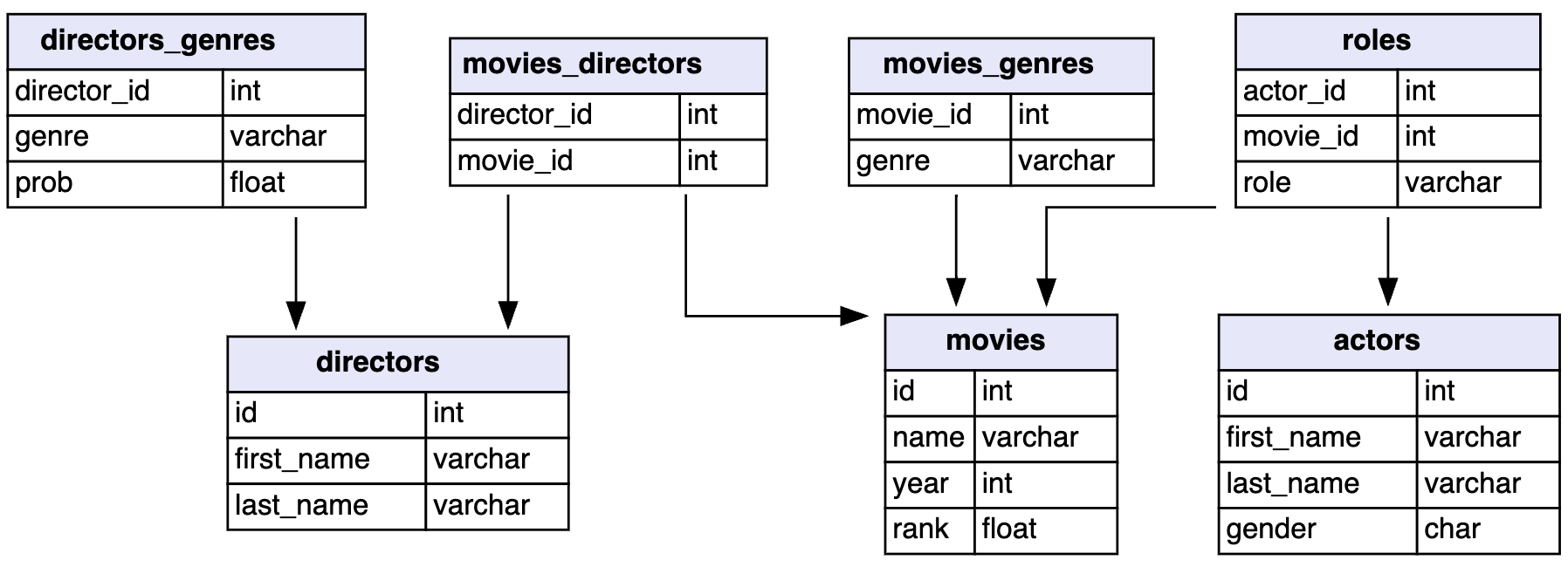
Week 4 Day 2: Lab Warm-up

|  |
| --- |
| Warning |
| For this lab, you need to download data from the Week 4 Notes page on the course website and add them to a data/ directory. |

# Data

We will use 7 datasets that describe movies from [IMDb](https://www.imdb.com/).



Relationship between data sets in IMDb movie data.

load(file = "data/imdb\_data.Rdata")

On Thursday will also look at joining datasets created from the Lab 2 Rodent data. **Note that you will need to change the file path to be appropriate for your directory strucure!**

rodent <- read\_csv("../../labs/lab2/surveys.csv")  
  
species <- rodent |>   
 select(genus:taxa, species\_id) |>   
 distinct()  
  
measurements <- rodent |>   
 select(genus, species, sex:weight) |>   
 rename(genus\_name = genus)

… and daily rainfall observed in SLO in January 2023. [Data source](cesanluisobispo.ucanr.edu)

slo\_rainfall <- read\_excel("data/2023-rainfall-slo.xlsx")  
  
slo\_rainfall <- slo\_rainfall |>   
 mutate(across(Sunday:Saturday, as.numeric))

… and to practice working with factor variables, we will use data on songs that Taylor Swift included in the Era’s Tour.

full\_eras <- read\_excel("data/TS\_data.xlsx", sheet = "full")  
  
# randomly sample 25 songs  
set.seed(2)  
eras\_data <- full\_eras |>   
 slice\_sample(n = 25) |>   
 select(Song, Album)

# Tuesday

## Data Joins

inner\_join(directors\_genres,   
 movies\_directors,   
 by = "director\_id")

inner\_join(directors\_genres,   
 directors,   
 by = join\_by(director\_id == id))

directors\_genres |>   
 inner\_join(movies\_directors,   
 by = "director\_id")

## Which Join?

How many movies are there in the data for each director (by name), including if any directors don’t have any movies in the data?

directors |>   
 ??\_join(movies\_directors,   
 by = join\_by(id == director\_id))

What is the complete set of movies and actors included in the data?

roles |>   
 ??\_join(actors,   
 by = join\_by(actor\_id == id))

## Filtering Joins

directors\_genres |>   
 semi\_join(movies\_directors)

directors\_genres |>  
 filter(director\_id %in% movies\_directors$director\_id)

directors\_genres |>   
 anti\_join(movies\_directors)

directors\_genres |>  
 filter(!director\_id %in% movies\_directors$director\_id)

## Data Pivots

Example data includes the cereal dataset from the liver package (which we saw last week)…

data(cereal)

How would we plot the mean cereal nutrients by shelf (as shown below) with the wide data?

my\_colors <- c("calories\_col" = "steelblue", "sugars\_col" = "orange3")  
  
cereal |>   
 group\_by(shelf) |>   
 summarise(across(calories:vitamins, mean)) |>   
 ggplot() +  
 geom\_point(aes(x = shelf, y = calories, color = "calories\_col")) +  
 geom\_line(aes(x = shelf, y = calories, color = "calories\_col")) +   
 geom\_point(aes(x = shelf, y = sugars, color = "sugars\_col")) +  
 geom\_line(aes(x = shelf, y = sugars, color = "sugars\_col")) +  
 scale\_color\_manual(values = my\_colors, labels = names(my\_colors)) +  
 labs(x = "Shelf", y = "", subtitle = "Mean Amount", color = "Nutrient")

cereal\_long <- cereal |>   
 pivot\_longer(cols = calories:vitamins,  
 names\_to = "Nutrient",  
 values\_to = "Amount") |>   
 group\_by(shelf, Nutrient) |>   
 summarise(mean\_amount = mean(Amount))

cereal\_long |>   
 ggplot(aes(x = shelf,   
 y = mean\_amount,   
 color = Nutrient)) +  
 geom\_point() +  
 geom\_line() +  
 labs(x = "Shelf", y = "", subtitle = "Mean Amount")

slo\_rainfall |>   
 pivot\_longer(cols = Sunday:Saturday,  
 names\_to = "Day\_of\_Week",  
 values\_to = "Daily\_Rainfall")

mean\_protein <- cereal |>   
 group\_by(manuf, shelf) |>   
 summarize(mean\_protein = mean(protein))

mean\_protein |>   
 pivot\_wider(id\_cols = manuf,  
 names\_from = shelf,  
 values\_from = mean\_protein)

# Thursday

## Extending Joins

movies\_directors |>   
 inner\_join(directors,   
 join\_by(director\_id == id)) |>   
 inner\_join(movies,  
 join\_by(movie\_id == id)) |>   
 group\_by(first\_name, last\_name) |>  
 summarize(start\_year = min(year),  
 end\_year = max(year)) |>   
 mutate(n\_years\_active = end\_year - start\_year) |>   
 arrange(desc(n\_years\_active))

directors |>   
 inner\_join(movies\_directors,   
 join\_by(id == director\_id))

head(species)  
head(measurements)

measurements |>   
 inner\_join(species,   
 by = join\_by(genus\_name == genus))

species |>   
 full\_join(measurements,  
 join\_by(species == species,   
 genus == genus\_name))

## Factors with forcats

eras\_data |>   
 mutate(Album = fct(Album)) |>   
 pull(Album)

eras\_data |>   
 mutate(Album = fct(Album,  
 levels = c("Fearless","Speak Now","Red",  
 "1989", "Reputation","Lover",  
 "Folklore", "Evermore","Midnights"))) |>   
 pull(Album)

eras\_data |>   
 mutate(Album = fct(Album,   
 levels = c("Taylor Swift",  
 "Fearless","Speak Now","Red",  
 "1989", "Reputation","Lover",  
 "Folklore", "Evermore","Midnights",  
 "The Tortured Poets Department"))) |>   
 pull(Album)

eras\_data |>  
 mutate(Album = fct\_recode(.f = Album,  
 "folklore" = "Folklore",  
 "evermore" = "Evermore",  
 "reputation" = "Reputation")) |>  
 pull(Album)

eras\_data |>   
 mutate(Genre = fct\_collapse(.f= Album,  
 "country pop" = c("Taylor Swift", "Fearless"),  
 "pop rock" = c("Speak Now","Red"),  
 "electropop" = c("1989","reputation","Lover"),  
 "folk pop" = c("folklore","evermore"),  
 "alt-pop" = "Midnights")) |>   
 slice\_sample(n = 6)

eras\_data |>   
 mutate(Album = fct\_relevel(.f = Album,   
 c("Fearless","1989","Taylor Swift",  
 "Speak Now","Red","Midnights","reputation",  
 "folklore","Lover","evermore"))) |>  
 pull(Album) |>  
 levels()

## Re-order Factors for Plots

full\_eras |>   
 mutate(Album = fct(Album,  
 levels = c("Fearless","Speak Now","Red",  
 "1989","Reputation","Lover",  
 "Folklore","Evermore",  
 "Midnights"))) |>   
 ggplot() +  
 geom\_bar(aes(y = Album), fill = "#A5C9A5") +  
 theme\_minimal() +  
 labs(x = "Number of Songs",  
 y = "",  
 subtitle = "Album",  
 title = "Songs Played on the Eras Tour")

full\_eras |>   
 ggplot() +  
 geom\_bar(aes(y = fct\_infreq(Album)),   
 fill = "#A5C9A5") +  
 theme\_minimal() +  
 labs(x = "Number of Songs",  
 y = "",  
 subtitle = "Album",  
 title = "Songs Played on the Eras Tour")

full\_eras |>   
 ggplot(aes(x = Length,   
 y = fct\_reorder(.f = Album,  
 .x = Length,  
 .fun = mean),   
 fill = Album)) +  
 geom\_density\_ridges() +  
 theme\_minimal() +  
 theme(legend.position = "none")+  
 labs(x = "Song Length (mins)",  
 y = "",  
 subtitle = "Album",  
 title = "Songs Played on the Eras Tour")

cereal\_long |>   
 ggplot(aes(x = shelf,   
 y = mean\_amount,   
 color = fct\_reorder2(.x = shelf,  
 .y = mean\_amount,  
 .f = Nutrient))) +  
 geom\_point() +  
 geom\_line() +  
 labs(x = "Shelf", y = "",   
 subtitle = "Mean Amount",  
 color = "Nutrient")

## Code Formatting

Don’t forget, writing “tidy” and “well documented” code are two of the learning targets for this course. As such, I would strongly encourage you to use every opportunity to practice these skills.

As you are writing code for this assignment, make sure your code follows the [tidyverse style guide for dplyr code](https://style.tidyverse.org/pipes.html). Specifically, your code should:

* use whitespace liberally
  + before & after every = sign
  + after every ,
  + before every |>
* use new lines liberally
  + after every |>
  + after , when needed (if code is more than 80 characters in length)