------|--------|--------|
| 1  | SEQID | PROVID | GRADE | DVGENDEI | DVURBAN | DVRES | DVORIENT | DVDESCR | WTPUMF   | GH_010 | GH_020 | SS_010 | SS_020 | TS_011 | SS_030 |
| 2  | 18338 | 11     | 9     | 2        | 1       | 3     | 2        | 1       | 1.164716 | 2      | 3      | 2      | 96     | 4      | 96     |
| 3  | 16111 | 24     | 10    | 1        | 1       | 1     | 2        | 4       | 36.14117 | 3      | 3      | 2      | 96     | 4      | 96     |
| 4  | 20587 | 10     | 7     | 2        | 1       | 1     | 2        | 1       | 1.448578 | 2      | 1      | 2      | 96     | 4      | 96     |
| 5  | 54568 | 12     | 8     | 99       | 2       | 1     | 1        | 1       | 3.03666  | 4      | 5      | 2      | 96     | 3      | 96     |
| 6  | 40991 | 10     | 7     | 1        | 2       | 1     | 2        | 1       | 3.745233 | 2      | 3      | 2      | 96     | 4      | 96     |
| 7  | 5254  | 12     | 7     | 1        | 1       | 2     | 2        | 1       | 10.0715  | 3      | 3      | 2      | 96     | 4      | 96     |
| 8  | 10440 | 35     | 9     | 99       | 1       | 1     | 99       | 1       | 75.0031  | 5      | 4      | 2      | 96     | 4      | 96     |
| 9  | 38149 | 48     | 12    | 1        | 1       | 1     | 2        | 1       | 25.83337 | 1      | 3      | 1      | 16     | 4      | 2      |
| 10 | 32243 | 59     | 11    | 2        | 1       | 1     | 2        | 1       | 47.16534 | 1      | 3      | 1      | 16     | 3      | 2      |
| 11 | 5513  | 35     | 11    | 1        | 2       | 1     | 99       | 1       | 99.52681 | 2      | 5      | 2      | 96     | 3      | 96     |
| 12 | 37146 | 48     | 10    | 1        | 1       | 2     | 2        | 4       | 24.62732 | 3      | 2      | 2      | 96     | 4      | 96     |
| 13 | 21979 | 46     | 9     | 1        | 2       | 1     | 2        | 1       | 65.69607 | 1      | 4      | 2      | 96     | 4      | 96     |
| 14 | 33925 | 12     | 7     | 1        | 2       | 3     | 1        | 1       | 3.60877  | 3      | 4      | 2      | 96     | 4      | 96     |
| 15 | 19919 | 94     | 7     | 2        | 1       | 2     | 00       | 00      | 76.14944 | 9      | 9      | 9      | 00     | 4      | 00     |

The user guide gives detailed information on the survey methodology, data collection process, and how to use the public use microdata file (PUMF). You can also find the variable names and their descriptions in this DOCX file.

2021-2022 CSTADS: PUMF — **User Guide**

[English](#)
[guide](#)
[DOCX](#)

Explore

2021-2022 ECTADE : FMGD — Mode d'emploi

[French](#)
[guide](#)
[DOCX](#)

Explore

2021-2022 CSTADS: PUMF — **Data**

[English](#)
[French](#)
[dataset](#)
[CSV](#)

Explore

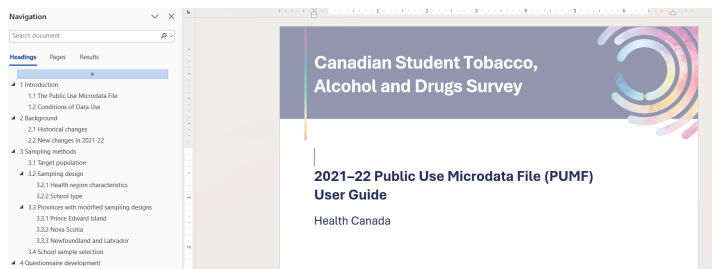
2021-2022 CSTADS: PUMF — Bootstrap Weights

[English](#)
[French](#)
[dataset](#)
[CSV](#)

Explore

## Canadian Student Alcohol and Drugs Survey (CSADS): Public Use Microdata File - 2021-2022 CSTADS: PUMF – User Guide

Download



To find a variable description, you are referred to the **User Guide**. For example, the variable GH\_010 is described as follows:

| Navigation                                      |       |         |   |
|-------------------------------------------------|-------|---------|---|
| SH_010                                          |       | WTPUMP  | 9 |
| 1 result                                        |       |         |   |
| Headings                                        | Pages | Results |   |
| 6.2.1 Skip patterns and inconsistent responses  |       |         |   |
| 6.2.2 Imputation of core smoking questions      |       |         |   |
| 6.2.3 Reducing underestimation risk in the PLMF |       |         |   |
| 6.2.4 Standard codes                            |       |         |   |
| 6.3 Survey weights                              |       |         |   |

|        |    |                                                                                                                                                                   |                                                                                                            |
|--------|----|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------|
|        |    | Unknown: if not selected, was assigned to the perturbation to maintain privacy; more information on how these weights were generated can be found in section 6.3. |                                                                                                            |
| SH_010 | 10 | In general, would you say your physical health is excellent, very good, good, fair or poor?                                                                       | 1 = Excellent<br>2 = Very good<br>3 = Good<br>4 = Fair<br>5 = Poor<br>6 = I do not know<br>99 = Not stated |
|        |    |                                                                                                                                                                   | 1 = Excellent<br>2 = Very good                                                                             |

# PHAC Health Infobase

The Health Infobase is a comprehensive digital platform maintained by the Public Health Agency of Canada (PHAC). The Infobase provides access to a wide range of health-related data tools, infographics, charts and blogs. There are 182 data sources available on the Infobase. The Health Infobase serves as a central hub for national surveillance data, interactive data tools, and public health reports. The platform is frequently updated with the latest surveillance data.



[PHAC Health Infobase](#)

## Data Categories

The Infobase provides access to health data under six categories:

- **Diseases and disorders:** Comprehensive tracking of chronic conditions such as cancer, cardiovascular diseases, and COVID-19.
- **Drugs and vaccines:** Data on prescription drugs, vaccine coverage, and immunization rates across the country.

- **Mental health, healthy living and environment:** Information on mental health indicators, physical activity levels, nutrition, and environmental factors such as wildfires.
- **Injuries and death:** Data on death, suicide, violence, and drowning across regions.
- **Population groups and demography:** Data on birth, maternity, children, family, and disabilities.
- **Substance use:** Data on substance use such as alcohol, cannabis use, opioids, and smoking.

Under each category, users can select the data for specific health topics. The three top categories are **Respiratory diseases**, **Opioids** and **Mental health**.

### Find a data product

Use filters to find specific data products. You can filter by **category**, by **format**, or by typing in **keywords**.

The screenshot shows the 'Find a data product' interface. On the left, a sidebar titled 'Categories' lists several categories with expandable icons: Diseases and disorders, Drugs and vaccines, Mental health, healthy living and environment, Injuries and death, Population groups and demography, and Substance use. The main content area features a search bar with a 'Filter' button and a text input. Below the search bar, a data product titled 'Healthy Canadians and Communities Fund' is displayed, featuring a heart icon with a plus sign. The product description mentions an interactive map of funded interventions across Canada and includes a 'Data exploration tool' link. The last updated date is 2026-04-07.

### Find a data product

Use filters to find specific data products. You can filter by **category**, by **format**, or by typing in **keywords**.

This screenshot shows the 'Find a data product' interface with filters applied. In the 'Categories' sidebar, 'Diseases and disorders' is expanded, and 'Blood and metabolic diseases' is selected, indicated by a red box around the checkbox. The main content area shows a search bar and a filter button labeled 'Blood and metabolic diseases X'. Below the search bar, it states 'Showing 5 filtered from 182 total products' and provides a 'Clear all filters' link. The data product displayed is the 'Canadian Chronic Disease Surveillance System (CCDSS)', featuring a heart icon with a plus sign. It includes a 'Data exploration tool' link and a list of related topics: Chronic diseases, Cardiovascular diseases, Musculoskeletal disorders, Neurological diseases, Respiratory diseases, Mental health, Blood and metabolic diseases, and Autism.

You can filter the data by category, by format, or by typing in keywords.

## Find a data product

Use filters to find specific data products. You can filter by **category**, by **format**, or by typing in **keywords**.

▼ Categories

Diseases and disorders

+

Drugs and vaccines

+

Mental health, healthy living and environment

+

Injuries and death

+

Population groups and demography

+

Substance use

+

Filter covid

Showing 27 filtered from 182 total products

[Clear all filters](#)



[COVID-19 Vaccine mistrust among Black Communities in Canada](#)

Interactive report

COVID-19

Vaccines

Interactive report on COVID-19 vaccine mistrust among Black communities in Canada, based on data from the 2022 CanBlack Vax survey.

You can also filter the data by type, such as blog, infographic, or report.

▼ Type

☐ Blog

☐ Data exploration tool

☐ Indicator framework

☐ Infographic

☐ Interactive report

at greater risk of developing chronic disease.

Last updated: 2026-04-07



[Transfusion and transplantation-related adverse events](#)

Interactive report

Injuries

Summary of national surveillance systems for transfusion errors, transfusion-transmitted injuries, and cells, tissues and

## Data Availability

The level of information provided in the data sources varies. For some data sources/products, you can find blogs, infographics, and reports along with individual-level data. However, for some data sources/products, you may only find the summary statistics without individual-level data, i.e., only aggregated-level data. For example, the **Human Papillomavirus Indicator Framework** product provides quick statistics, data tools and a list of publications:

▼ Categories

Diseases and disorders

—

☐ All diseases and disorders

☐ Blood and metabolic diseases

☐ Chronic diseases

☒ Cancer

☐ COVID-19

☐ Cardiovascular diseases

☐ Developmental

Filter Type in keywords to filter results as you type

Cancer X

Showing 10 filtered from 182 total products

[Clear all filters](#)



[Human Papillomavirus Indicator Framework \(HPVIF\)](#)

Data exploration tool

Infections

Vaccines

Cancer

The Human Papillomavirus Indicator Framework (HPVIF) presents HPV vaccination coverage and HPV outcome indicators for Canada.

Last updated: 2025-12-15

612



The **Drug and alcohol use in Canada: Postsecondary students** product provides summary statistics as well as links to download data in CSV format:

**Substance use**

- ☐ All substance use
- ☒ Alcohol
- ☐ Cannabis
- ☐ Opioids
- ☐ Smoking and vaping
- ☐ Stimulants



### Drug and alcohol use in Canada: Postsecondary students

Data exploration tool
Alcohol
Cannabis

Opioids
Prescription drugs
Smoking and vaping

Stimulants

Explore Health Canada's latest data on drug and alcohol use by postsecondary students in Canada.

613

# CCDSS

The Canadian Chronic Disease Surveillance System (CCDSS) is a collaborative network of provincial and territorial chronic disease surveillance systems in Canada. The CCDSS is supported by the Public Health Agency of Canada (PHAC) and provincial/territorial governments. The CCDSS was established to provide the estimates of the prevalence and incidence of chronic diseases in Canada, as well as to monitor trends over time for more than 20 chronic diseases to help inform health policies and resource planning. The CCDSS is part of the PHAC Health Infobase that we covered in the previous tutorial on [Health Infobase](#). The CCDSS uses health administrative data, such as hospital discharge records, physician billing claims, and prescription drug data.



[CCDSS website](#)

## Key Indicator Categories

The CCDSS website provides a variety of indicators related to chronic diseases, such as autism, cardiovascular diseases,

chronic respiratory diseases, diabetes, mental health disorders, musculoskeletal disorders, neurological disorders, and multimorbidity.

Diseases, conditions and indicators

Table 1: Diseases, conditions and indicators included in the CCDSS

| Morbidity and mortality                                                                                                                                                                        | Health events and complications                                                                                        | Use of health services                                            |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| <ul style="list-style-type: none"><li>incidence/rate of newly identified cases</li><li>prevalence (cumulative and active for select diseases/conditions)</li><li>all-cause mortality</li></ul> | <ul style="list-style-type: none"><li>annual prevalence</li><li>all-cause mortality following a health event</li></ul> | <ul style="list-style-type: none"><li>annual prevalence</li></ul> |

Table 2: CCDSS case definitions and surveillance period reported

| Disease, condition and indicator | Age  | Case definition summary                                                  | First reported year | Last reported year |
|----------------------------------|------|--------------------------------------------------------------------------|---------------------|--------------------|
| <b>Autism</b>                    |      |                                                                          |                     |                    |
| Autism                           | 1-19 | One or more hospital separation records, or two or more physician claims | 2000-2001           | 2023-2024          |
| <b>Cardiovascular diseases</b>   |      |                                                                          |                     |                    |
| Acute myocardial infarction      | 20+  | One or more hospital inpatient admission records                         | 2000-2001           | 2023-2024          |

Data Tool

The CCDSS website provides a data tool that allows users to explore and visualize the data on chronic diseases in Canada. The data tool provides interactive charts and tables that allow users to explore the data by province/territory.

About the Canadian Chronic Disease Surveillance System (CCDSS)

The CCDSS is a collaborative network of provincial and territorial surveillance systems, supported by the Public Health Agency of Canada (PHAC). The system collects data on all residents who are eligible for provincial or territorial health insurance. It can generate national estimates and trends over time for over 20 chronic diseases and conditions, and other selected health outcomes. To identify people with chronic diseases and conditions, provincial and territorial health insurance registry records are linked using a unique personal identifier to the corresponding physician claims, hospital separation records and prescription drug records.

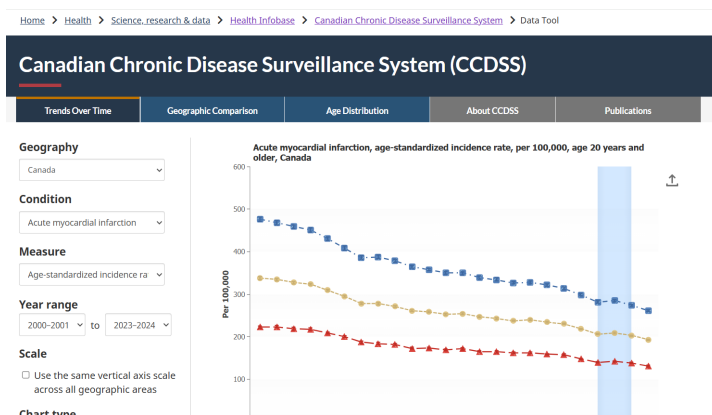
Key information about the CCDSS methods and case definitions is presented below. More detailed documents are available here:

[CCDSS summary of methods](#); [CCDSS case definitions](#); [CCDSS COVID-19 guidance for data users](#).

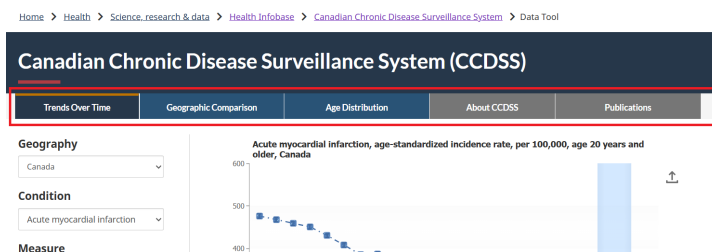
Use the CCDSS Data Tool

Users can use the data tool to explore the prevalence and incidence of cases, mortality rates, and health service utilization, and to track individuals living with two or more chronic conditions.

[CCDSS Data Tool](#)

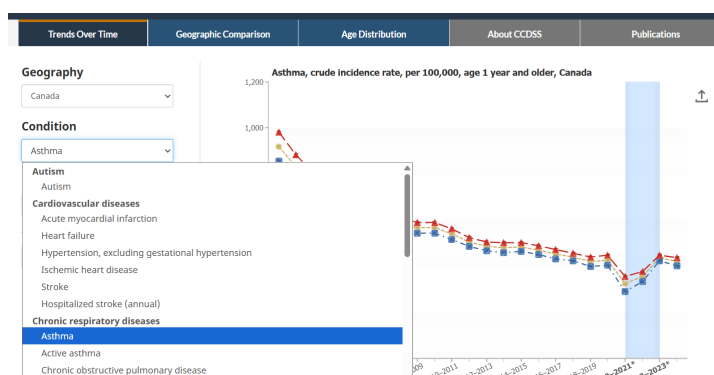


Users can explore trends over time, compare data across provinces/territories, and view the distribution of cases by age group. Under the **Publication** tab, you can find information on the repository for data-driven insights, fact sheets, peer-reviewed publications, and reports.



## Interactive chart example

Consider **Asthma** under **Chronic respiratory diseases** as an example.



You can explore different summary measures by filtering by geography, measure, and year. Again, you can further explore trends over time, compare data across provinces/territories, and view the distribution of cases by age group.



You can also download the summary data in CSV format for further analysis. To do that, you need to scroll down to the bottom of the **Summary Table**, and click on the **Download** button.

|       |            |     |      |         |      |        |
|-------|------------|-----|------|---------|------|--------|
| Males | 2019-2020  | 404 | 1.54 | 401-407 | 0.38 | 68,605 |
| Males | 2020-2021* | 292 | 1.31 | 290-295 | 0.45 | 49,690 |
| Males | 2021-2022* | 333 | 1.39 | 331-336 | 0.42 | 57,745 |
| Males | 2022-2023* | 425 | 1.55 | 422-428 | 0.36 | 75,625 |
| Males | 2023-2024  | 402 | 1.48 | 399-405 | 0.37 | 73,400 |

Download Detailed Table: Asthma, crude incidence rate

Save Summary Table

Once you open the CSV file in Excel or other software, you will find that the data is not at the individual or person

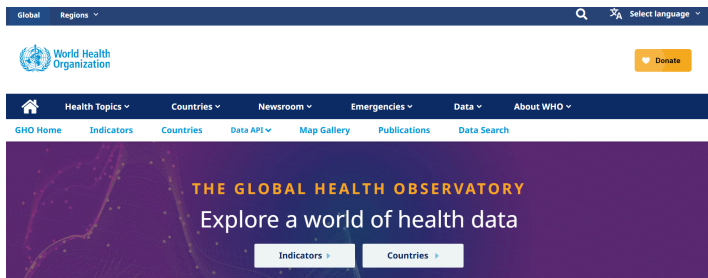
level. Instead, the data are aggregated at the country and province/territory levels, and summary measures are provided for both sexes and for years. For example, see row 4. The **Geography** column says *Canada* and the **Rate (per 100,000)** column says *919*. This is an example of aggregate data.

|    | A                                                     | B          | C              | D         | E           | F                  | G              | H            | I            | J           | K           | L          |
|----|-------------------------------------------------------|------------|----------------|-----------|-------------|--------------------|----------------|--------------|--------------|-------------|-------------|------------|
| 1  | Data Table: Asthma, Crude incidence rate, per 100,000 |            |                |           |             |                    |                |              |              |             |             |            |
| 2  |                                                       |            |                |           |             |                    |                |              |              |             |             |            |
| 3  | Geography                                             | Sex        | Age group type | Age group | Fiscal year | Rate (per 100,000) | Standard Error | Lower 95% CI | Upper 95% CI | Rate CV (%) | Case counts | Population |
| 4  | Canada                                                | Both sexes | Total          | 1+        | 2000-2001   | 919                | 1.8            | 916          | 923          | 0.2         | 260730      | 28367690   |
| 5  | Canada                                                | Females    | Total          | 1+        | 2000-2001   | 961                | 2.6206         | 976          | 986          | 0.27        | 140225      | 14289240   |
| 6  | Canada                                                | Males      | Total          | 1+        | 2000-2001   | 856                | 2.4657         | 851          | 861          | 0.29        | 120505      | 14078655   |
| 7  | Canada                                                | Both sexes | Total          | 1+        | 2001-2002   | 833                | 1.7103         | 830          | 837          | 0.21        | 237405      | 28488200   |
| 8  | Canada                                                | Females    | Total          | 1+        | 2001-2002   | 883                | 2.4814         | 878          | 886          | 0.26        | 126365      | 14337965   |
| 9  | Canada                                                | Males      | Total          | 1+        | 2001-2002   | 763                | 2.3526         | 779          | 788          | 0.3         | 110820      | 14150240   |
| 10 | Canada                                                | Both sexes | Total          | 1+        | 2002-2003   | 762                | 1.6318         | 759          | 765          | 0.21        | 218155      | 28623720   |
| 11 | Canada                                                | Females    | Total          | 1+        | 2002-2003   | 804                | 2.3637         | 800          | 809          | 0.29        | 115840      | 14399350   |
| 12 | Canada                                                | Males      | Total          | 1+        | 2002-2003   | 719                | 2.2487         | 715          | 724          | 0.31        | 102310      | 14224360   |
| 13 | Canada                                                | Both sexes | Total          | 1+        | 2003-2004   | 727                | 1.5893         | 724          | 730          | 0.22        | 209140      | 28774260   |
| 14 | Canada                                                | Females    | Total          | 1+        | 2003-2004   | 773                | 2.3117         | 769          | 778          | 0.3         | 111860      | 14468175   |
| 15 | Canada                                                | Males      | Total          | 1+        | 2003-2004   | 680                | 2.1801         | 676          | 684          | 0.32        | 97275       | 14306080   |
| 16 | Canada                                                | Both sexes | Total          | 1+        | 2004-2005   | 696                | 1.5513         | 693          | 699          | 0.22        | 201445      | 28932495   |
| 17 | Canada                                                | Females    | Total          | 1+        | 2004-2005   | 738                | 2.2531         | 734          | 743          | 0.31        | 107350      | 14541550   |
| 18 | Canada                                                | Males      | Total          | 1+        | 2004-2005   | 654                | 2.1316         | 650          | 658          | 0.33        | 94095       | 14398940   |

Because this data is not at the person-level, and the summary measures are aggregated, you do not need a codebook to translate columns. However, you cannot use this data to calculate the prevalence or incidence of asthma in Canada because it is already aggregated. You can only use this data to explore the trends over time, compare data across provinces/territories, and view the distribution of cases by age group and sex. The individual patients have already been erased to protect their privacy.

# GHO

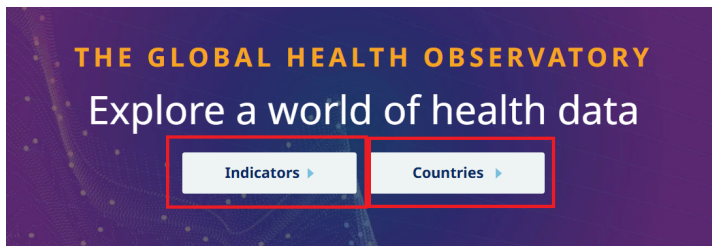
The Global Health Observatory (GHO) is the World Health Organization's (WHO) main health statistics repository. The GHO provides data on a wide range of health topics, such as mortality, morbidity, risk factors, health systems, and health equity. The GHO collects data from member states, international organizations, and other sources to provide a comprehensive picture of global health. The GHO is a valuable resource for researchers, policymakers, and public health professionals who are interested in understanding global health trends and disparities. The GHO provides various ways to interact with its data, including interactive maps and downloadable CSV files.



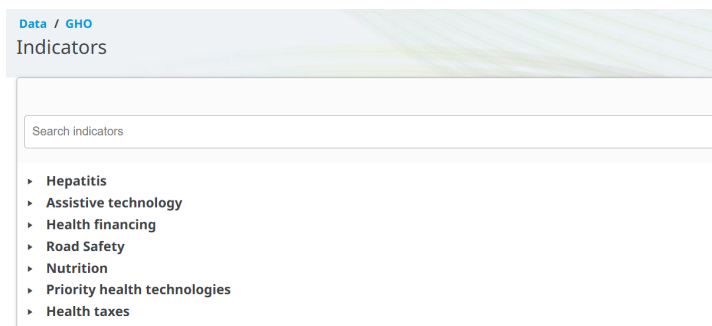
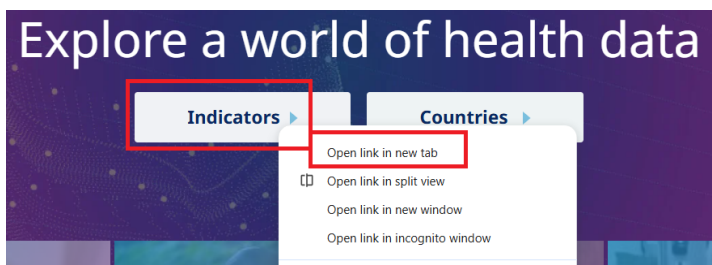
[WHO Global Health Observatory](#)

## World Health Data

You can browse information by broader health indicators or by specific countries.



To explore the data by indicators, click on the **Indicators** tab. You can find a list of indicators across categories such as Hepatitis, Nutrition, Mental health, and so on.



You can also use the search bar to search for specific indicators. For example, if you search for **Life expectancy at birth**, you will find a list of indicators related to life expectancy, such as **Life expectancy at birth (years)**.



Data / GHO

## Indicators

Life expectancy at birth

- World Health Statistics
  - Life expectancy and life tables
    - Life expectancy at birth (years)
- World Health Statistics
  - Life expectancy at birth (years)
- Global Health Estimates: Life expectancy and leading causes of death and disability
  - Mortality and global health estimates
    - Life expectancy at birth (years)

To explore the data by country, click on the **Countries** tab. You can find a list of countries and regions. You can click the country name to see the data for that country.

World Health Organization

**Data**

WHO Glob

Indicators Countries **Dasht**

|     |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|-----|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| All | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T |
| U   | V | W | X | Y | Z |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |

**A**

Afghanistan  
Albania  
Algeria  
Andorra  
Angola  
Antigua and Barbuda  
Argentina  
Armenia  
Australia  
Austria  
Azerbaijan

**B**

Bahamas  
Bahrain  
Bangladesh  
Barbados

 **Australia**  
Health data overview for Australia

|                                       |                   |
|---------------------------------------|-------------------|
| Population                            | 26 451 124 (2023) |
| Current health expenditure (% of GDP) | 10.54 (2021)      |
| WHO region                            | Western Pacific   |
| World Bank income level               | High income (HIC) |



### Population, Australia

In Australia, the current population is 26,451,124 as of 2023 with a projected increase of 23% to 32,506,969 by 2050.

**Population growth rate**  
Australia, 2023

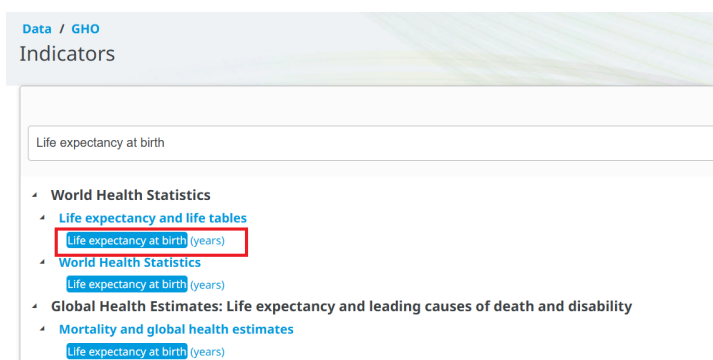
**0.95%** +0.012 percentage points  
rate change since 2022

On this page

- Population**
- Life expectancy
- Leading causes of death
- Health statistics

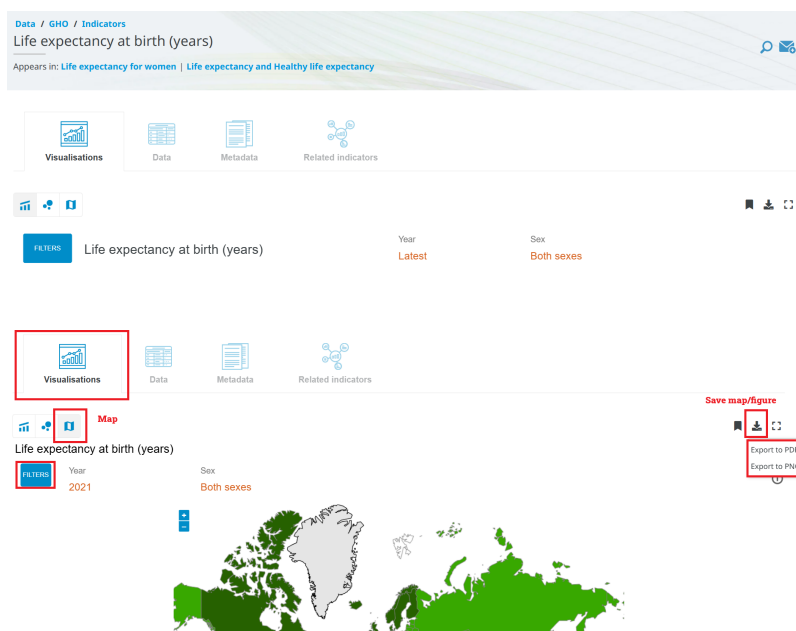
## Data by indicator example

Let's click on the Life expectancy at birth (years) indicator to see the details:



You can find different visualization options, such as charts and maps. You can also download the chart and map to your computer.

[GHO Indicators](#)



 Visualisations
  **Data**
 Metadata
  Related indicators

Similar to the [Canadian Chronic Disease Surveillance System \(CCDSS\)](#), [14](#), [15](#), the life expectancy estimation is based on

The data on the website, as well as the downloaded data in CSV

 Visualisations
  **Data**
 Metadata
  Related indicators

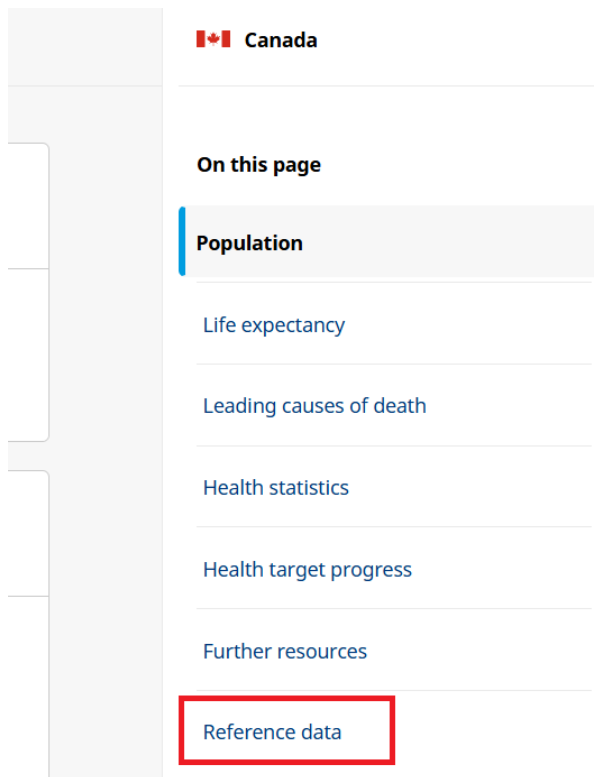
## Data by country example

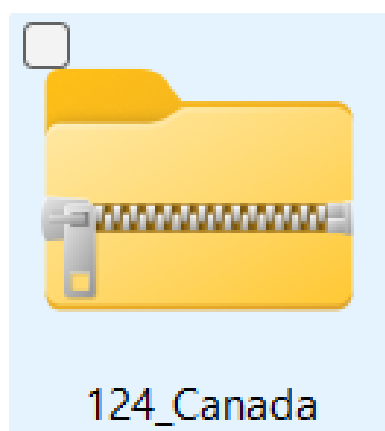
Let's click on **Canada** to explore the data for Canada.



You can explore different indicators for Canada. You can also download all indicators as a ZIP file by clicking **Reference data**.

[GHO Data by Country](#)





Once you unzip the folder, you will find several folders. Each folder contains a codelist, data, a data dictionary, along with a license and terms of use for different indicators.

|                                                                                         |                    |             |
|-----------------------------------------------------------------------------------------|--------------------|-------------|
| 1F96863_3.4.1 - cardiovascular disease, cancer, diabetes or chronic respiratory disease | 03-Mar-26 10:33 AM | File folder |
| 2D6FBE4_3.3.5 - Neglected tropical diseases                                             | 03-Mar-26 10:33 AM | File folder |
| 5C8435F_3.C.1 - Health worker density and distribution                                  | 03-Mar-26 10:33 AM | File folder |
| 6A64C9A_7.1.2 - Clean fuels                                                             | 03-Mar-26 10:33 AM | File folder |
| 12EE54A_6.2.1 - Safely managed sanitation & hand-washing                                | 03-Mar-26 10:33 AM | File folder |
| 168BF41_3.4.2 - Suicide                                                                 | 03-Mar-26 10:33 AM | File folder |
| 45CA7C8_3.C.1 - Health worker density and distribution                                  | 03-Mar-26 10:33 AM | File folder |
| 75DDA77_3.A.1 - Age-standardized prevalence of tobacco use                              | 03-Mar-26 10:33 AM | File folder |
| 77D059C_3.3.1 - HIV infections                                                          | 03-Mar-26 10:33 AM | File folder |
| 84FD3DE_3.9.3 - Unintentional poisoning                                                 | 03-Mar-26 10:33 AM | File folder |
| 608DE39                                                                                 | 03-Mar-26 10:33 AM | File folder |
| 1548EA3_6.1.1 - Safely managed drinking water                                           | 03-Mar-26 10:33 AM | File folder |
| 217795A_3.C.1 - Health worker density and distribution                                  | 03-Mar-26 10:33 AM | File folder |
| 361734E_16.1.1 - Intentional homicide                                                   | 03-Mar-26 10:33 AM | File folder |
| 1772666_3.1.2 - Births attended by skilled health personnel                             | 03-Mar-26 10:33 AM | File folder |

For example, the 77D059C\_3.3.1 - HIV infections folder contains the following files:

|                                                                                                                                 |                    |
|---------------------------------------------------------------------------------------------------------------------------------|--------------------|
|  124_77D059C_Code list_2025-03-26              | 13-Apr-26 11:43 PM |
|  124_77D059C_Dataset_2025-03-26                | 13-Apr-26 11:43 PM |
|  124_77D059C_License & Terms of Use_2025-03-26 | 13-Apr-26 11:43 PM |
|  124_77D059C_Metadata_2025-03-26               | 13-Apr-26 11:43 PM |
|  DDS - Data Dictionary 2024-12-12              | 13-Apr-26 11:43 PM |

Metadata

The GHO provides metadata for each indicator, providing important context for understanding the data and its limitations. For example, the metadata for the **Hepatitis - new infections** indicator includes information on the definition of the indicator, the methods used to calculate the indicator, data sources, and other relevant information.

[Indicator Metadata Registry List](#)

Data / GHO

Indicators

Search indicators

Hepatitis

Hepatitis

Hepatitis - new infections

Hepatitis - number of persons living with chronic hepatitis

Hepatitis - deaths caused by chronic hepatitis, number

Hepatitis - prevalence of chronic hepatitis among the general population

Hepatitis - deaths caused by chronic hepatitis B and C (per 100 000)

Data / GHO / Indicators

Hepatitis - new infections

Appears in: [Hepatitis cases and deaths](#)

Data

Metadata

Related indicators

Indicator name:

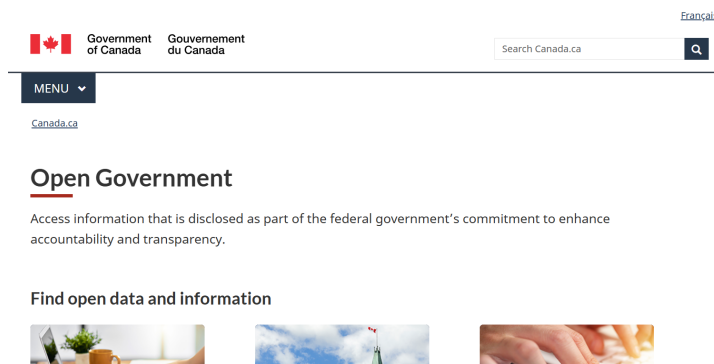
Hepatitis - (chronic HBV and HCV) incidence

Short name:

Chronic hepatitis incidence number

# Open Portal

Open Government Portal Canada is a platform that provides access to searching for a wide range of public data. Users can explore and download data in different formats under the [Open Government Licence - Canada](#).

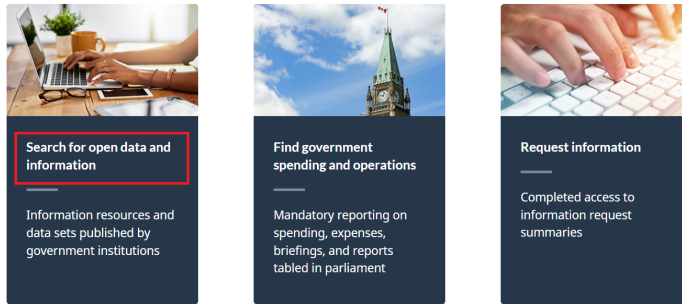


[Open Government Portal Canada](#)

## Search for Open Data

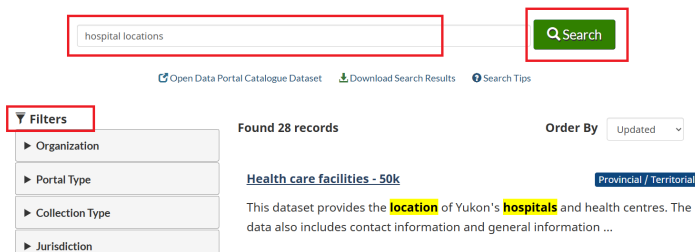
To search for open data, you can use the search bar on the Open Government Portal Canada homepage. For example, if you are interested in finding data on “hospital locations”, you can enter that term in the search bar and click the search icon. There are many filter options to narrow your search results, such as format and organization.

## Find open data and information



[Canada.ca](#) > [Open Government](#)

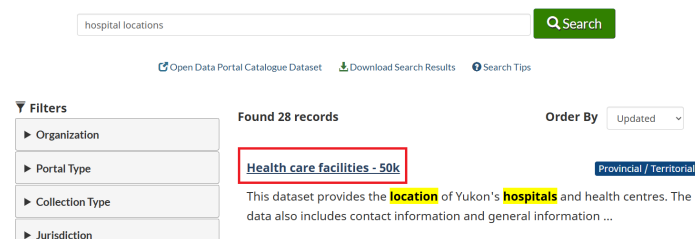
## Open Government Portal



To explore a specific dataset, click its title. This will take you to a page with more information about the dataset, including its description, format options, and download links. For example, if you click on the “Health care facilities - 50k” dataset, you will find detailed information about the location of Yukon’s hospitals and health centres.

[Canada.ca](#) > [Open Government](#)

## Open Government Portal





## Health care facilities - 50k

### Have your say

★★★★★

[Rate this dataset](#)  
[Provide feedback](#)

This dataset provides the location of Yukon's hospitals and health centres. The data also includes contact information and general information about each facility.

Distributed from GeoYukon by the Government of Yukon. Discover more digital map data and interactive maps from Yukon's digital map data collection. For more information: [geomatics.help@yukon.ca](mailto:geomatics.help@yukon.ca)

Under the **Data** and **Resources** tab, you can find the available formats for downloading the location of Yukon's hospitals and health centres.

### Data and Resources

View on Map

Search resources...

**ArcGIS Online layers**

English web service ESRI REST

Explore

**ArcGIS Online layers**

English web service ESRI REST

Explore

**Shape files**

English dataset HTML

Explore

**XML metadata**

English dataset HTML

Explore

**Original metadata (<https://open.yukon.ca>)**

English dataset HTML

Explore

Similarly, say we are interested in “opioid use” and want to find open data related to that topic. We can enter “opioid use” into the search bar and filter by **Open data**:

opioid use

Search

[Open Data Portal Catalogue Dataset](#)
[Download Search Results](#)
[Search Tips](#)

Filters

Portal Type - Open Data

Organization

Portal Type

☒ Open Data 10
 ☐ Open Information 34

Collection Type

Found 10 records

Order By Updated

**Opioid agonist treatment archives**

Federal

Correctional Service Canada provides Opiate Agonist Treatment to patients with an **opioid use** disorder. Archives of these quarterly web pages ...

Record Modified: Apr 9, 2026
 Record Released: Nov 28, 2025

Publisher: Correctional Service of Canada

Formats: [CSV](#) [PDF](#)

Keywords: CSC OAT OUD opiate treatment Suboxone Methadone Sublocade health drugs institutions

629

To explore the data on opioid agonist treatment, we can click the dataset title. This will take us to a page with more information about the dataset, including its description, format options, and download links.

opiod use Search

[Open Data Portal Catalogue Dataset](#) [Download Search Results](#) [Search Tips](#)

**Filters**

Portal Type: Open Data

Organization

Portal Type

☒ Open Data 10

☐ Open Information 34

Collection Type

**Found 10 records**

Order By Updated

**Opioid agonist treatment archives** Federal

Correctional Service Canada provides Opiate Agonist Treatment to patients with an **opiod use** disorder. Archives of these quarterly web pages ...

Record Modified: Apr 9, 2026 Record Released: Nov 28, 2025

Publisher: Correctional Service of Canada

Formats: [CSV](#) [PDF](#)

Keywords: CSC OAT OUD opiate treatment Suboxone Methadone Sublocade health drugs institutions

## Data and Resources

Search resources... Search

**Opioid Agonist Treatment (OAT) in CSC: July 2019** Explore

English dataset CSV

**Opioid Agonist Treatment (OAT) in CSC: July 2019** Explore

French dataset CSV

**Opioid Agonist Treatment: December 2019** Explore

English dataset CSV

**Opioid Agonist Treatment: December 2019** Explore

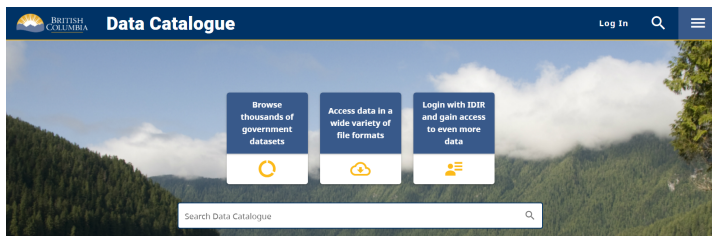
French dataset CSV

**Opioid Agonist Treatment: March 2020** Explore

English dataset CSV

# BC Data Catalogue

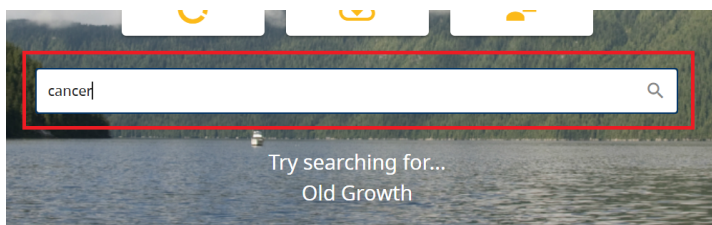
The British Columbia Data Catalogue is a central hub provided by the Government of British Columbia (BC). Its primary purpose is to provide public access to thousands of datasets managed by the provincial government and its partners. Essentially, it is a digital library designed to make government information transparent and usable for citizens, researchers, and developers.



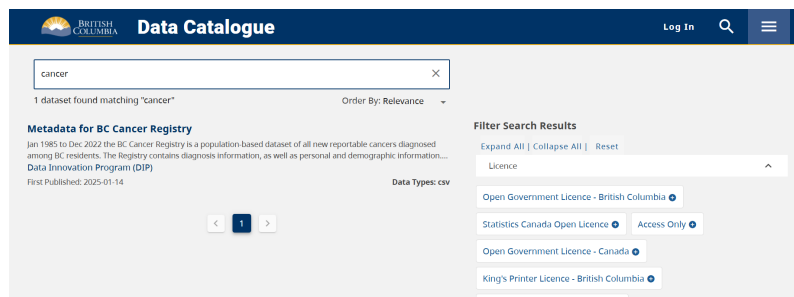
[British Columbia Data Catalogue](#)

## Search for Data

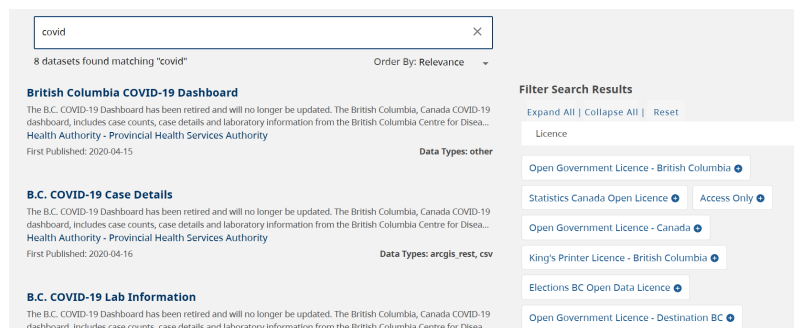
To search for data, you can use the search bar on the BC Data Catalogue homepage. For example, if you are interested in finding data on “cancer”, you can enter that term in the search bar and click the search icon.



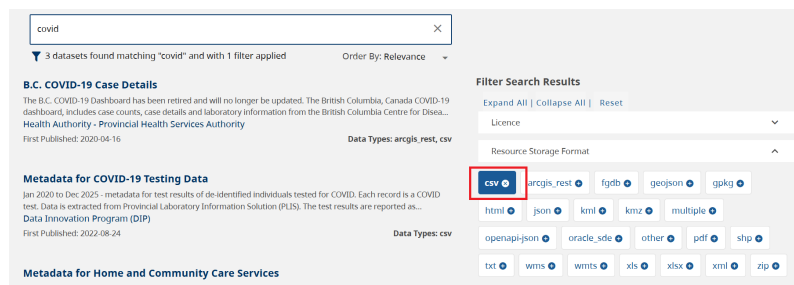
[Search Data Catalogue](#)



Similarly, say we are interested in “COVID-19” and want to find open data related to that topic. We can enter “covid” into the search bar:



You can filter your search by license, format, group, and so on:



Let’s explore the “COVID-19 Case Details” dataset. To do that, we can click the dataset title. This will take us to a page with more information about the dataset, including its description and related links.

Search results for "covid" (8 datasets found). The B.C. COVID-19 Dashboard is highlighted as a relevant dataset. The dashboard includes case counts, case details, and laboratory information from the British Columbia Centre for Disease Control, Provincial Health Services Authority. The data is published by the Health Authority - Provincial Health Services Authority. The dashboard is retired and will no longer be updated. The data types are: other.

**B.C. COVID-19 Case Details**

The B.C. COVID-19 Dashboard has been retired and will no longer be updated. The British Columbia, Canada COVID-19 dashboard, includes case counts, case details and laboratory information from the British Columbia Centre for Disease Control, Provincial Health Services Authority. The data is published by the Health Authority - Provincial Health Services Authority. The dashboard is retired and will no longer be updated. The data types are: arcgis\_rest, csv.

**B.C. COVID-19 Lab Information**

The B.C. COVID-19 Dashboard has been retired and will no longer be updated. The British Columbia, Canada COVID-19 dashboard, includes case counts, case details and laboratory information from the British Columbia Centre for Disease Control, Provincial Health Services Authority. The data is published by the Health Authority - Provincial Health Services Authority. The dashboard is retired and will no longer be updated. The data types are: other.

**Filter Search Results**

Expand All | Collapse All | Reset

Licence

Open Government Licence - British Columbia

Statistics Canada Open Licence

Open Government Licence - Canada

King's Printer Licence - British Columbia

Elections BC Open Data Licence

Open Government Licence - Destination BC

You will find information on COVID-19 case details as well as a link to download the data:

Navigation: Back to Datasets list | Scroll to Bottom | Groups | Contact Data Expert | Share

**B.C. COVID-19 Case Details** PUBLISHED

Published By: Health Authority - Provincial Health Services Authority

Description: The B.C. COVID-19 Dashboard has been retired and will no longer be updated. The British Columbia, Canada COVID-19 dashboard, includes case counts, case details and laboratory information from the British Columbia Centre for Disease Control, Provincial Health Services Authority and the British Columbia Ministry of Health. Data are represented by B.C. Health Authority region.

Terms of use, disclaimer and limitation of liability

The following data notes define the indicators presented on the public dashboard and describe the data sources involved. Data changes as new cases are identified, characteristics of reported cases change or as additional and data information is made. Please refer to the dashboard for more information.

**Data and Resources**

B.C. COVID-19 Case Details - Item Details

arcgis\_rest [Access/Download](#) [View](#)

B.C. COVID-19 Case Details - Service

arcgis\_rest [Access/Download](#) [View](#)

**BC COVID-19 Dashboard Case Details**

csv [Access/Download](#) [View](#)

To download the dataset, right-click on the **Access/Download** link and select **Save link as...**. This will allow you to save the dataset to your computer.

ida COVID-19 dashboard, ie Control, Provincial Health rity region.

sources involved. Data rrections are made. Specific deaths, testing and st caveats about the data,

ncluding the British inistry of Health makes no ted data, nor will it accent

**Data and Resources**

B.C. COVID-19 Case Details - Item Details

arcgis\_rest [Access/Download](#) [View](#)

B.C. COVID-19 Case Details - Service

arcgis\_rest [Access/Download](#) [View](#)

**BC COVID-19 Dashboard Case Details**

csv [Access/Download](#) [View](#)

Open link in new tab

Open link in split view

Open link in new window

Open link in incognito window

**Save link as...**

Copy link address

File name: BCCDC COVID19 Dashboard Case Details

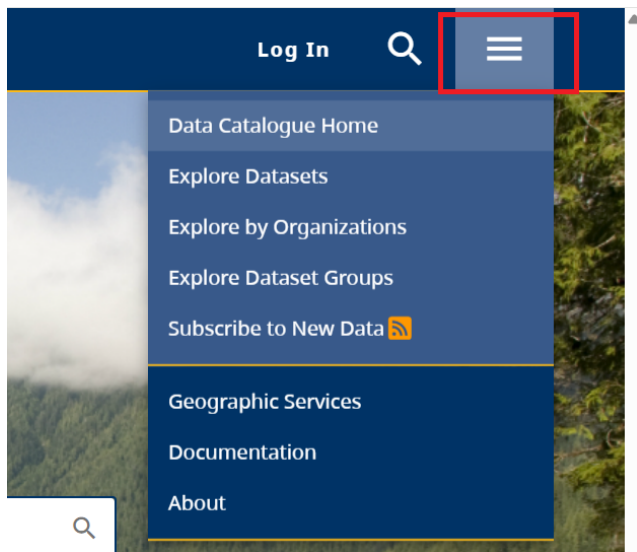
Save as type: Microsoft Excel Comma Separated Values File

Save

After downloading, you can open the data in Excel, Google Sheets, or any statistical software. The data is structured in a tabular format, with each row representing an individual case and each column representing different variables.

| 1  | Reported_Date | HA                | Sex | Age_Group | Classification_Reported |
|----|---------------|-------------------|-----|-----------|-------------------------|
| 2  | 29-01-20      | Out of Canada     | M   | 40-49     | Lab-diagnosed           |
| 3  | 06-02-20      | Vancouver Coastal | F   | 50-59     | Lab-diagnosed           |
| 4  | 10-02-20      | Out of Canada     | M   | 30-39     | Lab-diagnosed           |
| 5  | 10-02-20      | Out of Canada     | F   | 20-29     | Lab-diagnosed           |
| 6  | 18-02-20      | Interior          | F   | 30-39     | Lab-diagnosed           |
| 7  | 24-02-20      | Fraser            | F   | 30-39     | Lab-diagnosed           |
| 8  | 24-02-20      | Fraser            | M   | 40-49     | Lab-diagnosed           |
| 9  | 03-03-20      | Fraser            | M   | 50-59     | Lab-diagnosed           |
| 10 | 03-03-20      | Vancouver Coastal | F   | 60-69     | Lab-diagnosed           |
| 11 | 03-03-20      | Vancouver Coastal | F   | 30-39     | Lab-diagnosed           |
| 12 | 03-03-20      | Vancouver Coastal | F   | 80-89     | Lab-diagnosed           |
| 13 | 03-03-20      | Vancouver Coastal | M   | 60-69     | Lab-diagnosed           |
| 14 | 05-03-20      | Out of Canada     | F   | 50-59     | Lab-diagnosed           |
| 15 | 05-03-20      | Vancouver Coastal | M   | 60-69     | Lab-diagnosed           |
| 16 | 05-03-20      | Fraser            | F   | 50-59     | Lab-diagnosed           |
| 17 | 05-03-20      | Vancouver Coastal | M   | 30-39     | Lab-diagnosed           |

To learn more about BC Data Catalogue, click on the three bars on the right side of the page:



Centers for Disease Control and Prevention, National Center for Health Statistics. 2026. *National Health and Nutrition Examination Survey*. <https://www.cdc.gov/nchs/nhanes/>.