Using either the same dataset(s) you used in the previous weeks’ exercise or a brand-new dataset of your choosing, perform the following transformations (Remember, anything you learn about the Housing dataset in these two weeks can be used for a later exercise!)

library(readxl)  
setwd("C:/Users/brean/OneDrive/Desktop/NucampFolder/projects/dsc520-1")  
housing\_data = read\_excel("data/week-7-housing.xlsx")  
print(housing\_data)

## # A tibble: 12,865 × 24  
## `Sale Date` Sale Pric…¹ sale\_…² sale\_…³ sale\_…⁴ sitet…⁵ addr\_…⁶ zip5  
## <dttm> <dbl> <dbl> <dbl> <chr> <chr> <chr> <dbl>  
## 1 2006-01-03 00:00:00 698000 1 3 <NA> R1 17021 … 98052  
## 2 2006-01-03 00:00:00 649990 1 3 <NA> R1 11927 … 98052  
## 3 2006-01-03 00:00:00 572500 1 3 <NA> R1 13315 … 98052  
## 4 2006-01-03 00:00:00 420000 1 3 <NA> R1 3303 1… 98052  
## 5 2006-01-03 00:00:00 369900 1 3 15 R1 16126 … 98052  
## 6 2006-01-03 00:00:00 184667 1 15 18 51 R1 8101 2… 98053  
## 7 2006-01-04 00:00:00 1050000 1 3 <NA> R1 21634 … 98053  
## 8 2006-01-04 00:00:00 875000 1 3 <NA> R1 21404 … 98053  
## 9 2006-01-04 00:00:00 660000 1 3 <NA> R1 7525 2… 98053  
## 10 2006-01-04 00:00:00 650000 1 3 <NA> R1 17703 … 98052  
## # … with 12,855 more rows, 16 more variables: ctyname <chr>, postalctyn <chr>,  
## # lon <dbl>, lat <dbl>, building\_grade <dbl>, square\_feet\_total\_living <dbl>,  
## # bedrooms <dbl>, bath\_full\_count <dbl>, bath\_half\_count <dbl>,  
## # bath\_3qtr\_count <dbl>, year\_built <dbl>, year\_renovated <dbl>,  
## # current\_zoning <chr>, sq\_ft\_lot <dbl>, prop\_type <chr>, present\_use <dbl>,  
## # and abbreviated variable names ¹​`Sale Price`, ²​sale\_reason,  
## # ³​sale\_instrument, ⁴​sale\_warning, ⁵​sitetype, ⁶​addr\_full

Using the dplyr package, use the 6 different operations to analyze/transform the data - GroupBy, Summarize, Mutate, Filter, Select, and Arrange – Remember this isn’t just modifying data, you are learning about your data also – so play around and start to understand your dataset in more detail

library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

colnames(housing\_data)[2] <- "sale\_price"  
colnames(housing\_data)[14] <- "square\_feet"  
colnames(housing\_data)[1] <- "sale\_date"  
  
bedrooms\_group <- housing\_data %>% group\_by(bedrooms)  
print(bedrooms\_group)

## # A tibble: 12,865 × 24  
## # Groups: bedrooms [12]  
## sale\_date sale\_price sale\_r…¹ sale\_…² sale\_…³ sitet…⁴ addr\_…⁵ zip5  
## <dttm> <dbl> <dbl> <dbl> <chr> <chr> <chr> <dbl>  
## 1 2006-01-03 00:00:00 698000 1 3 <NA> R1 17021 … 98052  
## 2 2006-01-03 00:00:00 649990 1 3 <NA> R1 11927 … 98052  
## 3 2006-01-03 00:00:00 572500 1 3 <NA> R1 13315 … 98052  
## 4 2006-01-03 00:00:00 420000 1 3 <NA> R1 3303 1… 98052  
## 5 2006-01-03 00:00:00 369900 1 3 15 R1 16126 … 98052  
## 6 2006-01-03 00:00:00 184667 1 15 18 51 R1 8101 2… 98053  
## 7 2006-01-04 00:00:00 1050000 1 3 <NA> R1 21634 … 98053  
## 8 2006-01-04 00:00:00 875000 1 3 <NA> R1 21404 … 98053  
## 9 2006-01-04 00:00:00 660000 1 3 <NA> R1 7525 2… 98053  
## 10 2006-01-04 00:00:00 650000 1 3 <NA> R1 17703 … 98052  
## # … with 12,855 more rows, 16 more variables: ctyname <chr>, postalctyn <chr>,  
## # lon <dbl>, lat <dbl>, building\_grade <dbl>, square\_feet <dbl>,  
## # bedrooms <dbl>, bath\_full\_count <dbl>, bath\_half\_count <dbl>,  
## # bath\_3qtr\_count <dbl>, year\_built <dbl>, year\_renovated <dbl>,  
## # current\_zoning <chr>, sq\_ft\_lot <dbl>, prop\_type <chr>, present\_use <dbl>,  
## # and abbreviated variable names ¹​sale\_reason, ²​sale\_instrument,  
## # ³​sale\_warning, ⁴​sitetype, ⁵​addr\_full

year <- filter(housing\_data, year\_built == 1933)  
print(year)

## # A tibble: 6 × 24  
## sale\_date sale\_price sale\_re…¹ sale\_…² sale\_…³ sitet…⁴ addr\_…⁵ zip5  
## <dttm> <dbl> <dbl> <dbl> <chr> <chr> <chr> <dbl>  
## 1 2006-07-13 00:00:00 715000 1 3 <NA> R1 11815 … 98053  
## 2 2007-05-16 00:00:00 286000 8 3 12 R1 12841 … 98052  
## 3 2012-05-11 00:00:00 162000 1 3 60 R1 15504 … 98052  
## 4 2013-10-22 00:00:00 525000 1 3 15 46 R1 11815 … 98053  
## 5 2014-09-29 00:00:00 550000 1 3 <NA> R1 14314 … 98052  
## 6 2016-09-16 00:00:00 405000 1 3 <NA> R1 12607 … 98052  
## # … with 16 more variables: ctyname <chr>, postalctyn <chr>, lon <dbl>,  
## # lat <dbl>, building\_grade <dbl>, square\_feet <dbl>, bedrooms <dbl>,  
## # bath\_full\_count <dbl>, bath\_half\_count <dbl>, bath\_3qtr\_count <dbl>,  
## # year\_built <dbl>, year\_renovated <dbl>, current\_zoning <chr>,  
## # sq\_ft\_lot <dbl>, prop\_type <chr>, present\_use <dbl>, and abbreviated  
## # variable names ¹​sale\_reason, ²​sale\_instrument, ³​sale\_warning, ⁴​sitetype,  
## # ⁵​addr\_full

a <- housing\_data %>% summarize(mean(bedrooms))  
print(a)

## # A tibble: 1 × 1  
## `mean(bedrooms)`  
## <dbl>  
## 1 3.48

c <- select(housing\_data, square\_feet, bedrooms, year\_built, sale\_price)  
g <- select(housing\_data, sale\_date, ctyname, bath\_full\_count, sq\_ft\_lot)  
print(c)

## # A tibble: 12,865 × 4  
## square\_feet bedrooms year\_built sale\_price  
## <dbl> <dbl> <dbl> <dbl>  
## 1 2810 4 2003 698000  
## 2 2880 4 2006 649990  
## 3 2770 4 1987 572500  
## 4 1620 3 1968 420000  
## 5 1440 3 1980 369900  
## 6 4160 4 2005 184667  
## 7 3960 5 1993 1050000  
## 8 3720 4 1988 875000  
## 9 4160 4 1978 660000  
## 10 2760 4 1976 650000  
## # … with 12,855 more rows

h <- c %>% mutate(price\_per\_sq = (sale\_price / square\_feet))  
print(h)

## # A tibble: 12,865 × 5  
## square\_feet bedrooms year\_built sale\_price price\_per\_sq  
## <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 2810 4 2003 698000 248.   
## 2 2880 4 2006 649990 226.   
## 3 2770 4 1987 572500 207.   
## 4 1620 3 1968 420000 259.   
## 5 1440 3 1980 369900 257.   
## 6 4160 4 2005 184667 44.4  
## 7 3960 5 1993 1050000 265.   
## 8 3720 4 1988 875000 235.   
## 9 4160 4 1978 660000 159.   
## 10 2760 4 1976 650000 236.   
## # … with 12,855 more rows

d <- arrange(c, sale\_price, year\_built)  
print(d)

## # A tibble: 12,865 × 4  
## square\_feet bedrooms year\_built sale\_price  
## <dbl> <dbl> <dbl> <dbl>  
## 1 1040 3 1900 698  
## 2 5830 4 1969 698  
## 3 3430 3 1955 873  
## 4 2160 2 1968 873  
## 5 1850 3 1960 998  
## 6 4610 4 2015 1000  
## 7 1610 3 1909 1070  
## 8 2610 3 1988 1500  
## 9 4610 4 2015 1500  
## 10 2000 3 1987 2031  
## # … with 12,855 more rows

Using the purrr package – perform 2 functions on your dataset. You could use zip\_n, keep, discard, compact, etc.

library(purrr)  
#f <- keep(housing\_data, starts\_with("s"))  
#discard()  
  
#housing\_data %>% map(function(x) sum(x, na.rm=TRUE))  
  
#f <- housing\_data %>% keep(~ .$bath\_full\_count >= 2)  
  
#f <- housing\_data %>% keep(~ get\_element(housing\_data, "bath\_full\_count") >= 2)  
  
#could not get keep to work for some reason??? if you know what I'm doing wrong, please let me know!   
  
f <- housing\_data %>% mutate(sales = compact(sale\_price))  
  
print(f)

## # A tibble: 12,865 × 25  
## sale\_date sale\_price sale\_r…¹ sale\_…² sale\_…³ sitet…⁴ addr\_…⁵ zip5  
## <dttm> <dbl> <dbl> <dbl> <chr> <chr> <chr> <dbl>  
## 1 2006-01-03 00:00:00 698000 1 3 <NA> R1 17021 … 98052  
## 2 2006-01-03 00:00:00 649990 1 3 <NA> R1 11927 … 98052  
## 3 2006-01-03 00:00:00 572500 1 3 <NA> R1 13315 … 98052  
## 4 2006-01-03 00:00:00 420000 1 3 <NA> R1 3303 1… 98052  
## 5 2006-01-03 00:00:00 369900 1 3 15 R1 16126 … 98052  
## 6 2006-01-03 00:00:00 184667 1 15 18 51 R1 8101 2… 98053  
## 7 2006-01-04 00:00:00 1050000 1 3 <NA> R1 21634 … 98053  
## 8 2006-01-04 00:00:00 875000 1 3 <NA> R1 21404 … 98053  
## 9 2006-01-04 00:00:00 660000 1 3 <NA> R1 7525 2… 98053  
## 10 2006-01-04 00:00:00 650000 1 3 <NA> R1 17703 … 98052  
## # … with 12,855 more rows, 17 more variables: ctyname <chr>, postalctyn <chr>,  
## # lon <dbl>, lat <dbl>, building\_grade <dbl>, square\_feet <dbl>,  
## # bedrooms <dbl>, bath\_full\_count <dbl>, bath\_half\_count <dbl>,  
## # bath\_3qtr\_count <dbl>, year\_built <dbl>, year\_renovated <dbl>,  
## # current\_zoning <chr>, sq\_ft\_lot <dbl>, prop\_type <chr>, present\_use <dbl>,  
## # sales <dbl>, and abbreviated variable names ¹​sale\_reason, ²​sale\_instrument,  
## # ³​sale\_warning, ⁴​sitetype, ⁵​addr\_full

#j <- c %>% discard(sitetype)  
#could also not figure out discard  
  
#print(j)

Use the cbind and rbind function on your dataset

c\_bind <- cbind(c,g)  
#print(c\_bind)  
#removed these two print functions because it made my homework 500+ pages but they worked!   
  
new\_row <- c("2170", "4", "1970", "730000")  
  
r\_bind <- rbind(c, new\_row)  
#print(r\_bind)

Split a string, then concatenate the results back together

library(stringr)  
  
split\_string <- str\_split(string = year$ctyname, "&")  
split\_string2 <- str\_split(string = year$sale\_date, '&')  
print(split\_string)

## [[1]]  
## [1] NA  
##   
## [[2]]  
## [1] NA  
##   
## [[3]]  
## [1] "REDMOND"  
##   
## [[4]]  
## [1] NA  
##   
## [[5]]  
## [1] NA  
##   
## [[6]]  
## [1] NA

print(split\_string2)

## [[1]]  
## [1] "2006-07-13"  
##   
## [[2]]  
## [1] "2007-05-16"  
##   
## [[3]]  
## [1] "2012-05-11"  
##   
## [[4]]  
## [1] "2013-10-22"  
##   
## [[5]]  
## [1] "2014-09-29"  
##   
## [[6]]  
## [1] "2016-09-16"

paste\_string <- paste(split\_string, split\_string2)  
print(paste\_string)

## [1] "NA 2006-07-13" "NA 2007-05-16" "REDMOND 2012-05-11"  
## [4] "NA 2013-10-22" "NA 2014-09-29" "NA 2016-09-16"