R - Practice 01 - v1.1

한상곤(sangkon@pusan.ac.kr / sigmadream@gmail.com)

2023.06.13, updated: 2023.11.20

Contents

dplyr & tidyr

```
df <- mpg
str(df)
```

```
## tibble [234 x 11] (S3: tbl_df/tbl/data.frame)
##  $ manufacturer: chr [1:234] "audi" "audi" "audi" "audi" ...
##  $ model       : chr [1:234] "a4" "a4" "a4" "a4" ...
##  $ displ       : num [1:234] 1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...
...
```

```
nrow(df); ncol(df)
```

```
## [1] 234
```

```
## [1] 11
```

Manipulate variables(columns)

select(), rename()

```
df.car.info <- select(df, manufacturer, model, year)
head(df.car.info)
```

```
## # A tibble: 6 x 3
##   manufacturer model  year
##   <chr>        <chr> <int>
## 1 audi         a4     1999
...
```

```
select(df, starts_with(match = "m")) %>% head()
```

```
## # A tibble: 6 x 2
##   manufacturer model
##   <chr>        <chr>
## 1 audi         a4
...
```

```
select(df, contains(match = "r")) %>% head()
```

```
## # A tibble: 6 x 4
##   manufacturer  year trans      drv
##   <chr>        <int> <chr>      <chr>
## 1 audi          1999 auto(l5)   f
...
```

```
select(df, ends_with(match = "y")) %>% head()
```

```
## # A tibble: 6 x 2
##     cty   hwy
##   <int> <int>
## 1    18    29
...
```

```
select(df, 1:3) %>% head()
```

```
## # A tibble: 6 x 3
##   manufacturer model displ
##   <chr>        <chr> <dbl>
```

```
## 1 audi         a4      1.8
...
```

```
select(df, c(2,5,7)) %>% head()
```

```
## # A tibble: 6 x 3
##   model   cyl drv
##   <chr> <int> <chr>
## 1 a4        4 f
...
```

```
select(df, 9:11) %>% head()
```

```
## # A tibble: 6 x 3
##     hwy fl    class
##   <int> <chr> <chr>
## 1    29 p     compact
...
```

```
select(df, (ncol(df)-2):ncol(df)) %>% head()
```

```
## # A tibble: 6 x 3
##     hwy fl    class
##   <int> <chr> <chr>
## 1    29 p     compact
...
```

```
df1 <- rename(df, mnfc = manufacturer, mod = model)
head(df1)
```

```
## # A tibble: 6 x 11
##   mnfc  mod   displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr> <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi  a4      1.8  1999     4 auto(15)   f        18    29 p     compact
...
```

```
df1 <- select(df, mnfc = manufacturer, mod = model, everything())
head(df1)
```

```
## # A tibble: 6 x 11
##   mnfc  mod   displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr> <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi  a4      1.8  1999     4 auto(15)   f        18    29 p     compact
...
```

mutate() / transmute()

```
df <- mutate(df, `avg miles per gallon` = (cty + hwy) / 2)
head(df)
```

```
## # A tibble: 6 x 12
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi         a4      1.8  1999     4 auto(15)   f        18    29 p     compa~
...
```

```
df <- mutate(df, car = paste(manufacturer, model, sep = " "),
             `cyl / trans` = paste(cyl, " cylinders", " / ", trans, " transmission", sep = ""))
head(df)
```

```
## # A tibble: 6 x 14
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi         a4      1.8  1999     4 auto(15)   f        18    29 p     compa~
...
```

```
df1 <- transmute(df, `avg miles per gallon` = (cty + hwy) / 2)
head(df1)
```

```
## # A tibble: 6 x 1
##   `avg miles per gallon`
##                    <dbl>
## 1                   23.5
...
```

```
df2 <- mutate(df, car = paste(manufacturer, model, sep = " "),
              `cyl / trans` = paste(cyl, " cylinders", " / ", trans, " transmission", sep = ""))
head(df2)
```

```
## # A tibble: 6 x 14
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi         a4      1.8  1999     4 auto(15)   f        18    29 p     compa~
...
```

```
df2 <- transmute(df, car = paste(manufacturer, model, sep = " "),
              `cyl / trans` = paste(cyl, " cylinders", " / ", trans, " transmission", sep = ""))
head(df2)
```

```
## # A tibble: 6 x 2
##   car     `cyl / trans`
##   <chr>   <chr>
## 1 audi a4 4 cylinders / auto(15) transmission
...
```

Manipulate variables(row)

filter(), slice()

```
filter(df, manufacturer == "audi") %>% head()
```

```
## # A tibble: 6 x 14
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi         a4      1.8  1999     4 auto(15)   f        18    29 p     compa~
...
```

```
filter(df, manufacturer == "audi" & year == 1999) %>% head()
```

```
## # A tibble: 6 x 14
##   manufacturer model      displ  year   cyl trans  drv     cty   hwy fl    class
##   <chr>        <chr>      <dbl> <int> <int> <chr>  <chr> <int> <int> <chr> <chr>
## 1 audi         a4           1.8  1999     4 auto(~ f        18    29 p     comp~
...
```

```
df1 <- filter(df, manufacturer == "audi" | manufacturer == "dodge")
head(df1)
```

```
## # A tibble: 6 x 14
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi         a4      1.8  1999     4 auto(15)   f        18    29 p     compa~
...
```

```
df2 <- filter(df, manufacturer %in% c("audi", "dodge"))
head(df2)
```

```
## # A tibble: 6 x 14
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi         a4      1.8  1999     4 auto(15)   f        18    29 p     compa~
...
```

```
filter(df, hwy >= 30) %>% head()
```

```
## # A tibble: 6 x 14
##   manufacturer model  displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr>  <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi         a4       2    2008     4 manual(m6) f        20    31 p     comp~
...
```

```
filter(df, year != 1999) %>% head()
```

```
## # A tibble: 6 x 14
##   manufacturer model      displ  year   cyl trans  drv     cty   hwy fl    class
##   <chr>        <chr>      <dbl> <int> <int> <chr>  <chr> <int> <int> <chr> <chr>
## 1 audi         a4           2    2008     4 manua~ f        20    31 p     comp~
...
```

```
slice(df, 1:5) %>% head()
```

```
## # A tibble: 5 x 14
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi         a4      1.8  1999     4 auto(15)   f        18    29 p     compa~
...
```

```
slice(df, 20:30) %>% head()
```

```
## # A tibble: 6 x 14
##   manufacturer model       displ  year   cyl trans drv     cty   hwy fl    class
##   <chr>        <chr>       <dbl> <int> <int> <chr> <chr> <int> <int> <chr> <chr>
## 1 chevrolet    c1500 subu~   5.3  2008     8 auto~ r        11    15 e     suv
...
```

```
slice(df, (nrow(df)-9):nrow(df)) %>% head()
```

```
## # A tibble: 6 x 14
##   manufacturer model      displ  year   cyl trans  drv     cty   hwy fl    class
##   <chr>        <chr>      <dbl> <int> <int> <chr>  <chr> <int> <int> <chr> <chr>
## 1 volkswagen   new beetle   2    1999     4 auto(~ f        19    26 r     subc~
...
```

arrange

```r
# Sort rows by year (ascending order)
arrange(df, year) %>% head()
```

```
## # A tibble: 6 x 14
##   manufacturer model      displ  year   cyl trans  drv     cty   hwy fl    class
##   <chr>        <chr>      <dbl> <int> <int> <chr>  <chr> <int> <int> <chr> <chr>
## 1 audi         a4           1.8  1999     4 auto(~ f        18    29 p     comp~
...
```

```r
# Sort rows by year (descending order)
arrange(df, desc(year)) %>% head()
```

```
## # A tibble: 6 x 14
##   manufacturer model      displ  year   cyl trans  drv     cty   hwy fl    class
##   <chr>        <chr>      <dbl> <int> <int> <chr>  <chr> <int> <int> <chr> <chr>
## 1 audi         a4           2    2008     4 manua~ f        20    31 p     comp~
...
```

```r
# Sort rows by year (ascending order), cyl and displ
df.sort <- arrange(df, year, cyl, displ)
head(df.sort)
```

```
## # A tibble: 6 x 14
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 honda        civic   1.6  1999     4 manual(m5) f        28    33 r     subco~
...
```

distinct

```r
df.example <- data.frame(id = 1:3, name = c("John", "Max", "Julia"))
df.example <- bind_rows(df.example, slice(df.example, 2)) # create duplicate of 2nd row
df.example <- arrange(df.example, id)
head(df.example)
```

```
##   id  name
## 1  1  John
## 2  2   Max
## 3  2   Max
...
```

```r
# show table without duplicates
distinct(df.example) %>% head()
```

```
##   id  name
## 1  1  John
## 2  2   Max
## 3  3 Julia
```

```r
#  Back to mpg example - lets create a table with duplicates
df.dupl <- select(df, manufacturer, model)
head(df.dupl)
```

```
## # A tibble: 6 x 2
##   manufacturer model
##   <chr>        <chr>
## 1 audi         a4
```

```
...
```

```
#   Keep only unique rows without duplicates
df.nodupl <- distinct(df.dupl)
head(df.nodupl)
```

```
## # A tibble: 6 x 2
##   manufacturer model
##   <chr>        <chr>
## 1 audi         a4
...
```

Sample rows

```
# sample_n() - Filter n randomly selected rows
set.seed(42)
```

```
# 10 randomly selected rows without replacement
sample_n(df, size = 10, replace = F) %>% head()
```

```
## # A tibble: 6 x 14
##   manufacturer model       displ  year   cyl trans drv     cty   hwy fl    class
##   <chr>        <chr>       <dbl> <int> <int> <chr> <chr> <int> <int> <chr> <chr>
## 1 dodge        dakota pic~   3.7  2008     6 manu~ 4        15    19 r     pick~
...
```

```
# 10 randomly selected rows with replacement
sample_n(df, size = 10, replace = T) %>% head()
```

```
## # A tibble: 6 x 14
##   manufacturer model       displ  year   cyl trans drv     cty   hwy fl    class
##   <chr>        <chr>       <dbl> <int> <int> <chr> <chr> <int> <int> <chr> <chr>
## 1 dodge        caravan 2wd   3.8  2008     6 auto~ f        16    23 r     mini~
...
```

```
# sample_frac() - Filter a fraction of randomly selected rows
# 10% of table rows randomly selected
sample_frac(df, size = 0.1, replace = F) %>% head()
```

```
## # A tibble: 6 x 14
##   manufacturer model       displ  year   cyl trans drv     cty   hwy fl    class
##   <chr>        <chr>       <dbl> <int> <int> <chr> <chr> <int> <int> <chr> <chr>
## 1 hyundai      sonata        2.4  2008     4 auto~ f        21    30 r     mids~
...
```

summarise

```
# Calculate average hwy
summarise(df, `mean hwy` = mean(hwy)) %>% head()
```

```
## # A tibble: 1 x 1
##   `mean hwy`
##        <dbl>
## 1       23.4
```

```
# Count table rows, and count distinct car models
summarise(df, rows = n(), `nr models` = n_distinct(model)) %>% head()
```

```
## # A tibble: 1 x 2
##    rows `nr models`
##   <int>       <int>
## 1   234          38
```

```
# Calculate min / max hwy & cty
summarise(df, `min hwy` = min(hwy),
          `min cty` = min(cty),
          `max hwy` = max(hwy),
          `max cty` = max(cty))
```

```
## # A tibble: 1 x 4
##   `min hwy` `min cty` `max hwy` `max cty`
##       <int>     <int>     <int>     <int>
## 1        12         9        44        35
```

group_by()

```
# Group cars by manufacturer
group_by(df, manufacturer) %>% head()
```

```
## # A tibble: 6 x 14
## # Groups:   manufacturer [1]
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
...
```

```
# Combine summarise() & group_by() - summary statistics for grouped data
# Count number of cars for each manufacturer
summarise(group_by(df, manufacturer), cars = n()) %>% head()
```

```
## # A tibble: 6 x 2
##   manufacturer  cars
##   <chr>        <int>
## 1 audi            18
...
```

```
# Calculate mean / min / max hwy for each model
summarise(group_by(df, model),
          `mean hwy` = mean(hwy),
          `min hwy` = min(hwy),
          `max hwy` = max(hwy)) %>% head()
```

```
## # A tibble: 6 x 4
##   model              `mean hwy` `min hwy` `max hwy`
##   <chr>                   <dbl>     <int>     <int>
## 1 4runner 4wd              18.8        17        20
...
```

count()

```
# Count number of table rows
count(df)
```

```
## # A tibble: 1 x 1
##       n
##   <int>
```

```
## 1   234
```

```
# Count number of cars per model
count(group_by(df, model)) %>% head()
```

```
## # A tibble: 6 x 2
## # Groups:   model [6]
##   model                  n
##   <chr>              <int>
...
```

pipe operator %>%

```
df %>%
  filter(manufacturer == "audi") %>%
  count()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1    18
```

```
df %>%
  filter(manufacturer %in% c("dodge", "chevrolet")) %>%
  select(manufacturer, model, year, class) %>%
  head()
```

```
## # A tibble: 6 x 4
##   manufacturer model               year class
##   <chr>        <chr>              <int> <chr>
## 1 chevrolet    c1500 suburban 2wd  2008 suv
...
```

```
df %>%
  group_by(manufacturer, model, class, trans) %>%
  summarise(`mean hwy` = mean(hwy), cars = n()) %>%
  ungroup() %>%
  filter(`mean hwy` > 30) %>%
  arrange(desc(`mean hwy`)) %>%
  head()
```

```
## # A tibble: 6 x 6
##   manufacturer model      class      trans      `mean hwy`  cars
##   <chr>        <chr>      <chr>      <chr>           <dbl> <int>
## 1 honda        civic      subcompact auto(15)         36       2
...
```

pivoting()

```
table.long <- data.frame(id = 1:6,
                         type = c("a", "b", "a", "c", "c", "a"),
                         count = c(20, 50, 45, 15, 12, 5))
head(table.long)
```

```
##   id type count
## 1  1    a    20
```

```
## 2  2    b    50
## 3  3    a    45
...
```

```
table.wide <- pivot_wider(table.long,
                          names_from = type,
                          values_from = count)
head(table.wide)
```

```
## # A tibble: 6 x 4
##      id     a     b     c
##   <int> <dbl> <dbl> <dbl>
## 1     1    20    NA    NA
...
```

```
table.long1 <- pivot_longer(table.wide,
                            cols = c("a", "b", "c"),
                            names_to = "type",
                            values_to = "count",
                            values_drop_na = T)
head(table.long1)
```

```
## # A tibble: 6 x 3
##      id type  count
##   <int> <chr> <dbl>
## 1     1 a        20
...
```

```
df.long <- df %>%
  filter(manufacturer %in% c("jeep", "land rover", "hyundai")) %>%
  select(model, trans, hwy) %>%
  group_by(model, trans) %>%
  summarise(`mean hwy` = mean(hwy)) %>%
  ungroup()
head(df.long)
```

```
## # A tibble: 6 x 3
##   model              trans    `mean hwy`
##   <chr>              <chr>         <dbl>
## 1 grand cherokee 4wd auto(l4)       18.5
...
```

```
df.wide <- df.long %>%
  pivot_wider(names_from = trans,
              values_from = `mean hwy`)
head(df.wide)
```

```
## # A tibble: 4 x 6
##   model              `auto(l4)` `auto(l5)` `auto(s6)` `manual(m5)` `manual(m6)`
##   <chr>                   <dbl>      <dbl>      <dbl>        <dbl>        <dbl>
## 1 grand cherokee 4wd       18.5       17.3         NA           NA           NA
...
```

```
df.long1 <- df.wide %>%
  pivot_longer(-model,  # exclude column "model" and use all remaining columns!!!
               names_to = "trans",
               values_to = "mean hwy",
```

```
              values_drop_na = T)
head(df.long1)
```

```
## # A tibble: 6 x 3
##   model              trans    `mean hwy`
##   <chr>              <chr>         <dbl>
## 1 grand cherokee 4wd auto(l4)       18.5
...
```

separating and uniting

```
dates <- seq.Date(from = as.Date("2021-01-01"), to = as.Date("2021-12-31"), by = "day") # generate date
table <- data.frame(date = dates)
table %>% head()
```

```
##         date
## 1 2021-01-01
## 2 2021-01-02
## 3 2021-01-03
...
```

```
table %>% tail()
```

```
##           date
## 360 2021-12-26
## 361 2021-12-27
## 362 2021-12-28
...
```

separate()

```
table.sep <- table %>%
  separate(data = .,
           col = date,
           into = c("year", "month", "dayofmonth"),
           sep = "-") %>%
  mutate(month = as.numeric(month),
         dayofmonth = as.numeric(dayofmonth)) %>%
  arrange(year, month, dayofmonth)
head(table.sep)
```

```
##   year month dayofmonth
## 1 2021     1          1
## 2 2021     1          2
## 3 2021     1          3
...
```

```
table.sep_ <- table %>%
  separate(data = .,
           col = date,
           into = c("year", "month", "dayofmonth"),
           sep = "-") %>%
  mutate_at(.tbl = .,                          # which table? - . stands for table in the pipe line!
            .vars = c("month", "dayofmonth"), # which variables are mutated?
            .funs = as.numeric) %>%           # which functions is applied?
```

```
  arrange(year, month, dayofmonth)
head(table.sep_)
```

```
##   year month dayofmonth
## 1 2021     1          1
## 2 2021     1          2
## 3 2021     1          3
...
```

unite()

```
table.unite <- table.sep %>%
  # add leading zeros
  mutate(month = str_pad(month, width = 2, side = "left", pad = "0"), # add leading zeros to month
         dayofmonth = str_pad(dayofmonth, width = 2, side = "left", pad = "0")) %>% # add leading zeros
  unite(data = .,
        col = "date",
        year, month, dayofmonth,
        sep = "-") %>%
  arrange(date)
head(table.unite)
```

```
##         date
## 1 2021-01-01
## 2 2021-01-02
## 3 2021-01-03
...
```

```
table.unite_ <- table.sep %>%
  # add leading zeros
  mutate_at(.tbl = .,                                  # which table? - . stands for table in the pipe line
            .vars = c("month", "dayofmonth"),          # which variables are mutated?
            .funs = str_pad, 2, "left", "0") %>%       # which functions is applied? - function parameters
  unite(data = .,
        col = "date",
        year, month, dayofmonth,
        sep = "-") %>%
  arrange(date)
head(table.unite_)
```

```
##         date
## 1 2021-01-01
## 2 2021-01-02
## 3 2021-01-03
...
```

dplyr and tidyr in action

pull() - extract column as vector

```
df %>% pull(hwy) %>% head()
```

```
## [1] 29 29 31 30 26 26
```

```
df %>% pull(hwy) %>% class() %>% head()
```

```
## [1] "integer"
```

```
df %>% select(hwy) %>% class() %>% head()
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

group_by() + mutate()

```
df <- df %>%
  group_by(manufacturer, model) %>%
  mutate(`mean hwy` = mean(hwy)) %>%
  ungroup()
head(df)
```

```
## # A tibble: 6 x 15
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi         a4      1.8  1999     4 auto(15)   f        18    29 p     compa~
...
```

case_when() - case when statements

```
df <- df %>%
  mutate(trans_ = str_sub(string = trans,
                          start = 1,
                          end = 1)) %>%  # extract first letter from trans
  mutate(`transmission type` = case_when(trans_ == "a" ~ "automatic",
                                         trans_ == "m" ~ "manual",
                                         TRUE ~ "NA")) %>%
  select(-trans_)
df %>% count(`transmission type`, trans)  # check car count
```

```
## # A tibble: 10 x 3
##    `transmission type` trans          n
##    <chr>               <chr>      <int>
##  1 automatic           auto(av)       5
...
```

row_number() - add ranks

```
df <- df %>%
  mutate(`car id` = row_number())
head(df)
```

```
## # A tibble: 6 x 17
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi         a4      1.8  1999     4 auto(15)   f        18    29 p     compa~
...
```

```
df <- df %>%
  group_by(manufacturer) %>%
  mutate(`car id1` = row_number()) %>%
```

```
  ungroup()
head(df)
```

```
## # A tibble: 6 x 18
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi         a4      1.8  1999     4 auto(l5)   f        18    29 p     compa~
...
```

Transform table holding flights data

```
df <- hflights
head(df)
```

```
##      Year Month DayofMonth DayOfWeek DepTime ArrTime UniqueCarrier FlightNum
## 5424 2011     1          1         6    1400    1500            AA       428
## 5425 2011     1          2         7    1401    1501            AA       428
## 5426 2011     1          3         1    1352    1502            AA       428
...
```

count number of rows/columns, different flights

```
# one flight is represented with!: UniqueCarrier, FlightNum, TailNum, Year, Month, DayofMonth
nrow(df); ncol(df)
```

```
## [1] 227496
```

```
## [1] 21
```

```
df %>%
  count(UniqueCarrier, FlightNum, TailNum, Year, Month, DayofMonth) %>%
  arrange(desc(n)) %>%
  head()
```

```
##   UniqueCarrier FlightNum TailNum Year Month DayofMonth n
## 1            AA       322  N262AA 2011     7          9 1
## 2            AA       322  N435AA 2011     7          2 1
## 3            AA       322  N463AA 2011     6         18 1
...
```

how many columns begin with word "Taxi"?

```
df %>%
  select(starts_with("Taxi")) %>%
  head()
```

```
##      TaxiIn TaxiOut
## 5424      7      13
## 5425      6       9
## 5426      5      17
...
```

how many flights were flown less than 1000 miles / greater or equal than 1000 miles

```
df %>%
  mutate(dist1000 = case_when(Distance < 1000  ~ "< 1000 miles",
                              Distance >= 1000 ~ ">= 1000 miles")) %>%
  count(dist1000)
```

```
##        dist1000      n
## 1  < 1000 miles 162107
## 2 >= 1000 miles  65389
```

flights per carrier - sort by top to bottom

```
df %>%
  group_by(UniqueCarrier) %>%
  count() %>%
  ungroup() %>%
  arrange(desc(n)) %>%
  head()
```

```
## # A tibble: 6 x 2
##   UniqueCarrier     n
##   <chr>         <int>
## 1 XE            73053
...
```

number of cancelled flights per carrier

```
df %>% count(Cancelled) # 1 for cancelled
```

```
##   Cancelled      n
## 1         0 224523
## 2         1   2973
```

```
df %>%
  filter(Cancelled == 1) %>%
  group_by(UniqueCarrier) %>%
  count() %>%
  ungroup() %>%
  arrange(desc(n)) %>%
  head()
```

```
## # A tibble: 6 x 2
##   UniqueCarrier     n
##   <chr>         <int>
## 1 XE             1132
...
```

percentage of cancelled flights per carrier

```
df %>%
  # count flights break down by cancellation
  group_by(UniqueCarrier, Cancelled) %>%
  count() %>%
  ungroup() %>%
  # calculate total flights
  group_by(UniqueCarrier) %>%
```

```
  mutate(`n tot` = sum(n)) %>%
  ungroup() %>%
  # calculate percentages
  mutate(`n percent %` = (n / `n tot`) * 100) %>%
  # keep only cancelled flights - arrange top to bottom
  filter(Cancelled == 1) %>%
  arrange(desc(`n percent %`)) %>%
  head()
```

```
## # A tibble: 6 x 5
##   UniqueCarrier Cancelled     n `n tot` `n percent %`
##   <chr>             <int> <int>   <int>         <dbl>
## 1 EV                    1    76    2204          3.45
...
```

create column date by combining year + month + dayofmonth (remove this columns)

```
df <- df %>%
  # add leading zeros to month and dayofmonth
  mutate_at(.vars = c("Month", "DayofMonth"),
            .funs = str_pad, 2, "left", "0") %>%
  unite(col = "Date", Year, Month, DayofMonth, sep = "-") %>%
  head()
```

check date range

```
df %>%
  summarise(min = min(Date), max = max(Date), n_distinct = n_distinct(Date))
```

```
##          min        max n_distinct
## 1 2011-01-01 2011-01-06          6
```

```
# count flights per cancellation codes (codes in columns)
# and per carrirs (carriers in rows)
# pivoting required!!!
df %>% count(CancellationCode) # cancellation code "" must have some name other than empty string!
```

```
##   CancellationCode n
## 1                  6
```

```
df %>%
  mutate(CancellationCode = case_when(CancellationCode == "" ~ "0",  # this is not cancelled flight!!!
                                      TRUE ~ CancellationCode)) %>%
  group_by(UniqueCarrier, CancellationCode) %>%
  count() %>%
  ungroup() %>%
  pivot_wider(names_from = CancellationCode,
              values_from = n) %>%
  head()
```

```
## # A tibble: 1 x 2
##   UniqueCarrier   `0`
##   <chr>         <int>
## 1 AA                6
```

Column-wise operations: across()

```
mpg <- ggplot2::mpg # mpg data
```

summarise() & across()

```
#  count distinct values in each column
mpg %>%
  summarise(across(.cols = everything(), # which columns:   all columns
                   .fns = n_distinct))   # which function:  count distinct/unique values
```

```
## # A tibble: 1 x 11
##   manufacturer model displ  year   cyl trans   drv   cty   hwy    fl class
##          <int> <int> <int> <int> <int> <int> <int> <int> <int> <int> <int>
## 1           15    38    35     2     4    10     3    21    27     5     7
```

```
mpg %>%
  summarise(across(everything(),
                   n_distinct))
```

```
## # A tibble: 1 x 11
##   manufacturer model displ  year   cyl trans   drv   cty   hwy    fl class
##          <int> <int> <int> <int> <int> <int> <int> <int> <int> <int> <int>
## 1           15    38    35     2     4    10     3    21    27     5     7
```

```
#  calculate mean values for selected columns (list of columns)
mpg %>%
  summarise(across(c(displ, cty, hwy),
                   mean))
```

```
## # A tibble: 1 x 3
##   displ   cty   hwy
##   <dbl> <dbl> <dbl>
## 1  3.47  16.9  23.4
```

```
#  calculate median value for all numeric columns
mpg %>%
  summarise(across(where(is.numeric),   # "where" clause supports conditions for columns selection!
                   median))
```

```
## # A tibble: 1 x 5
##   displ  year   cyl   cty   hwy
##   <dbl> <dbl> <dbl> <dbl> <dbl>
## 1   3.3 2004.     6    17    24
```

```
#  calculate distinct values of character columns
mpg %>%
  summarise(across(where(is.character), n_distinct))
```

```
## # A tibble: 1 x 6
##   manufacturer model trans   drv    fl class
##          <int> <int> <int> <int> <int> <int>
## 1           15    38    10     3     5     7
```

```
#  apply more than one function across multiple columns
#   - calculate mean and median of all numeric columns
mpg %>%
  summarise(across(where(is.numeric),
```

```
                  list(avg = ~mean(.x, na.rm = T),     # multiple functions: provided as a list of fun
                       med = ~median(.x, na.rm = T))))
```

```
## # A tibble: 1 x 10
##   displ_avg displ_med year_avg year_med cyl_avg cyl_med cty_avg cty_med hwy_avg
##       <dbl>     <dbl>    <dbl>    <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1      3.47       3.3    2004.    2004.    5.89       6    16.9      17    23.4
...
```

```
avgmed <- list(avg = ~mean(.x, na.rm = T), med = ~median(.x, na.rm = T))
mpg %>%
  summarise(across(where(is.numeric), avgmed))
```

```
## # A tibble: 1 x 10
##   displ_avg displ_med year_avg year_med cyl_avg cyl_med cty_avg cty_med hwy_avg
##       <dbl>     <dbl>    <dbl>    <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1      3.47       3.3    2004.    2004.    5.89       6    16.9      17    23.4
...
```

```
#  control names of output columns
mpg %>%
  summarise(across(where(is.numeric), avgmed, .names = "{.fn}:{.col}")) # argument used for column name
```

```
## # A tibble: 1 x 10
##   `avg:displ` `med:displ` `avg:year` `med:year` `avg:cyl` `med:cyl` `avg:cty`
##         <dbl>       <dbl>      <dbl>      <dbl>     <dbl>     <dbl>     <dbl>
## 1        3.47         3.3      2004.      2004.      5.89         6      16.9
...
```

```
#  use multiple conditions for column selection
mpg %>%
  summarise(across(where(is.numeric) & ends_with("y"), median))
```

```
## # A tibble: 1 x 2
##     cty   hwy
##   <dbl> <dbl>
## 1    17    24
```

summarise() ~ group_by() & across()

```
#  calculate sum of all numeric columns break down by car model
mpg %>%
  group_by(model) %>%
  summarise(across(where(is.numeric),
                   sum)) %>%
  ungroup() %>%
  head()
```

```
## # A tibble: 6 x 6
##   model              displ  year   cyl   cty   hwy
##   <chr>              <dbl> <int> <int> <int> <int>
## 1 4runner 4wd         20.9 12012    34    91   113
...
```

```
# calculate mean value for selected columns break down by car manufacturer & model
mpg %>%
  group_by(manufacturer, model) %>%
```

```
  summarise(across(c(displ, cty, hwy), mean)) %>%
  ungroup() %>%
  head()
```

```
## # A tibble: 6 x 5
##   manufacturer model              displ   cty   hwy
##   <chr>        <chr>              <dbl> <dbl> <dbl>
## 1 audi         a4                  2.33  18.9  28.3
...
```

mutate() & across()

```
#  round up (ceiling) all numeric columns
mpg %>%
  mutate(across(where(is.numeric), ~ceiling(.x))) %>%
  head()
```

```
## # A tibble: 6 x 11
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <dbl> <dbl> <chr>      <chr> <dbl> <dbl> <chr> <chr>
## 1 audi         a4        2  1999     4 auto(15)   f        18    29 p     compa~
...
```

```
#  convert all character columns to upper case
mpg %>%
  mutate(across(where(is.character),~str_to_upper(.x))) %>%
  head()
```

```
## # A tibble: 6 x 11
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 AUDI         A4      1.8  1999     4 AUTO(L5)   F        18    29 P     COMPA~
...
```

mutate() ~ group_by() & across()

```
#  calculate mean value for all numeric columns break down by car manufacturer
#   - aggregate mean value of numeric columns for each manufacturer
#   - keep all the rows!
mpg %>%
  group_by(manufacturer) %>%
  mutate(across(where(is.numeric) & -year, # column "year" is removed from calculation!
                ~mean(.x, na.rm = T),
                .names = "{.col}_avg_manufacturer")) %>%
  ungroup() %>%
  head()
```

```
## # A tibble: 6 x 15
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi         a4      1.8  1999     4 auto(15)   f        18    29 p     compa~
...
```

```
# if_any() / if_all() with filter()
# if_any() : keeps the rows where the predicate is true for at least one selected column
# if_all() : keeps the rows where the predicate is true for all selected columns
```

```
starwars <- dplyr::starwars # star wars data set
?starwars
```

```
#  filter rows where at least one column doesn't have NA value
starwars %>%
  filter(if_any(.cols = everything(), .fns = ~ !is.na(.x))) %>%
  head()
```

```
## # A tibble: 6 x 14
##   name      height  mass hair_color skin_color eye_color birth_year sex   gender
##   <chr>      <int> <dbl> <chr>      <chr>      <chr>          <dbl> <chr> <chr>
## 1 Luke Sky~    172    77 blond      fair       blue            19   male  mascu~
...
```

```
#  filter rows where all columns don't have NA value
starwars %>%
  filter(if_all(.cols = everything(), .fns = ~ !is.na(.x))) %>%
  head()
```

```
## # A tibble: 6 x 14
##   name      height  mass hair_color skin_color eye_color birth_year sex   gender
##   <chr>      <int> <dbl> <chr>      <chr>      <chr>          <dbl> <chr> <chr>
## 1 Luke Sky~    172    77 blond      fair       blue            19   male  mascu~
...
```

```
#  filter rows where column "cty" or "hwy" have values greater than 20
mpg %>%
  filter(if_any(c(cty, hwy), ~ . > 20)) %>%  # condition written as function
  head()
```

```
## # A tibble: 6 x 11
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi         a4      1.8  1999     4 auto(15)   f        18    29 p     compa~
...
```

```
#  filter rows where column "cty" and "hwy" have values greater than 20
mpg %>%
  filter(if_all(c(cty, hwy), ~ . > 20)) %>%
  head()
```

```
## # A tibble: 6 x 11
##   manufacturer model  displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr>  <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi         a4       1.8  1999     4 manual(m5) f        21    29 p     comp~
...
```