ICMA D2SC Notebook

Vidushi Kataria

2024-10-30

Table of Contents

# ICMA Sept 9 2024

plot(cars)



This plot states that the longer the distance of the car, the higher the speed is used.

# ICMA Sept 11 2024

This is where we put plain text.

ohno\_this\_is\_a\_nightmare <- "hello"

2+2

## [1] 4

x <- 2+2

my\_vector <- c(2,3,4)

mean(my\_vector)

## [1] 3

help(sd)  
sd(1:2) ^ 2

## [1] 0.5

# ICMA Sept 16 2024

## Warning: package 'ggplot2' was built under R version 4.3.1

## Warning: package 'tidyr' was built under R version 4.3.1

## Warning: package 'dplyr' was built under R version 4.3.1

## Warning: package 'lubridate' was built under R version 4.3.1

Wide format is NOT tidy. All tidy data will be in long form.

The pipe (%>%) is an operator that takes the outcome of the thing before it and passes it, as an argument, to the next function.

For example…

billboard %>% # take this dataset AND THEN...  
 select(artist, date.entered) %>% # select these two columns AND THEN...  
 filter(artist < "a") %>% # filter the dataset based on this value  
 head(n=3) # only show first 3 rows

## # A tibble: 3 × 2  
## artist date.entered  
## <chr> <date>   
## 1 2 Pac 2000-02-26   
## 2 2Ge+her 2000-09-02   
## 3 3 Doors Down 2000-04-08

ICMA: the pipe

# not using pipe  
my\_numbers <-c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)   
mean(my\_numbers)

## [1] 5.5

# output appears in source window and in the console

# using pipe  
mean\_my\_numbers <- my\_numbers %>%  
 mean()  
  
# output appears in the environment window under values  
# output is the same-- 5.5

Tidyverse core packages include: readr: reading & writing data tidyr: tidying data dplyr and purr: manipulating and transforming data lubridate: working with dates stringr: working with strings forcats: working with factors ggplot2: the “grammar of graphics” for creating data visualizations

getwd() #finding my working directory

## [1] "/Users/vidushikataria/Downloads/CogSci412-Data2SciComm/ICMA"

haven for SPSS, Stata, SAS readxl for Microsoft Excel files readr for csv, tsv, txt, and other types of deliminated files

?mtcars

write\_csv(mtcars, "mtcars\_fromR.csv")  
  
read\_csv("mtcars\_fromR.csv")

## Rows: 32 Columns: 11  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## dbl (11): mpg, cyl, disp, hp, drat, wt, qsec, vs, am, gear, carb  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

## # A tibble: 32 × 11  
## mpg cyl disp hp drat wt qsec vs am gear carb  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 21 6 160 110 3.9 2.62 16.5 0 1 4 4  
## 2 21 6 160 110 3.9 2.88 17.0 0 1 4 4  
## 3 22.8 4 108 93 3.85 2.32 18.6 1 1 4 1  
## 4 21.4 6 258 110 3.08 3.22 19.4 1 0 3 1  
## 5 18.7 8 360 175 3.15 3.44 17.0 0 0 3 2  
## 6 18.1 6 225 105 2.76 3.46 20.2 1 0 3 1  
## 7 14.3 8 360 245 3.21 3.57 15.8 0 0 3 4  
## 8 24.4 4 147. 62 3.69 3.19 20 1 0 4 2  
## 9 22.8 4 141. 95 3.92 3.15 22.9 1 0 4 2  
## 10 19.2 6 168. 123 3.92 3.44 18.3 1 0 4 4  
## # ℹ 22 more rows

glimpse(mtcars)

## Rows: 32  
## Columns: 11  
## $ mpg <dbl> 21.0, 21.0, 22.8, 21.4, 18.7, 18.1, 14.3, 24.4, 22.8, 19.2, 17.8,…  
## $ cyl <dbl> 6, 6, 4, 6, 8, 6, 8, 4, 4, 6, 6, 8, 8, 8, 8, 8, 8, 4, 4, 4, 4, 8,…  
## $ disp <dbl> 160.0, 160.0, 108.0, 258.0, 360.0, 225.0, 360.0, 146.7, 140.8, 16…  
## $ hp <dbl> 110, 110, 93, 110, 175, 105, 245, 62, 95, 123, 123, 180, 180, 180…  
## $ drat <dbl> 3.90, 3.90, 3.85, 3.08, 3.15, 2.76, 3.21, 3.69, 3.92, 3.92, 3.92,…  
## $ wt <dbl> 2.620, 2.875, 2.320, 3.215, 3.440, 3.460, 3.570, 3.190, 3.150, 3.…  
## $ qsec <dbl> 16.46, 17.02, 18.61, 19.44, 17.02, 20.22, 15.84, 20.00, 22.90, 18…  
## $ vs <dbl> 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0,…  
## $ am <dbl> 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0,…  
## $ gear <dbl> 4, 4, 4, 3, 3, 3, 3, 4, 4, 4, 4, 3, 3, 3, 3, 3, 3, 4, 4, 4, 3, 3,…  
## $ carb <dbl> 4, 4, 1, 1, 2, 1, 4, 2, 2, 4, 4, 3, 3, 3, 4, 4, 4, 1, 2, 1, 1, 2,…

* The number of rows and columns are 32 x 11. This is what I expected after using the glimpse function, which told me the data frame has 32 observations on 11 (numeric) variables.

Checking column names and class types:

colnames(mtcars)

## [1] "mpg" "cyl" "disp" "hp" "drat" "wt" "qsec" "vs" "am" "gear"  
## [11] "carb"

sapply(mtcars, class)

## mpg cyl disp hp drat wt qsec vs   
## "numeric" "numeric" "numeric" "numeric" "numeric" "numeric" "numeric" "numeric"   
## am gear carb   
## "numeric" "numeric" "numeric"

The column names are as I expect. The help window explaining the format of the data frame explains each column. I do not need to change the labels as it accurately conveys each variable. The classes for each column are numeric, which is accurate in regards to the dataset and what it is trying to convey.

There are now empty rows and columns just by scanning through the dataset, so it is already clean.

library(readr)  
mtcars <- read\_csv("mtcars\_fromR.csv", na = (c("", "NA", "-999")))

## Rows: 32 Columns: 11  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## dbl (11): mpg, cyl, disp, hp, drat, wt, qsec, vs, am, gear, carb  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

mtcars %>%  
 group\_by(cyl) %>%  
 count()

## # A tibble: 3 × 2  
## # Groups: cyl [3]  
## cyl n  
## <dbl> <int>  
## 1 4 11  
## 2 6 7  
## 3 8 14

# ICMA Sept 18th 2024

?ChickWeight  
glimpse(ChickWeight)

## Rows: 578  
## Columns: 4  
## $ weight <dbl> 42, 51, 59, 64, 76, 93, 106, 125, 149, 171, 199, 205, 40, 49, 5…  
## $ Time <dbl> 0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 21, 0, 2, 4, 6, 8, 10, 1…  
## $ Chick <ord> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, …  
## $ Diet <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, …

chick\_clean <- ChickWeight %>%  
 select(chick\_id = Chick,  
 diet = Diet,  
 age\_days = Time,  
 weight\_gm = weight)

mutate() also has it’s own if/then type capability that can be useful: case\_when()

chick\_clean %>%  
 mutate(session = case\_when(age\_days == 0 ~ "birth",  
 age\_days == 20 ~ "last1",  
 age\_days ==21 ~ "last2",  
 TRUE ~as.character(age\_days)))

## chick\_id diet age\_days weight\_gm session  
## 1 1 1 0 42 birth  
## 2 1 1 2 51 2  
## 3 1 1 4 59 4  
## 4 1 1 6 64 6  
## 5 1 1 8 76 8  
## 6 1 1 10 93 10  
## 7 1 1 12 106 12  
## 8 1 1 14 125 14  
## 9 1 1 16 149 16  
## 10 1 1 18 171 18  
## 11 1 1 20 199 last1  
## 12 1 1 21 205 last2  
## 13 2 1 0 40 birth  
## 14 2 1 2 49 2  
## 15 2 1 4 58 4  
## 16 2 1 6 72 6  
## 17 2 1 8 84 8  
## 18 2 1 10 103 10  
## 19 2 1 12 122 12  
## 20 2 1 14 138 14  
## 21 2 1 16 162 16  
## 22 2 1 18 187 18  
## 23 2 1 20 209 last1  
## 24 2 1 21 215 last2  
## 25 3 1 0 43 birth  
## 26 3 1 2 39 2  
## 27 3 1 4 55 4  
## 28 3 1 6 67 6  
## 29 3 1 8 84 8  
## 30 3 1 10 99 10  
## 31 3 1 12 115 12  
## 32 3 1 14 138 14  
## 33 3 1 16 163 16  
## 34 3 1 18 187 18  
## 35 3 1 20 198 last1  
## 36 3 1 21 202 last2  
## 37 4 1 0 42 birth  
## 38 4 1 2 49 2  
## 39 4 1 4 56 4  
## 40 4 1 6 67 6  
## 41 4 1 8 74 8  
## 42 4 1 10 87 10  
## 43 4 1 12 102 12  
## 44 4 1 14 108 14  
## 45 4 1 16 136 16  
## 46 4 1 18 154 18  
## 47 4 1 20 160 last1  
## 48 4 1 21 157 last2  
## 49 5 1 0 41 birth  
## 50 5 1 2 42 2  
## 51 5 1 4 48 4  
## 52 5 1 6 60 6  
## 53 5 1 8 79 8  
## 54 5 1 10 106 10  
## 55 5 1 12 141 12  
## 56 5 1 14 164 14  
## 57 5 1 16 197 16  
## 58 5 1 18 199 18  
## 59 5 1 20 220 last1  
## 60 5 1 21 223 last2  
## 61 6 1 0 41 birth  
## 62 6 1 2 49 2  
## 63 6 1 4 59 4  
## 64 6 1 6 74 6  
## 65 6 1 8 97 8  
## 66 6 1 10 124 10  
## 67 6 1 12 141 12  
## 68 6 1 14 148 14  
## 69 6 1 16 155 16  
## 70 6 1 18 160 18  
## 71 6 1 20 160 last1  
## 72 6 1 21 157 last2  
## 73 7 1 0 41 birth  
## 74 7 1 2 49 2  
## 75 7 1 4 57 4  
## 76 7 1 6 71 6  
## 77 7 1 8 89 8  
## 78 7 1 10 112 10  
## 79 7 1 12 146 12  
## 80 7 1 14 174 14  
## 81 7 1 16 218 16  
## 82 7 1 18 250 18  
## 83 7 1 20 288 last1  
## 84 7 1 21 305 last2  
## 85 8 1 0 42 birth  
## 86 8 1 2 50 2  
## 87 8 1 4 61 4  
## 88 8 1 6 71 6  
## 89 8 1 8 84 8  
## 90 8 1 10 93 10  
## 91 8 1 12 110 12  
## 92 8 1 14 116 14  
## 93 8 1 16 126 16  
## 94 8 1 18 134 18  
## 95 8 1 20 125 last1  
## 96 9 1 0 42 birth  
## 97 9 1 2 51 2  
## 98 9 1 4 59 4  
## 99 9 1 6 68 6  
## 100 9 1 8 85 8  
## 101 9 1 10 96 10  
## 102 9 1 12 90 12  
## 103 9 1 14 92 14  
## 104 9 1 16 93 16  
## 105 9 1 18 100 18  
## 106 9 1 20 100 last1  
## 107 9 1 21 98 last2  
## 108 10 1 0 41 birth  
## 109 10 1 2 44 2  
## 110 10 1 4 52 4  
## 111 10 1 6 63 6  
## 112 10 1 8 74 8  
## 113 10 1 10 81 10  
## 114 10 1 12 89 12  
## 115 10 1 14 96 14  
## 116 10 1 16 101 16  
## 117 10 1 18 112 18  
## 118 10 1 20 120 last1  
## 119 10 1 21 124 last2  
## 120 11 1 0 43 birth  
## 121 11 1 2 51 2  
## 122 11 1 4 63 4  
## 123 11 1 6 84 6  
## 124 11 1 8 112 8  
## 125 11 1 10 139 10  
## 126 11 1 12 168 12  
## 127 11 1 14 177 14  
## 128 11 1 16 182 16  
## 129 11 1 18 184 18  
## 130 11 1 20 181 last1  
## 131 11 1 21 175 last2  
## 132 12 1 0 41 birth  
## 133 12 1 2 49 2  
## 134 12 1 4 56 4  
## 135 12 1 6 62 6  
## 136 12 1 8 72 8  
## 137 12 1 10 88 10  
## 138 12 1 12 119 12  
## 139 12 1 14 135 14  
## 140 12 1 16 162 16  
## 141 12 1 18 185 18  
## 142 12 1 20 195 last1  
## 143 12 1 21 205 last2  
## 144 13 1 0 41 birth  
## 145 13 1 2 48 2  
## 146 13 1 4 53 4  
## 147 13 1 6 60 6  
## 148 13 1 8 65 8  
## 149 13 1 10 67 10  
## 150 13 1 12 71 12  
## 151 13 1 14 70 14  
## 152 13 1 16 71 16  
## 153 13 1 18 81 18  
## 154 13 1 20 91 last1  
## 155 13 1 21 96 last2  
## 156 14 1 0 41 birth  
## 157 14 1 2 49 2  
## 158 14 1 4 62 4  
## 159 14 1 6 79 6  
## 160 14 1 8 101 8  
## 161 14 1 10 128 10  
## 162 14 1 12 164 12  
## 163 14 1 14 192 14  
## 164 14 1 16 227 16  
## 165 14 1 18 248 18  
## 166 14 1 20 259 last1  
## 167 14 1 21 266 last2  
## 168 15 1 0 41 birth  
## 169 15 1 2 49 2  
## 170 15 1 4 56 4  
## 171 15 1 6 64 6  
## 172 15 1 8 68 8  
## 173 15 1 10 68 10  
## 174 15 1 12 67 12  
## 175 15 1 14 68 14  
## 176 16 1 0 41 birth  
## 177 16 1 2 45 2  
## 178 16 1 4 49 4  
## 179 16 1 6 51 6  
## 180 16 1 8 57 8  
## 181 16 1 10 51 10  
## 182 16 1 12 54 12  
## 183 17 1 0 42 birth  
## 184 17 1 2 51 2  
## 185 17 1 4 61 4  
## 186 17 1 6 72 6  
## 187 17 1 8 83 8  
## 188 17 1 10 89 10  
## 189 17 1 12 98 12  
## 190 17 1 14 103 14  
## 191 17 1 16 113 16  
## 192 17 1 18 123 18  
## 193 17 1 20 133 last1  
## 194 17 1 21 142 last2  
## 195 18 1 0 39 birth  
## 196 18 1 2 35 2  
## 197 19 1 0 43 birth  
## 198 19 1 2 48 2  
## 199 19 1 4 55 4  
## 200 19 1 6 62 6  
## 201 19 1 8 65 8  
## 202 19 1 10 71 10  
## 203 19 1 12 82 12  
## 204 19 1 14 88 14  
## 205 19 1 16 106 16  
## 206 19 1 18 120 18  
## 207 19 1 20 144 last1  
## 208 19 1 21 157 last2  
## 209 20 1 0 41 birth  
## 210 20 1 2 47 2  
## 211 20 1 4 54 4  
## 212 20 1 6 58 6  
## 213 20 1 8 65 8  
## 214 20 1 10 73 10  
## 215 20 1 12 77 12  
## 216 20 1 14 89 14  
## 217 20 1 16 98 16  
## 218 20 1 18 107 18  
## 219 20 1 20 115 last1  
## 220 20 1 21 117 last2  
## 221 21 2 0 40 birth  
## 222 21 2 2 50 2  
## 223 21 2 4 62 4  
## 224 21 2 6 86 6  
## 225 21 2 8 125 8  
## 226 21 2 10 163 10  
## 227 21 2 12 217 12  
## 228 21 2 14 240 14  
## 229 21 2 16 275 16  
## 230 21 2 18 307 18  
## 231 21 2 20 318 last1  
## 232 21 2 21 331 last2  
## 233 22 2 0 41 birth  
## 234 22 2 2 55 2  
## 235 22 2 4 64 4  
## 236 22 2 6 77 6  
## 237 22 2 8 90 8  
## 238 22 2 10 95 10  
## 239 22 2 12 108 12  
## 240 22 2 14 111 14  
## 241 22 2 16 131 16  
## 242 22 2 18 148 18  
## 243 22 2 20 164 last1  
## 244 22 2 21 167 last2  
## 245 23 2 0 43 birth  
## 246 23 2 2 52 2  
## 247 23 2 4 61 4  
## 248 23 2 6 73 6  
## 249 23 2 8 90 8  
## 250 23 2 10 103 10  
## 251 23 2 12 127 12  
## 252 23 2 14 135 14  
## 253 23 2 16 145 16  
## 254 23 2 18 163 18  
## 255 23 2 20 170 last1  
## 256 23 2 21 175 last2  
## 257 24 2 0 42 birth  
## 258 24 2 2 52 2  
## 259 24 2 4 58 4  
## 260 24 2 6 74 6  
## 261 24 2 8 66 8  
## 262 24 2 10 68 10  
## 263 24 2 12 70 12  
## 264 24 2 14 71 14  
## 265 24 2 16 72 16  
## 266 24 2 18 72 18  
## 267 24 2 20 76 last1  
## 268 24 2 21 74 last2  
## 269 25 2 0 40 birth  
## 270 25 2 2 49 2  
## 271 25 2 4 62 4  
## 272 25 2 6 78 6  
## 273 25 2 8 102 8  
## 274 25 2 10 124 10  
## 275 25 2 12 146 12  
## 276 25 2 14 164 14  
## 277 25 2 16 197 16  
## 278 25 2 18 231 18  
## 279 25 2 20 259 last1  
## 280 25 2 21 265 last2  
## 281 26 2 0 42 birth  
## 282 26 2 2 48 2  
## 283 26 2 4 57 4  
## 284 26 2 6 74 6  
## 285 26 2 8 93 8  
## 286 26 2 10 114 10  
## 287 26 2 12 136 12  
## 288 26 2 14 147 14  
## 289 26 2 16 169 16  
## 290 26 2 18 205 18  
## 291 26 2 20 236 last1  
## 292 26 2 21 251 last2  
## 293 27 2 0 39 birth  
## 294 27 2 2 46 2  
## 295 27 2 4 58 4  
## 296 27 2 6 73 6  
## 297 27 2 8 87 8  
## 298 27 2 10 100 10  
## 299 27 2 12 115 12  
## 300 27 2 14 123 14  
## 301 27 2 16 144 16  
## 302 27 2 18 163 18  
## 303 27 2 20 185 last1  
## 304 27 2 21 192 last2  
## 305 28 2 0 39 birth  
## 306 28 2 2 46 2  
## 307 28 2 4 58 4  
## 308 28 2 6 73 6  
## 309 28 2 8 92 8  
## 310 28 2 10 114 10  
## 311 28 2 12 145 12  
## 312 28 2 14 156 14  
## 313 28 2 16 184 16  
## 314 28 2 18 207 18  
## 315 28 2 20 212 last1  
## 316 28 2 21 233 last2  
## 317 29 2 0 39 birth  
## 318 29 2 2 48 2  
## 319 29 2 4 59 4  
## 320 29 2 6 74 6  
## 321 29 2 8 87 8  
## 322 29 2 10 106 10  
## 323 29 2 12 134 12  
## 324 29 2 14 150 14  
## 325 29 2 16 187 16  
## 326 29 2 18 230 18  
## 327 29 2 20 279 last1  
## 328 29 2 21 309 last2  
## 329 30 2 0 42 birth  
## 330 30 2 2 48 2  
## 331 30 2 4 59 4  
## 332 30 2 6 72 6  
## 333 30 2 8 85 8  
## 334 30 2 10 98 10  
## 335 30 2 12 115 12  
## 336 30 2 14 122 14  
## 337 30 2 16 143 16  
## 338 30 2 18 151 18  
## 339 30 2 20 157 last1  
## 340 30 2 21 150 last2  
## 341 31 3 0 42 birth  
## 342 31 3 2 53 2  
## 343 31 3 4 62 4  
## 344 31 3 6 73 6  
## 345 31 3 8 85 8  
## 346 31 3 10 102 10  
## 347 31 3 12 123 12  
## 348 31 3 14 138 14  
## 349 31 3 16 170 16  
## 350 31 3 18 204 18  
## 351 31 3 20 235 last1  
## 352 31 3 21 256 last2  
## 353 32 3 0 41 birth  
## 354 32 3 2 49 2  
## 355 32 3 4 65 4  
## 356 32 3 6 82 6  
## 357 32 3 8 107 8  
## 358 32 3 10 129 10  
## 359 32 3 12 159 12  
## 360 32 3 14 179 14  
## 361 32 3 16 221 16  
## 362 32 3 18 263 18  
## 363 32 3 20 291 last1  
## 364 32 3 21 305 last2  
## 365 33 3 0 39 birth  
## 366 33 3 2 50 2  
## 367 33 3 4 63 4  
## 368 33 3 6 77 6  
## 369 33 3 8 96 8  
## 370 33 3 10 111 10  
## 371 33 3 12 137 12  
## 372 33 3 14 144 14  
## 373 33 3 16 151 16  
## 374 33 3 18 146 18  
## 375 33 3 20 156 last1  
## 376 33 3 21 147 last2  
## 377 34 3 0 41 birth  
## 378 34 3 2 49 2  
## 379 34 3 4 63 4  
## 380 34 3 6 85 6  
## 381 34 3 8 107 8  
## 382 34 3 10 134 10  
## 383 34 3 12 164 12  
## 384 34 3 14 186 14  
## 385 34 3 16 235 16  
## 386 34 3 18 294 18  
## 387 34 3 20 327 last1  
## 388 34 3 21 341 last2  
## 389 35 3 0 41 birth  
## 390 35 3 2 53 2  
## 391 35 3 4 64 4  
## 392 35 3 6 87 6  
## 393 35 3 8 123 8  
## 394 35 3 10 158 10  
## 395 35 3 12 201 12  
## 396 35 3 14 238 14  
## 397 35 3 16 287 16  
## 398 35 3 18 332 18  
## 399 35 3 20 361 last1  
## 400 35 3 21 373 last2  
## 401 36 3 0 39 birth  
## 402 36 3 2 48 2  
## 403 36 3 4 61 4  
## 404 36 3 6 76 6  
## 405 36 3 8 98 8  
## 406 36 3 10 116 10  
## 407 36 3 12 145 12  
## 408 36 3 14 166 14  
## 409 36 3 16 198 16  
## 410 36 3 18 227 18  
## 411 36 3 20 225 last1  
## 412 36 3 21 220 last2  
## 413 37 3 0 41 birth  
## 414 37 3 2 48 2  
## 415 37 3 4 56 4  
## 416 37 3 6 68 6  
## 417 37 3 8 80 8  
## 418 37 3 10 83 10  
## 419 37 3 12 103 12  
## 420 37 3 14 112 14  
## 421 37 3 16 135 16  
## 422 37 3 18 157 18  
## 423 37 3 20 169 last1  
## 424 37 3 21 178 last2  
## 425 38 3 0 41 birth  
## 426 38 3 2 49 2  
## 427 38 3 4 61 4  
## 428 38 3 6 74 6  
## 429 38 3 8 98 8  
## 430 38 3 10 109 10  
## 431 38 3 12 128 12  
## 432 38 3 14 154 14  
## 433 38 3 16 192 16  
## 434 38 3 18 232 18  
## 435 38 3 20 280 last1  
## 436 38 3 21 290 last2  
## 437 39 3 0 42 birth  
## 438 39 3 2 50 2  
## 439 39 3 4 61 4  
## 440 39 3 6 78 6  
## 441 39 3 8 89 8  
## 442 39 3 10 109 10  
## 443 39 3 12 130 12  
## 444 39 3 14 146 14  
## 445 39 3 16 170 16  
## 446 39 3 18 214 18  
## 447 39 3 20 250 last1  
## 448 39 3 21 272 last2  
## 449 40 3 0 41 birth  
## 450 40 3 2 55 2  
## 451 40 3 4 66 4  
## 452 40 3 6 79 6  
## 453 40 3 8 101 8  
## 454 40 3 10 120 10  
## 455 40 3 12 154 12  
## 456 40 3 14 182 14  
## 457 40 3 16 215 16  
## 458 40 3 18 262 18  
## 459 40 3 20 295 last1  
## 460 40 3 21 321 last2  
## 461 41 4 0 42 birth  
## 462 41 4 2 51 2  
## 463 41 4 4 66 4  
## 464 41 4 6 85 6  
## 465 41 4 8 103 8  
## 466 41 4 10 124 10  
## 467 41 4 12 155 12  
## 468 41 4 14 153 14  
## 469 41 4 16 175 16  
## 470 41 4 18 184 18  
## 471 41 4 20 199 last1  
## 472 41 4 21 204 last2  
## 473 42 4 0 42 birth  
## 474 42 4 2 49 2  
## 475 42 4 4 63 4  
## 476 42 4 6 84 6  
## 477 42 4 8 103 8  
## 478 42 4 10 126 10  
## 479 42 4 12 160 12  
## 480 42 4 14 174 14  
## 481 42 4 16 204 16  
## 482 42 4 18 234 18  
## 483 42 4 20 269 last1  
## 484 42 4 21 281 last2  
## 485 43 4 0 42 birth  
## 486 43 4 2 55 2  
## 487 43 4 4 69 4  
## 488 43 4 6 96 6  
## 489 43 4 8 131 8  
## 490 43 4 10 157 10  
## 491 43 4 12 184 12  
## 492 43 4 14 188 14  
## 493 43 4 16 197 16  
## 494 43 4 18 198 18  
## 495 43 4 20 199 last1  
## 496 43 4 21 200 last2  
## 497 44 4 0 42 birth  
## 498 44 4 2 51 2  
## 499 44 4 4 65 4  
## 500 44 4 6 86 6  
## 501 44 4 8 103 8  
## 502 44 4 10 118 10  
## 503 44 4 12 127 12  
## 504 44 4 14 138 14  
## 505 44 4 16 145 16  
## 506 44 4 18 146 18  
## 507 45 4 0 41 birth  
## 508 45 4 2 50 2  
## 509 45 4 4 61 4  
## 510 45 4 6 78 6  
## 511 45 4 8 98 8  
## 512 45 4 10 117 10  
## 513 45 4 12 135 12  
## 514 45 4 14 141 14  
## 515 45 4 16 147 16  
## 516 45 4 18 174 18  
## 517 45 4 20 197 last1  
## 518 45 4 21 196 last2  
## 519 46 4 0 40 birth  
## 520 46 4 2 52 2  
## 521 46 4 4 62 4  
## 522 46 4 6 82 6  
## 523 46 4 8 101 8  
## 524 46 4 10 120 10  
## 525 46 4 12 144 12  
## 526 46 4 14 156 14  
## 527 46 4 16 173 16  
## 528 46 4 18 210 18  
## 529 46 4 20 231 last1  
## 530 46 4 21 238 last2  
## 531 47 4 0 41 birth  
## 532 47 4 2 53 2  
## 533 47 4 4 66 4  
## 534 47 4 6 79 6  
## 535 47 4 8 100 8  
## 536 47 4 10 123 10  
## 537 47 4 12 148 12  
## 538 47 4 14 157 14  
## 539 47 4 16 168 16  
## 540 47 4 18 185 18  
## 541 47 4 20 210 last1  
## 542 47 4 21 205 last2  
## 543 48 4 0 39 birth  
## 544 48 4 2 50 2  
## 545 48 4 4 62 4  
## 546 48 4 6 80 6  
## 547 48 4 8 104 8  
## 548 48 4 10 125 10  
## 549 48 4 12 154 12  
## 550 48 4 14 170 14  
## 551 48 4 16 222 16  
## 552 48 4 18 261 18  
## 553 48 4 20 303 last1  
## 554 48 4 21 322 last2  
## 555 49 4 0 40 birth  
## 556 49 4 2 53 2  
## 557 49 4 4 64 4  
## 558 49 4 6 85 6  
## 559 49 4 8 108 8  
## 560 49 4 10 128 10  
## 561 49 4 12 152 12  
## 562 49 4 14 166 14  
## 563 49 4 16 184 16  
## 564 49 4 18 203 18  
## 565 49 4 20 233 last1  
## 566 49 4 21 237 last2  
## 567 50 4 0 41 birth  
## 568 50 4 2 54 2  
## 569 50 4 4 67 4  
## 570 50 4 6 84 6  
## 571 50 4 8 105 8  
## 572 50 4 10 122 10  
## 573 50 4 12 155 12  
## 574 50 4 14 175 14  
## 575 50 4 16 205 16  
## 576 50 4 18 234 18  
## 577 50 4 20 264 last1  
## 578 50 4 21 264 last2

chick\_clean %>%  
 filter(age\_days == 20) %>%  
 mutate(weight\_lbs = weight\_gm/453.6) %>%  
 group\_by(diet) %>%  
 summarise(N = n(),   
 mean\_wgt\_gm = mean(weight\_gm),  
 mean\_wgt\_lbs = mean(weight\_lbs))

## # A tibble: 4 × 4  
## diet N mean\_wgt\_gm mean\_wgt\_lbs  
## <fct> <int> <dbl> <dbl>  
## 1 1 17 170. 0.376  
## 2 2 10 206. 0.453  
## 3 3 10 259. 0.571  
## 4 4 9 234. 0.516

# ICMA Sept 23 2024

Five main types of functions in tidyr: - Pivoting: Converts between long and wide forms - Splitting and combining: take character columns and split/combine them - Missing values: For working with NAs (missing values) - Rectangling: Turns deeply nested lists (as from JSON) into tidy tibbles - (Un)Nesting: Converts between grouped data and a form where each group becomes a single row containing a nested data frame

head(billboard)

## # A tibble: 6 × 79  
## artist track date.entered wk1 wk2 wk3 wk4 wk5 wk6 wk7 wk8  
## <chr> <chr> <date> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 2 Pac Baby… 2000-02-26 87 82 72 77 87 94 99 NA  
## 2 2Ge+her The … 2000-09-02 91 87 92 NA NA NA NA NA  
## 3 3 Doors Do… Kryp… 2000-04-08 81 70 68 67 66 57 54 53  
## 4 3 Doors Do… Loser 2000-10-21 76 76 72 69 67 65 55 59  
## 5 504 Boyz Wobb… 2000-04-15 57 34 25 17 17 31 36 49  
## 6 98^0 Give… 2000-08-19 51 39 34 26 26 19 2 2  
## # ℹ 68 more variables: wk9 <dbl>, wk10 <dbl>, wk11 <dbl>, wk12 <dbl>,  
## # wk13 <dbl>, wk14 <dbl>, wk15 <dbl>, wk16 <dbl>, wk17 <dbl>, wk18 <dbl>,  
## # wk19 <dbl>, wk20 <dbl>, wk21 <dbl>, wk22 <dbl>, wk23 <dbl>, wk24 <dbl>,  
## # wk25 <dbl>, wk26 <dbl>, wk27 <dbl>, wk28 <dbl>, wk29 <dbl>, wk30 <dbl>,  
## # wk31 <dbl>, wk32 <dbl>, wk33 <dbl>, wk34 <dbl>, wk35 <dbl>, wk36 <dbl>,  
## # wk37 <dbl>, wk38 <dbl>, wk39 <dbl>, wk40 <dbl>, wk41 <dbl>, wk42 <dbl>,  
## # wk43 <dbl>, wk44 <dbl>, wk45 <dbl>, wk46 <dbl>, wk47 <dbl>, wk48 <dbl>, …

billboard\_long <- billboard %>%  
 pivot\_longer(wk1:wk76,  
 names\_to = "week",  
 values\_to = "position",  
 names\_prefix = "wk",  
 values\_drop\_na = TRUE)

# reversing it back to wide format   
billboard\_long %>%  
 pivot\_wider(names\_from = "week",  
 values\_from = "position")

## # A tibble: 317 × 68  
## artist track date.entered `1` `2` `3` `4` `5` `6` `7` `8`  
## <chr> <chr> <date> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 2 Pac Baby… 2000-02-26 87 82 72 77 87 94 99 NA  
## 2 2Ge+her The … 2000-09-02 91 87 92 NA NA NA NA NA  
## 3 3 Doors D… Kryp… 2000-04-08 81 70 68 67 66 57 54 53  
## 4 3 Doors D… Loser 2000-10-21 76 76 72 69 67 65 55 59  
## 5 504 Boyz Wobb… 2000-04-15 57 34 25 17 17 31 36 49  
## 6 98^0 Give… 2000-08-19 51 39 34 26 26 19 2 2  
## 7 A\*Teens Danc… 2000-07-08 97 97 96 95 100 NA NA NA  
## 8 Aaliyah I Do… 2000-01-29 84 62 51 41 38 35 35 38  
## 9 Aaliyah Try … 2000-03-18 59 53 38 28 21 18 16 14  
## 10 Adams, Yo… Open… 2000-08-26 76 76 74 69 68 67 61 58  
## # ℹ 307 more rows  
## # ℹ 57 more variables: `9` <dbl>, `10` <dbl>, `11` <dbl>, `12` <dbl>,  
## # `13` <dbl>, `14` <dbl>, `15` <dbl>, `16` <dbl>, `17` <dbl>, `18` <dbl>,  
## # `19` <dbl>, `20` <dbl>, `21` <dbl>, `22` <dbl>, `23` <dbl>, `24` <dbl>,  
## # `25` <dbl>, `26` <dbl>, `27` <dbl>, `28` <dbl>, `29` <dbl>, `30` <dbl>,  
## # `31` <dbl>, `32` <dbl>, `33` <dbl>, `34` <dbl>, `35` <dbl>, `36` <dbl>,  
## # `37` <dbl>, `38` <dbl>, `39` <dbl>, `40` <dbl>, `41` <dbl>, `42` <dbl>, …

?pivot\_longer

billboard %>%  
 separate(col = date.entered,  
 into = c("year", "month", "day"))

## # A tibble: 317 × 81  
## artist track year month day wk1 wk2 wk3 wk4 wk5 wk6 wk7  
## <chr> <chr> <chr> <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 2 Pac Baby… 2000 02 26 87 82 72 77 87 94 99  
## 2 2Ge+her The … 2000 09 02 91 87 92 NA NA NA NA  
## 3 3 Doors Do… Kryp… 2000 04 08 81 70 68 67 66 57 54  
## 4 3 Doors Do… Loser 2000 10 21 76 76 72 69 67 65 55  
## 5 504 Boyz Wobb… 2000 04 15 57 34 25 17 17 31 36  
## 6 98^0 Give… 2000 08 19 51 39 34 26 26 19 2  
## 7 A\*Teens Danc… 2000 07 08 97 97 96 95 100 NA NA  
## 8 Aaliyah I Do… 2000 01 29 84 62 51 41 38 35 35  
## 9 Aaliyah Try … 2000 03 18 59 53 38 28 21 18 16  
## 10 Adams, Yol… Open… 2000 08 26 76 76 74 69 68 67 61  
## # ℹ 307 more rows  
## # ℹ 69 more variables: wk8 <dbl>, wk9 <dbl>, wk10 <dbl>, wk11 <dbl>,  
## # wk12 <dbl>, wk13 <dbl>, wk14 <dbl>, wk15 <dbl>, wk16 <dbl>, wk17 <dbl>,  
## # wk18 <dbl>, wk19 <dbl>, wk20 <dbl>, wk21 <dbl>, wk22 <dbl>, wk23 <dbl>,  
## # wk24 <dbl>, wk25 <dbl>, wk26 <dbl>, wk27 <dbl>, wk28 <dbl>, wk29 <dbl>,  
## # wk30 <dbl>, wk31 <dbl>, wk32 <dbl>, wk33 <dbl>, wk34 <dbl>, wk35 <dbl>,  
## # wk36 <dbl>, wk37 <dbl>, wk38 <dbl>, wk39 <dbl>, wk40 <dbl>, wk41 <dbl>, …

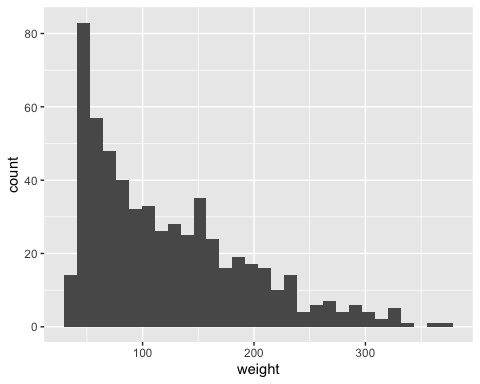
# ICMA Sept 25 2024

head(ChickWeight)

## weight Time Chick Diet  
## 1 42 0 1 1  
## 2 51 2 1 1  
## 3 59 4 1 1  
## 4 64 6 1 1  
## 5 76 8 1 1  
## 6 93 10 1 1

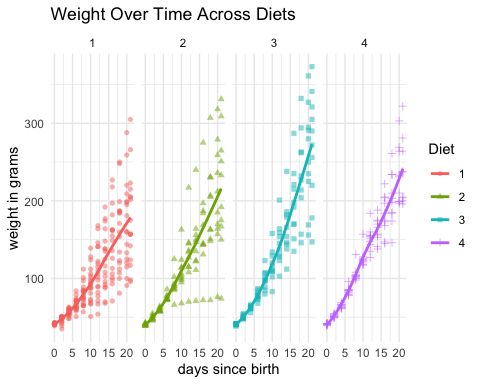
ChickWeight %>%  
 ggplot(aes(x = weight)) + #provide the aesthetic mapping   
 geom\_histogram() #tell ggplot which geometry to use, in this case a histogram

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



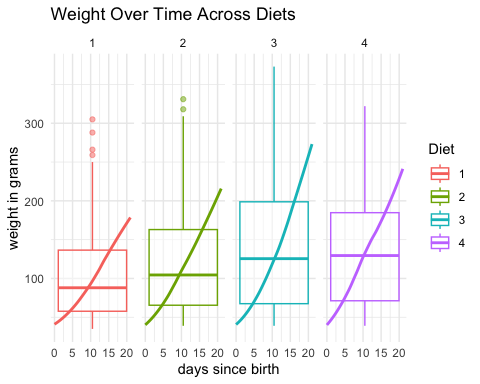
ChickWeight %>%  
 ggplot(aes(y = weight,  
 x = Time,  
 color = Diet,  
 shape = Diet)) +  
 geom\_point(alpha = 0.5) +  
 geom\_smooth(se = FALSE) +  
 theme\_minimal() +  
 facet\_grid(~Diet) + #faceting by diet groups  
 labs(y = "weight in grams",  
 x = "days since birth",  
 title = "Weight Over Time Across Diets")

## `geom\_smooth()` using method = 'loess' and formula = 'y ~ x'



ChickWeight %>%  
 ggplot(aes(y = weight,  
 x = Time,  
 color = Diet,  
 shape = Diet)) +  
 geom\_boxplot(alpha = 0.5) +  
 geom\_smooth(se = FALSE) +  
 theme\_minimal() +  
 facet\_grid(~Diet) + #faceting by diet groups  
 labs(y = "weight in grams",  
 x = "days since birth",  
 title = "Weight Over Time Across Diets")

## `geom\_smooth()` using method = 'loess' and formula = 'y ~ x'



# ICMA Oct 7 2024

as.numeric(12)

## [1] 12

is.numeric(mean(c(12, 31, 15.57, 4)))

## [1] TRUE

# prediction = TRUE

((1 + 2) < -1000) | (TRUE & NA)

## [1] NA

(2 + 3 + 9 - 4 == 2 + 6 + as.integer("2.9")) & TRUE | NA

## [1] TRUE

# ICMA Oct 9 2024

my\_number <- 12

age\_data <- tribble(  
 ~ id, ~ age,  
 1, 8,  
 2, 10,  
 3, 8,  
 5, 9  
)  
  
gender\_data <- tribble(  
 ~ id, ~ gender,  
 1, "f",  
 2, "m",  
 3, "nb",  
 4, "m",  
 6, "f"  
)

age\_data

## # A tibble: 4 × 2  
## id age  
## <dbl> <dbl>  
## 1 1 8  
## 2 2 10  
## 3 3 8  
## 4 5 9

gender\_data

## # A tibble: 5 × 2  
## id gender  
## <dbl> <chr>   
## 1 1 f   
## 2 2 m   
## 3 3 nb   
## 4 4 m   
## 5 6 f

full\_join(age\_data, gender\_data)

## Joining with `by = join\_by(id)`

## # A tibble: 6 × 3  
## id age gender  
## <dbl> <dbl> <chr>   
## 1 1 8 f   
## 2 2 10 m   
## 3 3 8 nb   
## 4 5 9 <NA>   
## 5 4 NA m   
## 6 6 NA f

right\_join(age\_data, gender\_data)

## Joining with `by = join\_by(id)`

## # A tibble: 5 × 3  
## id age gender  
## <dbl> <dbl> <chr>   
## 1 1 8 f   
## 2 2 10 m   
## 3 3 8 nb   
## 4 4 NA m   
## 5 6 NA f

left\_join(age\_data, gender\_data)

## Joining with `by = join\_by(id)`

## # A tibble: 4 × 3  
## id age gender  
## <dbl> <dbl> <chr>   
## 1 1 8 f   
## 2 2 10 m   
## 3 3 8 nb   
## 4 5 9 <NA>

inner\_join(age\_data, gender\_data)

## Joining with `by = join\_by(id)`

## # A tibble: 3 × 3  
## id age gender  
## <dbl> <dbl> <chr>   
## 1 1 8 f   
## 2 2 10 m   
## 3 3 8 nb

# ICMA Oct 16 2024

mean(c(1, 2, 3))

## [1] 2

(1 + 2 + 3)/3

## [1] 2

vec <- c(1, 2, 3)

(vec[1] + vec[2] + vec[3])/length(vec)

## [1] 2

fave\_number <- tribble(~name, ~number,  
 "MC", 8,  
 "AP", 9,  
 "NG", 12,  
 "VE", 6,  
 "AL", 17,  
 "AR", 3,  
 "MC", 7,  
 "RK", 8,  
 "AC", 4,  
 "SC", 21,  
 "KK", 16,  
 "RK", 28,  
 "VK", 17,  
 "MH", 12  
 )

for (i in fave\_number) {  
 print(i)  
}

## [1] "MC" "AP" "NG" "VE" "AL" "AR" "MC" "RK" "AC" "SC" "KK" "RK" "VK" "MH"  
## [1] 8 9 12 6 17 3 7 8 4 21 16 28 17 12

for (i in fave\_number$number) {  
 print(i)  
}

## [1] 8  
## [1] 9  
## [1] 12  
## [1] 6  
## [1] 17  
## [1] 3  
## [1] 7  
## [1] 8  
## [1] 4  
## [1] 21  
## [1] 16  
## [1] 28  
## [1] 17  
## [1] 12

How are fave\_number[“number”] and fave\_number$number different?

typeof(fave\_number["number"])

## [1] "list"

typeof(fave\_number$number)

## [1] "double"

for (i in fave\_number[["number"]]) {  
 print(i)  
}

## [1] 8  
## [1] 9  
## [1] 12  
## [1] 6  
## [1] 17  
## [1] 3  
## [1] 7  
## [1] 8  
## [1] 4  
## [1] 21  
## [1] 16  
## [1] 28  
## [1] 17  
## [1] 12

for (i in seq\_along(fave\_number$name)) {  
 print(str\_c(fave\_number$name[i], "'s favorite number is ", fave\_number$number[i]))  
}

## [1] "MC's favorite number is 8"  
## [1] "AP's favorite number is 9"  
## [1] "NG's favorite number is 12"  
## [1] "VE's favorite number is 6"  
## [1] "AL's favorite number is 17"  
## [1] "AR's favorite number is 3"  
## [1] "MC's favorite number is 7"  
## [1] "RK's favorite number is 8"  
## [1] "AC's favorite number is 4"  
## [1] "SC's favorite number is 21"  
## [1] "KK's favorite number is 16"  
## [1] "RK's favorite number is 28"  
## [1] "VK's favorite number is 17"  
## [1] "MH's favorite number is 12"

# ICMA Oct 30 2024

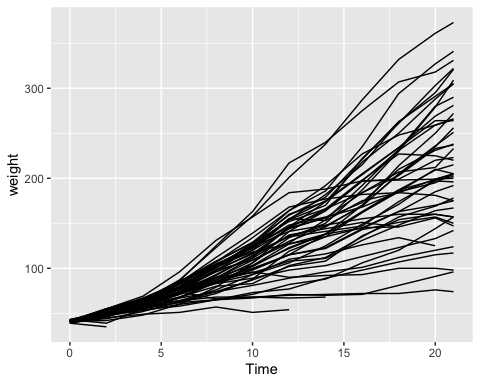
Let’s write a sentence that has **bold** and *italics*!

Part 2: create a variable, then write a sentence that uses that variable, bold, and italics

name <- "Vidushi"

My name is Vidushi and I really hate it when people use both **bold** and *italics* in the same sentence.

ChickWeight %>%  
 ggplot(aes(y = weight, x = Time, group = Chick)) +  
 geom\_line()



Chick weight across time