DSC520 Week5 - 5.2 Exercise

Maxim Bilenkin

2025-01-07

```
knitr::opts_chunk$set(echo = TRUE)

# Setting CRAN mirror.
options(repos = c(CRAN = "https://cran.rstudio.com/"))

# Installing necessary packages if not already installed
necessary_packages <- c("dplyr", "readxl", "purrr")
new_packages <- necessary_packages[!(necessary_packages %in%
                                       installed.packages()[,"Package"])]
if(length(new_packages)) install.packages(new_packages)

# Loading necessary packages
library(dplyr)
library(readxl)
library(purrr)
```

```
# a. 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.

# Loading the Housing dataset file.
housing_data_file <- read_excel("C:/Users/maxim/Downloads/week-6-housing.xlsx")

# Displaying column names to identify the correct ones
print(head(colnames(housing_data_file), 5))
```

```
## [1] "Sale Date"       "Sale Price"      "sale_reason"     "sale_instrument"
## [5] "sale_warning"
```

```
# Temporarily set output width to prevent truncation
options(width = 80)

# Displaying the first 5 rows to understand data structure
print(head(housing_data_file, 5))
```

```
## # A tibble: 5 x 24
##   `Sale Date`         `Sale Price` sale_reason sale_instrument sale_warning
##   <dttm>                     <dbl>       <dbl>           <dbl> <chr>
## 1 2006-01-03 00:00:00       698000           1               3 <NA>
## 2 2006-01-03 00:00:00       649990           1               3 <NA>
## 3 2006-01-03 00:00:00       572500           1               3 <NA>
## 4 2006-01-03 00:00:00       420000           1               3 <NA>
```

```
## 5 2006-01-03 00:00:00       369900           1               3 15
## # i 19 more variables: sitetype <chr>, addr_full <chr>, zip5 <dbl>,
## #   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>
```

```
# Check and print column types with controlled width and list length
print(str(housing_data_file, width = 70, list.len = 5))
```

```
## tibble [12,865 x 24] (S3: tbl_df/tbl/data.frame)
##  $ Sale Date               : POSIXct[1:12865], format: "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 ...
##   [list output truncated]
## NULL
```

```
# Ensuring 'Sale Price' is numeric and handling non-numeric characters
housing_data_file <- housing_data_file %>%
  mutate(`Sale Price` = as.numeric(gsub("[^0-9.]", "",
                                        as.character(`Sale Price`))))

# Display the first 5 rows to confirm changes
print(head(housing_data_file, 5))
```

```
## # A tibble: 5 x 24
##   `Sale Date`         `Sale Price` sale_reason sale_instrument sale_warning
##   <dttm>                     <dbl>       <dbl>           <dbl> <chr>
## 1 2006-01-03 00:00:00       698000           1               3 <NA>
## 2 2006-01-03 00:00:00       649990           1               3 <NA>
## 3 2006-01-03 00:00:00       572500           1               3 <NA>
## 4 2006-01-03 00:00:00       420000           1               3 <NA>
## 5 2006-01-03 00:00:00       369900           1               3 15
## # i 19 more variables: sitetype <chr>, addr_full <chr>, zip5 <dbl>,
## #   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>
```

```
# Reset the output width to default
options(width = 80)

# Grouping data by 'sitetype'.
grouped_by_site_type <- housing_data_file %>% group_by(sitetype)

# Summarizing average housing price by sitetype.
summarized_data <- grouped_by_site_type %>%
  summarize(average_price = mean(`Sale Price`, na.rm = TRUE))

# Calculating and creating a new column for price per square foot.
mutated_data <- housing_data_file %>%
  mutate(price_per_sqft = `Sale Price` / square_feet_total_living)
```

```r
# Filtering houses with more than 2 bedrooms.
houses_with_more_than_2_bedrooms <- housing_data_file %>%
  filter(bedrooms > 2)

# Selecting specific columns.
specific_columns <- housing_data_file %>%
  select(`Sale Price`, square_feet_total_living, bedrooms)

# Sorting the data by Sale Price in descending order.
sorted_data <- housing_data_file %>%
  arrange(desc(`Sale Price`))

# Completed data set after transformation.
completed_data_set <- housing_data_file %>%
  filter(bedrooms > 2) %>%
  mutate(price_per_square_foot = `Sale Price` / square_feet_total_living) %>%
  select(sitetype, `Sale Price`, square_feet_total_living,
         price_per_square_foot, bedrooms) %>%
  arrange(desc(price_per_square_foot))

# Displaying the first 5 rows of the completed data set sample.
print(head(completed_data_set, 5))
```

```
## # A tibble: 5 x 5
##   sitetype `Sale Price` square_feet_total_living price_per_square_foot bedrooms
##   <chr>           <dbl>                   <dbl>                  <dbl>    <dbl>
## 1 R1            4311000                    1670                  2581.        3
## 2 R1            3175000                    1460                  2175.        3
## 3 R1            3175000                    1460                  2175.        3
## 4 R1            3175000                    1460                  2175.        3
## 5 R1            3150000                    1460                  2158.        3
```

```r
# b. Using the purrr package - perform 2 functions on your dataset. You could
#    use zip_n, keep, discard, compact, etc.

# Making data frames to have the same length otherwise system throws an error.
min_length <- min(nrow(mutated_data), nrow(houses_with_more_than_2_bedrooms),
                  nrow(sorted_data))
mutated_data <- mutated_data[1:min_length, ]
houses_with_more_than_2_bedrooms <-
    houses_with_more_than_2_bedrooms[1:min_length, ]
sorted_data <- sorted_data[1:min_length, ]

# Ensuring all data frames have the same columns otherwise system throws error.
common_cols <- Reduce(intersect, list(names(mutated_data),
                                      names(houses_with_more_than_2_bedrooms),
                                      names(sorted_data)))
mutated_data <- mutated_data[, common_cols]
houses_with_more_than_2_bedrooms <-
    houses_with_more_than_2_bedrooms[, common_cols]
sorted_data <- sorted_data[, common_cols]

# Converting data frames to lists of columns.
mutated_list <- as.list(mutated_data)
```

```
houses_2_plus_bedrooms_list <- as.list(houses_with_more_than_2_bedrooms)
sorted_data_list <- as.list(sorted_data)

# Using 'purrr' package and 'pmap' method to combine three lists. The zip_n()
# method not found in purrr package. Thus, using similar pmap() instead.
combined_multiple_columns <- pmap(list(mutated_list,
                                       houses_2_plus_bedrooms_list,
                                       sorted_data_list),
                                  function(x, y, z) list(x, y, z))

# Converting combined multiple columns of lists to a data frame for better readability.
combined_df <- data.frame(
    Sale_Date = sapply(combined_multiple_columns[[1]], function(x) x[[1]]),
    Sale_Price = sapply(combined_multiple_columns[[2]], function(x) x[[1]]),
    Sale_Reason = sapply(combined_multiple_columns[[3]], function(x) x[[1]]),
    Sale_Instrument = sapply(combined_multiple_columns[[4]], function(x) x[[1]]),
    Sale_Warning = sapply(combined_multiple_columns[[5]], function(x) x[[1]]) )

# Using keep() method to filter and keep only numeric columns from combined_df.
combined_numeric_columns <- keep(combined_df, is.numeric)

# Displaying summarized data frame.
print(head(combined_numeric_columns,5))
```

```
##    Sale_Date Sale_Price Sale_Reason Sale_Instrument
## 1 1136246400     698000           1               3
## 2 1136246400     698000           1               3
## 3 1267488000    4400000           1               3
```

```
# c. Use the cbind() and rbind() function on your dataset.

# Creating vector with 'Sale Price'.
sale_price <- housing_data_file$'Sale Price'

# Creating vector with complete address using the following columns.
address <- housing_data_file %>% select(addr_full, zip5, ctyname, postalctyn)

# Using cbin() method combining both vectors.
combined_two_vectors <- cbind(sale_price, address)

# Printing the first 5 rows of the combined two vectors.
print(head(combined_two_vectors, 5))
```

```
##   sale_price          addr_full  zip5 ctyname postalctyn
## 1     698000  17021 NE 113TH CT 98052 REDMOND    REDMOND
## 2     649990  11927 178TH PL NE 98052 REDMOND    REDMOND
## 3     572500 13315 174TH AVE NE 98052    <NA>    REDMOND
## 4     420000  3303 178TH AVE NE 98052 REDMOND    REDMOND
## 5     369900  16126 NE 108TH CT 98052 REDMOND    REDMOND
```

```
# Creating two subsets from dataset file.
subset1 <- housing_data_file[1:5, c("Sale Price", "addr_full", "zip5",
                                    "ctyname", "postalctyn")]
subset2 <- housing_data_file[6:10, c("Sale Price", "addr_full", "zip5",
                                     "ctyname", "postalctyn")]
```

```r
# Combining both subsets with rows using rbind() method.
combined_rows_data <- rbind(subset1, subset2)

# Printing the file with combined two subsets with rows.
print(combined_rows_data)
```

```
## # A tibble: 10 x 5
##    `Sale Price` addr_full           zip5 ctyname postalctyn
##           <dbl> <chr>              <dbl> <chr>   <chr>
##  1       698000 17021 NE 113TH CT  98052 REDMOND REDMOND
##  2       649990 11927 178TH PL NE  98052 REDMOND REDMOND
##  3       572500 13315 174TH AVE NE 98052 <NA>    REDMOND
##  4       420000 3303 178TH AVE NE  98052 REDMOND REDMOND
##  5       369900 16126 NE 108TH CT  98052 REDMOND REDMOND
##  6       184667 8101 229TH DR NE   98053 <NA>    REDMOND
##  7      1050000 21634 NE 87TH PL   98053 <NA>    REDMOND
##  8       875000 21404 NE 67TH ST   98053 <NA>    REDMOND
##  9       660000 7525 238TH AVE NE  98053 <NA>    REDMOND
## 10       650000 17703 NE 26TH ST   98052 REDMOND REDMOND
```

```r
# d. Split a string, then concatenate the results back together.

# Taking 'addr_full' column and making sure it is a character.
housing_data_file <- housing_data_file %>%
    mutate(addr_full = as.character(addr_full))

# Splitting the addr_full column into words.
splitted_addr_full <- strsplit(housing_data_file$addr_full, " ")

# Displaying the first 5 splitted addresses to verify.
print(head(splitted_addr_full, 5))
```

```
## [[1]]
## [1] "17021" "NE"    "113TH" "CT"
##
## [[2]]
## [1] "11927" "178TH" "PL"    "NE"
##
## [[3]]
## [1] "13315" "174TH" "AVE"   "NE"
##
## [[4]]
## [1] "3303"  "178TH" "AVE"   "NE"
##
## [[5]]
## [1] "16126" "NE"    "108TH" "CT"
```

```r
# Concatenating the splitted words back into one sentence using hyphen "-".
concatenated_address <- sapply(splitted_addr_full, function(x)
    paste(x, collapse ="-"))

# Creating new column to add concatenated address line in the data file.
housing_data_file <- housing_data_file %>%
    mutate(concatenated_address = concatenated_address)
```

```r
# Printing first 5 rows of the new crated alternate housing data file.
print(housing_data_file[1:5, c("addr_full", "concatenated_address")])
```

```
## # A tibble: 5 x 2
##   addr_full          concatenated_address
##   <chr>              <chr>
## 1 17021 NE 113TH CT  17021-NE-113TH-CT
## 2 11927 178TH PL NE  11927-178TH-PL-NE
## 3 13315 174TH AVE NE 13315-174TH-AVE-NE
## 4 3303 178TH AVE NE  3303-178TH-AVE-NE
## 5 16126 NE 108TH CT  16126-NE-108TH-CT
```