assignment_05_PothineniKalyan

PothineniKalyan

2023-04-15

R Markdown

```
# Assignment: ASSIGNMENT 5
# Name: Pothineni, Kalyan
# Date: 2023-04-15

## Load the `readxl` and 'dplyr library
library(readxl)
```

```
## Warning: package 'readxl' was built under R version 4.2.3
```

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.2.3
```

```
##
## 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
```

```
library(purrr)
```

```
## Warning: package 'purrr' was built under R version 4.2.3
```

```
setwd("C:/Users/kpothine/OneDrive - Waste Management/Documents/NDO_GIT/dsc520")

## Using the excel_sheets() function from the `readxl` package,
## list the worksheets from the file `data/G04ResultsDetail2004-11-02.xls`
housing_df <- read_excel('data/week-6-housing.xlsx', sheet = 'Sheet2')
str(housing_df)
```

```
## tibble [12,865 x 24] (S3: tbl_df/tbl/data.frame)
##  $ Sale Date               : POSIXct[1:12865], format: "2006-01-03" "2006-01-03" ...
##  $ Sale Price              : num [1:12865] 698000 649990 572500 420000 369900 ...
##  $ sale_reason             : num [1:12865] 1 1 1 1 1 1 1 1 1 1 ...
##  $ sale_instrument         : num [1:12865] 3 3 3 3 3 15 3 3 3 3 ...
##  $ sale_warning            : chr [1:12865] NA NA NA NA ...
##  $ sitetype                : chr [1:12865] "R1" "R1" "R1" "R1" ...
##  $ addr_full               : chr [1:12865] "17021 NE 113TH CT" "11927 178TH PL NE" "13315 174TH AVE N
##  $ zip5                    : num [1:12865] 98052 98052 98052 98052 98052 ...
##  $ ctyname                 : chr [1:12865] "REDMOND" "REDMOND" NA "REDMOND" ...
##  $ postalctyn              : chr [1:12865] "REDMOND" "REDMOND" "REDMOND" "REDMOND" ...
##  $ lon                     : num [1:12865] -122 -122 -122 -122 -122 ...
##  $ lat                     : num [1:12865] 47.7 47.7 47.7 47.6 47.7 ...
##  $ building_grade          : num [1:12865] 9 9 8 8 7 7 10 10 9 8 ...
##  $ square_feet_total_living: num [1:12865] 2810 2880 2770 1620 1440 4160 3960 3720 4160 2760 ...
##  $ bedrooms                : num [1:12865] 4 4 4 3 3 4 5 4 4 4 ...
##  $ bath_full_count         : num [1:12865] 2 2 1 1 1 2 3 2 2 1 ...
##  $ bath_half_count         : num [1:12865] 1 0 1 0 0 1 0 1 1 0 ...
##  $ bath_3qtr_count         : num [1:12865] 0 1 1 1 1 1 1 0 1 1 ...
##  $ year_built              : num [1:12865] 2003 2006 1987 1968 1980 ...
##  $ year_renovated          : num [1:12865] 0 0 0 0 0 0 0 0 0 0 ...
##  $ current_zoning          : chr [1:12865] "R4" "R4" "R6" "R4" ...
##  $ sq_ft_lot               : num [1:12865] 6635 5570 8444 9600 7526 ...
##  $ prop_type               : chr [1:12865] "R" "R" "R" "R" ...
##  $ present_use             : num [1:12865] 2 2 2 2 2 2 2 2 2 2 ...
```

```
##head(housing_df)

## Use colnames() to change the column name
colnames(housing_df)[colnames(housing_df)=="Sale Price"] <- "Sale_Price"

colnames(housing_df)[colnames(housing_df)=="Sale Date"] <- "Sale_Date"
##-------------------------------------------------------------------------------
## Exercise 5.2 (a)
##-------------------------------------------------------------------------------
## Group by operation
grouped_zon_df <- housing_df %>%
  group_by(current_zoning) %>%
  summarise(Total_value = sum(Sale_Price))

grouped_zon_df
```

```
## # A tibble: 24 x 2
##    current_zoning Total_value
##    <chr>                <dbl>
##  1 A10                8966322
##  2 A10SO              4400000
##  3 BC                  650000
##  4 EH                24302146
##  5 GC                  150000
##  6 R1                124218128
##  7 R12               139728961
##  8 R18                 6755950
##  9 R1P                 5129000
```

```
## 10 R3               211230843
## # ... with 14 more rows
```

```
## Group by operation
grouped_sale_reason_df <- housing_df %>% group_by(sale_reason) %>%
  summarise(mean_price = mean(Sale_Price), mean_size = mean(sq_ft_lot))

grouped_sale_reason_df
```

```
## # A tibble: 17 x 3
##    sale_reason mean_price mean_size
##          <dbl>      <dbl>     <dbl>
##  1           0    407645     15684.
##  2           1    672185.    21611.
##  3           2    203904     51328
##  4           3   1220217     46609
##  5           4    492739.    48760.
##  6           6    428900      7770
##  7           7    850779    123854.
##  8           8    465581.    28345.
##  9          10    539660.     9160
## 10          11    360000      9775
## 11          12    568962.    19044.
## 12          13    414200      6374
## 13          14    223170.    28572.
## 14          16    572597.    53274.
## 15          17    530000     11792
## 16          18    421678.    33097.
## 17          19    175161.    75438.
```

```
## Group by operation - cityname
grouped_ctyname_df <- housing_df %>%
  group_by(ctyname) %>%
  summarise(mean_price = mean(Sale_Price), mean_size = mean(sq_ft_lot))
grouped_ctyname_df
```

```
## # A tibble: 3 x 3
##   ctyname   mean_price mean_size
##   <chr>          <dbl>     <dbl>
## 1 REDMOND      644803.     9001.
## 2 SAMMAMISH    972480.    25475.
## 3 <NA>         674973.    36821.
```

```
## Summarize operation: calculate the average sales price and size of all houses
summary_df <- housing_df %>%
  summarise(avg_price = mean(Sale_Price), avg_size = mean(sq_ft_lot))
summary_df
```

```
## # A tibble: 1 x 2
##   avg_price avg_size
##       <dbl>    <dbl>
## 1   660738.   22229.
```

```
## Mutate operation: add a new column for the price per square foot
mutated_df <- housing_df %>%
  mutate(price_per_sqft = Sale_Price / sq_ft_lot)
str(mutated_df)
```

```
## tibble [12,865 x 25] (S3: tbl_df/tbl/data.frame)
##  $ Sale_Date               : POSIXct[1:12865], format: "2006-01-03" "2006-01-03" ...
##  $ Sale_Price              : num [1:12865] 698000 649990 572500 420000 369900 ...
##  $ sale_reason             : num [1:12865] 1 1 1 1 1 1 1 1 1 1 ...
##  $ sale_instrument         : num [1:12865] 3 3 3 3 3 15 3 3 3 3 ...
##  $ sale_warning            : chr [1:12865] NA NA NA NA ...
##  $ sitetype                : chr [1:12865] "R1" "R1" "R1" "R1" ...
##  $ addr_full               : chr [1:12865] "17021 NE 113TH CT" "11927 178TH PL NE" "13315 174TH AVE N
##  $ zip5                    : num [1:12865] 98052 98052 98052 98052 98052 ...
##  $ ctyname                 : chr [1:12865] "REDMOND" "REDMOND" NA "REDMOND" ...
##  $ postalctyn              : chr [1:12865] "REDMOND" "REDMOND" "REDMOND" "REDMOND" ...
##  $ lon                     : num [1:12865] -122 -122 -122 -122 -122 ...
##  $ lat                     : num [1:12865] 47.7 47.7 47.7 47.6 47.7 ...
##  $ building_grade          : num [1:12865] 9 9 8 8 7 7 10 10 9 8 ...
##  $ square_feet_total_living: num [1:12865] 2810 2880 2770 1620 1440 4160 3960 3720 4160 2760 ...
##  $ bedrooms                : num [1:12865] 4 4 4 3 3 4 5 4 4 4 ...
##  $ bath_full_count         : num [1:12865] 2 2 1 1 1 2 3 2 2 1 ...
##  $ bath_half_count         : num [1:12865] 1 0 1 0 0 1 0 1 1 0 ...
##  $ bath_3qtr_count         : num [1:12865] 0 1 1 1 1 1 1 0 1 1 ...
##  $ year_built              : num [1:12865] 2003 2006 1987 1968 1980 ...
##  $ year_renovated          : num [1:12865] 0 0 0 0 0 0 0 0 0 0 ...
##  $ current_zoning          : chr [1:12865] "R4" "R4" "R6" "R4" ...
##  $ sq_ft_lot               : num [1:12865] 6635 5570 8444 9600 7526 ...
##  $ prop_type               : chr [1:12865] "R" "R" "R" "R" ...
##  $ present_use             : num [1:12865] 2 2 2 2 2 2 2 2 2 2 ...
##  $ price_per_sqft          : num [1:12865] 105.2 116.7 67.8 43.8 49.1 ...
```

```
## Filter operation: filter the data set to only include houses built after 1990
filtered_df <- housing_df %>%
  filter(year_built > 2000)
##filtered_df
nrow(filtered_df)
```

```
## [1] 6226
```

```
## Select operation: select only the city and price columns
selected_df <- housing_df %>%
  select(ctyname, Sale_Price)
str(selected_df)
```

```
## tibble [12,865 x 2] (S3: tbl_df/tbl/data.frame)
##  $ ctyname   : chr [1:12865] "REDMOND" "REDMOND" NA "REDMOND" ...
##  $ Sale_Price: num [1:12865] 698000 649990 572500 420000 369900 ...
```

```
## Arrange operation: sort the data set by year built in ascending order
arranged_df <- housing_df %>%
  arrange(year_built)
arranged_df
```

```
## # A tibble: 12,865 x 24
##    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 2006-03-13 00:00:00     455000        1       3 <NA>    R1      16810 ~ 98052
##  2 2006-10-04 00:00:00     675000        1       3 <NA>    R1      19841 ~ 98053
##  3 2007-02-16 00:00:00     550000        8       3 12      R1      8008 1~ 98052
##  4 2009-12-04 00:00:00     400000       18       4 <NA>    R1      19841 ~ 98053
##  5 2010-07-06 00:00:00        698        1      26 24      R1      19805 ~ 98053
##  6 2013-05-23 00:00:00     286300        4      18 15 31   R1      13840 ~ 98052
##  7 2007-03-16 00:00:00     430000        1       3 <NA>    R1      26030 ~ 98053
##  8 2006-10-18 00:00:00     620000        1       3 <NA>    R1      8005 1~ 98052
##  9 2012-02-28 00:00:00     550000        1       3 16 45   R1      18631 ~ 98052
## 10 2010-05-14 00:00:00       1070        1      26 24      R1      20828 ~ 98053
## # ... 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_reason, 2: sale_instrument, 3: sale_warning, 4: sitetype, ...
```

```
## Arrange operation: sort the data set by year built in decending order
arranged_desc_df <- housing_df %>%
  arrange(desc(Sale_Price))
arranged_desc_df
```

```
## # A tibble: 12,865 x 24
##    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 2010-03-02 00:00:00    4400000        1       3 35 45   R1      12025 ~ 98052
##  2 2010-03-02 00:00:00    4400000        1       3 35 45   R1      12053 ~ 98052
##  3 2011-11-17 00:00:00    4380542        1      22 11 45   R1      17137 ~ 98052
##  4 2011-11-17 00:00:00    4380542        1      22 11 45   R1      11818 ~ 98052
##  5 2011-11-17 00:00:00    4380542        1      22 11 45   R1      17011 ~ 98052
##  6 2011-11-17 00:00:00    4380542        1      22 11 45   R1      16943 ~ 98052
##  7 2011-11-17 00:00:00    4380542        1      22 11 45   R1      16944 ~ 98052
##  8 2011-11-17 00:00:00    4380542        1      22 11 45   R1      16909 ~ 98052
##  9 2011-11-17 00:00:00    4380542        1      22 11 45   R1      17128 ~ 98052
## 10 2011-11-17 00:00:00    4380542        1      22 11 45   R1      17136 ~ 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_reason, 2: sale_instrument, 3: sale_warning, 4: sitetype, ...
```

```
##-------------------------------------------------------------------------------
## Exercise 5.2 (b)
##-------------------------------------------------------------------------------

str(housing_df)
```

```
## tibble [12,865 x 24] (S3: tbl_df/tbl/data.frame)
```

```
##  $ Sale_Date               : POSIXct[1:12865], format: "2006-01-03" "2006-01-03" ...
##  $ Sale_Price              : num [1:12865] 698000 649990 572500 420000 369900 ...
##  $ sale_reason             : num [1:12865] 1 1 1 1 1 1 1 1 1 1 ...
##  $ sale_instrument         : num [1:12865] 3 3 3 3 3 15 3 3 3 3 ...
##  $ sale_warning            : chr [1:12865] NA NA NA NA ...
##  $ sitetype                : chr [1:12865] "R1" "R1" "R1" "R1" ...
##  $ addr_full               : chr [1:12865] "17021 NE 113TH CT" "11927 178TH PL NE" "13315 174TH AVE N
##  $ zip5                    : num [1:12865] 98052 98052 98052 98052 98052 ...
##  $ ctyname                 : chr [1:12865] "REDMOND" "REDMOND" NA "REDMOND" ...
##  $ postalctyn              : chr [1:12865] "REDMOND" "REDMOND" "REDMOND" "REDMOND" ...
##  $ lon                     : num [1:12865] -122 -122 -122 -122 -122 ...
##  $ lat                     : num [1:12865] 47.7 47.7 47.7 47.6 47.7 ...
##  $ building_grade          : num [1:12865] 9 9 8 8 7 7 10 10 9 8 ...
##  $ square_feet_total_living: num [1:12865] 2810 2880 2770 1620 1440 4160 3960 3720 4160 2760 ...
##  $ bedrooms                : num [1:12865] 4 4 4 3 3 4 5 4 4 4 ...
##  $ bath_full_count         : num [1:12865] 2 2 1 1 1 2 3 2 2 1 ...
##  $ bath_half_count         : num [1:12865] 1 0 1 0 0 1 0 1 1 0 ...
##  $ bath_3qtr_count         : num [1:12865] 0 1 1 1 1 1 1 0 1 1 ...
##  $ year_built              : num [1:12865] 2003 2006 1987 1968 1980 ...
##  $ year_renovated          : num [1:12865] 0 0 0 0 0 0 0 0 0 0 ...
##  $ current_zoning          : chr [1:12865] "R4" "R4" "R6" "R4" ...
##  $ sq_ft_lot               : num [1:12865] 6635 5570 8444 9600 7526 ...
##  $ prop_type               : chr [1:12865] "R" "R" "R" "R" ...
##  $ present_use             : num [1:12865] 2 2 2 2 2 2 2 2 2 2 ...
```

```r
# use the zip_n function to combine the columns into a list
#zipped_list <- housing_df %>%
#  select(building_grade, Sale_Price) %>%
#  zip_n()


zipped_list <- housing_df %>%
  select(building_grade, Sale_Price) %>%
  pmap(list) %>%
  unname()

zipped_list_ctyname <- housing_df %>%
  select(ctyname, Sale_Price) %>%
  pmap(list) %>%
  unname()

head(zipped_list, n=5)
```

```
## [[1]]
## [[1]]$building_grade
## [1] 9
##
## [[1]]$Sale_Price
## [1] 698000
##
##
## [[2]]
## [[2]]$building_grade
## [1] 9
```

```
##
## [[2]]$Sale_Price
## [1] 649990
##
##
## [[3]]
## [[3]]$building_grade
## [1] 8
##
## [[3]]$Sale_Price
## [1] 572500
##
##
## [[4]]
## [[4]]$building_grade
## [1] 8
##
## [[4]]$Sale_Price
## [1] 420000
##
##
## [[5]]
## [[5]]$building_grade
## [1] 7
##
## [[5]]$Sale_Price
## [1] 369900
```

```
class(zipped_list)
```

```
## [1] "list"
```

```
zipped_list[[2]]
```

```
## $building_grade
## [1] 9
##
## $Sale_Price
## [1] 649990
```

```
## Keep function to allow Sale_Price greater than 500000
keep_list <- keep(zipped_list, ~ .x[[2]] > 500000)
head(keep_list, n=5)
```

```
## [[1]]
## [[1]]$building_grade
## [1] 9
##
## [[1]]$Sale_Price
## [1] 698000
##
##
```

```
## [[2]]
## [[2]]$building_grade
## [1] 9
##
## [[2]]$Sale_Price
## [1] 649990
##
##
## [[3]]
## [[3]]$building_grade
## [1] 8
##
## [[3]]$Sale_Price
## [1] 572500
##
##
## [[4]]
## [[4]]$building_grade
## [1] 10
##
## [[4]]$Sale_Price
## [1] 1050000
##
##
## [[5]]
## [[5]]$building_grade
## [1] 10
##
## [[5]]$Sale_Price
## [1] 875000
```

```
keep_list_02 <- keep(zipped_list, ~ .x$Sale_Price > 1000000)
head(keep_list_02, n=5)
```

```
## [[1]]
## [[1]]$building_grade
## [1] 10
##
## [[1]]$Sale_Price
## [1] 1050000
##
##
## [[2]]
## [[2]]$building_grade
## [1] 9
##
## [[2]]$Sale_Price
## [1] 1392000
##
##
## [[3]]
## [[3]]$building_grade
## [1] 11
##
```

```
## [[3]]$Sale_Price
## [1] 1445000
##
##
## [[4]]
## [[4]]$building_grade
## [1] 9
##
## [[4]]$Sale_Price
## [1] 1053649
##
##
## [[5]]
## [[5]]$building_grade
## [1] 11
##
## [[5]]$Sale_Price
## [1] 1900000
```

```
## discard function where building_grade is 7
discard_list <- discard(zipped_list, ~ .x[[1]] == 7)
head(discard_list, n=5)
```

```
## [[1]]
## [[1]]$building_grade
## [1] 9
##
## [[1]]$Sale_Price
## [1] 698000
##
##
## [[2]]
## [[2]]$building_grade
## [1] 9
##
## [[2]]$Sale_Price
## [1] 649990
##
##
## [[3]]
## [[3]]$building_grade
## [1] 8
##
## [[3]]$Sale_Price
## [1] 572500
##
##
## [[4]]
## [[4]]$building_grade
## [1] 8
##
## [[4]]$Sale_Price
## [1] 420000
##
```

```
##
## [[5]]
## [[5]]$building_grade
## [1] 10
##
## [[5]]$Sale_Price
## [1] 1050000
```

```
## discard function where building_grade is less than 9
discard_list_02 <- discard(zipped_list, ~ .x$building_grade < 9)
head(discard_list_02, n=5)
```

```
## [[1]]
## [[1]]$building_grade
## [1] 9
##
## [[1]]$Sale_Price
## [1] 698000
##
##
## [[2]]
## [[2]]$building_grade
## [1] 9
##
## [[2]]$Sale_Price
## [1] 649990
##
##
## [[3]]
## [[3]]$building_grade
## [1] 10
##
## [[3]]$Sale_Price
## [1] 1050000
##
##
## [[4]]
## [[4]]$building_grade
## [1] 10
##
## [[4]]$Sale_Price
## [1] 875000
##
##
## [[5]]
## [[5]]$building_grade
## [1] 9
##
## [[5]]$Sale_Price
## [1] 660000
```

```
## Clean the list with NA or blanks
cleaned_list <- discard(zipped_list_ctyname, ~ is.na(.x$ctyname))
head(cleaned_list, n=10)
```

```
## [[1]]
## [[1]]$ctyname
## [1] "REDMOND"
##
## [[1]]$Sale_Price
## [1] 698000
##
##
## [[2]]
## [[2]]$ctyname
## [1] "REDMOND"
##
## [[2]]$Sale_Price
## [1] 649990
##
##
## [[3]]
## [[3]]$ctyname
## [1] "REDMOND"
##
## [[3]]$Sale_Price
## [1] 420000
##
##
## [[4]]
## [[4]]$ctyname
## [1] "REDMOND"
##
## [[4]]$Sale_Price
## [1] 369900
##
##
## [[5]]
## [[5]]$ctyname
## [1] "REDMOND"
##
## [[5]]$Sale_Price
## [1] 650000
##
##
## [[6]]
## [[6]]$ctyname
## [1] "REDMOND"
##
## [[6]]$Sale_Price
## [1] 599950
##
##
## [[7]]
## [[7]]$ctyname
## [1] "REDMOND"
##
## [[7]]$Sale_Price
## [1] 526787
```

```
##
##
## [[8]]
## [[8]]$ctyname
## [1] "REDMOND"
##
## [[8]]$Sale_Price
## [1] 470000
##
##
## [[9]]
## [[9]]$ctyname
## [1] "REDMOND"
##
## [[9]]$Sale_Price
## [1] 507950
##
##
## [[10]]
## [[10]]$ctyname
## [1] "REDMOND"
##
## [[10]]$Sale_Price
## [1] 589950
```

```
## compact function by removing all rows with NA values
compact_list <- compact(keep_list_02)
head(compact_list, n=10)
```

```
## [[1]]
## [[1]]$building_grade
## [1] 10
##
## [[1]]$Sale_Price
## [1] 1050000
##
##
## [[2]]
## [[2]]$building_grade
## [1] 9
##
## [[2]]$Sale_Price
## [1] 1392000
##
##
## [[3]]
## [[3]]$building_grade
## [1] 11
##
## [[3]]$Sale_Price
## [1] 1445000
##
##
## [[4]]
```

```
## [[4]]$building_grade
## [1] 9
##
## [[4]]$Sale_Price
## [1] 1053649
##
##
## [[5]]
## [[5]]$building_grade
## [1] 11
##
## [[5]]$Sale_Price
## [1] 1900000
##
##
## [[6]]
## [[6]]$building_grade
## [1] 9
##
## [[6]]$Sale_Price
## [1] 1080135
##
##
## [[7]]
## [[7]]$building_grade
## [1] 11
##
## [[7]]$Sale_Price
## [1] 1075000
##
##
## [[8]]
## [[8]]$building_grade
## [1] 9
##
## [[8]]$Sale_Price
## [1] 1520000
##
##
## [[9]]
## [[9]]$building_grade
## [1] 6
##
## [[9]]$Sale_Price
## [1] 1390000
##
##
## [[10]]
## [[10]]$building_grade
## [1] 10
##
## [[10]]$Sale_Price
## [1] 1390000
```

```r
##-------------------------------------------------------------------------------
## Exercise 5.2 (c) Use the cbind and rbind function on your dataset
##-------------------------------------------------------------------------------

## Creating two new data frames to bind with housing_df
new_df1 <- grouped_sale_reason_df

new_df2 <- data.frame(sale_reason = c(30,40,50,60,70,80,90),
                      mean_price = c(120,130,140,150,160,170,280),
                      mean_size = c(1,2,3,4,5,6,7))

## Using rbind() to combine the housing_df with new_df1 and new_df2, basically a row binding
combined_df <- rbind(new_df1, new_df2)
print(combined_df)
```

```
## # A tibble: 24 x 3
##    sale_reason mean_price mean_size
##          <dbl>      <dbl>     <dbl>
##  1           0    407645     15684.
##  2           1    672185.    21611.
##  3           2    203904     51328
##  4           3   1220217     46609
##  5           4    492739.    48760.
##  6           6    428900      7770
##  7           7    850779    123854.
##  8           8    465581.    28345.
##  9          10    539660.     9160
## 10          11    360000      9775
## # ... with 14 more rows
```

```r
## Using cbind() to add a new column to the combined_df, basically a column binding
new_col <- c('A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W
combined_df <- cbind(combined_df, new_col)

print(combined_df)
```

```
##    sale_reason mean_price mean_size new_col
## 1            0   407645.0  15684.50       A
## 2            1   672185.3  21610.55       B
## 3            2   203904.0  51328.00       C
## 4            3  1220217.0  46609.00       D
## 5            4   492739.4  48759.87       E
## 6            6   428900.0   7770.00       F
## 7            7   850779.0 123853.67       G
## 8            8   465580.7  28344.51       H
## 9           10   539659.5   9160.00       I
## 10          11   360000.0   9775.00       J
## 11          12   568962.0  19043.95       K
## 12          13   414200.0   6374.00       L
## 13          14   223169.8  28571.90       M
## 14          16   572597.3  53274.33       N
## 15          17   530000.0  11792.00       O
## 16          18   421678.1  33097.18       P
```

```
## 17          19   175161.3  75438.33       Q
## 18          30      120.0      1.00       R
## 19          40      130.0      2.00       S
## 20          50      140.0      3.00       T
## 21          60      150.0      4.00       U
## 22          70      160.0      5.00       V
## 23          80      170.0      6.00       W
## 24          90      280.0      7.00       X
```

```
##-------------------------------------------------------------------------------
## Exercise 5.2 (d), Split a string, then concatenate the results back togethe
##-------------------------------------------------------------------------------

my_string <- "Splitting a string at a specific character"
## split at space character
my_list <- strsplit(my_string, " ")
print(my_list)
```

```
## [[1]]
## [1] "Splitting" "a"         "string"    "at"        "a"         "specific"
## [7] "character"
```

```
## Reference: https://stackoverflow.com/questions/2247045/chopping-a-string-into-a-vector-of-fixed-widt
## Splitting a string into fixed-width chunks
my_string_02 <- "1234567890"
my_list_02 <- strsplit(my_string_02, "(?<=\\G.{2})", perl = TRUE)
print(my_list_02)
```

```
## [[1]]
## [1] "12" "34" "56" "78" "90"
```

```
## Splitting a string into fixed-width chunks
my_string_03 <- "1234567890"
sst <- strsplit(my_string_03, "")[[1]]
paste0(sst[c(TRUE, FALSE)], sst[c(FALSE, TRUE)])
```

```
## [1] "12" "34" "56" "78" "90"
```

```
## Concatenate chunks back together with a delimiter
my_new_string <- paste(my_string_03, collapse = "-")
print(my_new_string)
```

```
## [1] "1234567890"
```

```
##2
# Split the string at the space character
my_string_split <- strsplit(my_string, " ")[[1]]
# Concatenate the split string back together with a hyphen
my_string_concat <- paste(my_string_split, collapse = " ")
# Print the result
print(my_string_concat)
```

```
## [1] "Splitting a string at a specific character"
```

```
##------------------------------------------------------------------------------
```