02-dplyr

조근수

2020 12 23

dplyr basic

dplyr 패키지의 `filter()`, `select()`, `mutate()`, `summarise()`, `group_by()` 함수로 데이터를 가공합니다.

가장 먼저 패키지를 불러오고 데이터의 요약정보를 확인한다.

```
library(nycflights13)
library(tidyverse)
glimpse(flights)
```

```
## Rows: 336,776
## Columns: 19
## $ year           <int> 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013...
## $ month          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
## $ day            <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
## $ dep_time       <int> 517, 533, 542, 544, 554, 554, 555, 557, 557, 558, 55...
## $ sched_dep_time <int> 515, 529, 540, 545, 600, 558, 600, 600, 600, 600, 60...
## $ dep_delay      <dbl> 2, 4, 2, -1, -6, -4, -5, -3, -3, -2, -2, -2, -2, -2,...
## $ arr_time       <int> 830, 850, 923, 1004, 812, 740, 913, 709, 838, 753, 8...
## $ sched_arr_time <int> 819, 830, 850, 1022, 837, 728, 854, 723, 846, 745, 8...
## $ arr_delay      <dbl> 11, 20, 33, -18, -25, 12, 19, -14, -8, 8, -2, -3, 7,...
## $ carrier        <chr> "UA", "UA", "AA", "B6", "DL", "UA", "B6", "EV", "B6"...
## $ flight         <int> 1545, 1714, 1141, 725, 461, 1696, 507, 5708, 79, 301...
## $ tailnum        <chr> "N14228", "N24211", "N619AA", "N804JB", "N668DN", "N...
## $ origin         <chr> "EWR", "LGA", "JFK", "JFK", "LGA", "EWR", "EWR", "LG...
## $ dest           <chr> "IAH", "IAH", "MIA", "BQN", "ATL", "ORD", "FLL", "IA...
## $ air_time       <dbl> 227, 227, 160, 183, 116, 150, 158, 53, 140, 138, 149...
## $ distance       <dbl> 1400, 1416, 1089, 1576, 762, 719, 1065, 229, 944, 73...
## $ hour           <dbl> 5, 5, 5, 5, 6, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 5, 6, 6...
## $ minute         <dbl> 15, 29, 40, 45, 0, 58, 0, 0, 0, 0, 0, 0, 0, 0, 0, 59...
## $ time_hour      <dttm> 2013-01-01 05:00:00, 2013-01-01 05:00:00, 2013-01-0...
```

- *dttm*은 날짜+시간을 나타낸다.

Filter()

`filter()` 는 조건에 따라 **행(row)** 을 추출한다. 함수 내에서 연산 적용이 가능한데, AND 조건은 콤마(`,`)로 구분하거나 `&` 를 사용하고, OR 조건은 `|` 를 사용한다. 또한 `x %in% y` 는 y의 값을 x에서 찾아준다. 비교 연산자로 `>`, `>=`, `<`, `<=`, `!=`, `==` 이 사용가능하며 `is.na()` 를 사용해 결측치 확인도 가능하다.

```
# 1월 1일 데이터 추출
flights %>% filter(month==1, day == 1) %>% head(3)
```

```
## # A tibble: 3 x 19
##    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
## 1  2013     1     1      517            515         2      830            819
## 2  2013     1     1      533            529         4      850            830
## 3  2013     1     1      542            540         2      923            850
## # ... with 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

```
# 출발 지연이 120분 이상이거나 도착지연이 120분 이상인 데이터
flights %>% filter(!(arr_delay > 120 | dep_delay > 120)) %>% head(3)
```

```
## # A tibble: 3 x 19
##    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
## 1  2013     1     1      517            515         2      830            819
## 2  2013     1     1      533            529         4      850            830
## 3  2013     1     1      542            540         2      923            850
## # ... with 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

```
# 11월이거나 12월 데이터 추출
flights %>% filter(month %in% c(11,12)) %>% head(3)
```

```
## # A tibble: 3 x 19
##    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
## 1  2013    11     1        5           2359         6      352            345
## 2  2013    11     1       35           2250       105      123           2356
## 3  2013    11     1      455            500        -5      641            651
## # ... with 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

```
# 출발시간에 결측치가 존재하는 데이터
flights %>% filter(is.na(dep_time)) %>% head(3)
```

```
## # A tibble: 3 x 19
##    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
## 1  2013     1     1       NA           1630        NA       NA           1815
## 2  2013     1     1       NA           1935        NA       NA           2240
## 3  2013     1     1       NA           1500        NA       NA           1825
## # ... with 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

arrange()

`arrange()` 는 조건에 따라 __행(row)__을 추출하기때문에 `filter()` 와 유사하나 데이터정렬도 가능하다.

```
# 날짜를 역순서로 정렬한다.
flights %>% arrange(desc(year), desc(month), desc(day)) %>% head(3)
```

```
## # A tibble: 3 x 19
##    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
## 1  2013    12    31       13           2359        14      439            437
## 2  2013    12    31       18           2359        19      449            444
## 3  2013    12    31       26           2245       101      129           2353
## # ... with 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

```
# dep_delay 가 가장 큰 데이터를 확인한다.
flights %>% arrange(dep_delay) %>% head(1)
```

```
## # A tibble: 1 x 19
##    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
## 1  2013    12     7     2040           2123       -43       40           2352
## # ... with 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

select()

`select()` 는 조건에 따라 **열(column)** 을 추출한다.
복수의 열을 추출할 땐 콤마(`,`)를 사용하며 인접한 열을 추출할 땐 `:` 를 사용한다. 또한 지정한 열 이외의 다른 열을 추출하고 싶을 땐 `-` 를 사용한다. 또한 다음과 같은 함수를 이용하면 다양한 방법으로 열을 추출할 수 있다.

함수	설명
starts_with("abc")	matches names that begin with "abc".
ends_with("xyz")	matches names that end with "xyz".
contains("ijk")	matches names that contain "ijk".
num_range("x", 1:3)	matches x1, x2 and x3.

```
# 년, 월, 일만 출력한다.
flights %>% select(year, month, day)  %>% head(3)
```

```
## # A tibble: 3 x 3
##    year month   day
##   <int> <int> <int>
## 1  2013     1     1
## 2  2013     1     1
## 3  2013     1     1
```

```
# 년부터 일까지 출력한다. (위와 동일)
flights %>% select(year:day)  %>% head(3)
```

```
## # A tibble: 3 x 3
##    year month   day
##   <int> <int> <int>
## 1  2013     1     1
## 2  2013     1     1
## 3  2013     1     1
```

```
# 년월일을 제외한 데이터를 출력한다.
flights %>% select(-(year:day)) %>% head(3)
```

```
## # A tibble: 3 x 16
##   dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay carrier
##      <int>          <int>     <dbl>    <int>          <int>     <dbl> <chr>
## 1      517            515         2      830            819        11 UA
## 2      533            529         4      850            830        20 UA
## 3      542            540         2      923            850        33 AA
## # ... with 9 more variables: flight <int>, tailnum <chr>, origin <chr>,
## #   dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>,
## #   time_hour <dttm>
```

```
# 원하는 데이터를 앞에 출력시킨다.
flights %>% select(time_hour, air_time, everything()) %>% head(3)
```

```
## # A tibble: 3 x 19
##   time_hour           air_time  year month   day dep_time sched_dep_time
##   <dttm>                 <dbl> <int> <int> <int>    <int>          <int>
## 1 2013-01-01 05:00:00      227  2013     1     1      517            515
## 2 2013-01-01 05:00:00      227  2013     1     1      533            529
## 3 2013-01-01 05:00:00      160  2013     1     1      542            540
## # ... with 12 more variables: dep_delay <dbl>, arr_time <int>,
## #   sched_arr_time <int>, arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, distance <dbl>, hour <dbl>,
## #   minute <dbl>
```

```
#변수명이 arr로 시작하는 변수 출력
select(flights, starts_with("arr")) %>% head(3)
```

```
## # A tibble: 3 x 2
##   arr_time arr_delay
##      <int>     <dbl>
## 1      830        11
## 2      850        20
## 3      923        33
```

```
#변수명이 time으로 끝나는 변수 출력
select(flights, ends_with("time")) %>% head(3)
```

```
## # A tibble: 3 x 5
##   dep_time sched_dep_time arr_time sched_arr_time air_time
##      <int>          <int>    <int>          <int>    <dbl>
## 1      517            515      830            819      227
## 2      533            529      850            830      227
## 3      542            540      923            850      160
```

```
#변수명에 dep를 포함하는 변수 출력
select(flights, contains("dep", ignore.case = F)) %>% head(3) # 디폴트는 ignore.case == T
```

```
## # A tibble: 3 x 3
##   dep_time sched_dep_time dep_delay
##      <int>          <int>     <dbl>
## 1      517            515         2
## 2      533            529         4
## 3      542            540         2
```

벡터를 입력받는 형식으로도 `select()`를 사용할 수 있다.

```
vars <- c('dep_time', 'dep_delay', 'aa')
# vars에 변수명이 모두 존재할때만 출력(없는것이 있다면 오류)
flights %>% select(all_of(vars)) %>% head(1)
```

```
## Error: Can't subset columns that don't exist.
##  [31mx [39m Column `aa` doesn't exist.
```

```
# vars에 일부만 존재한다면 출력
flights %>% select(any_of(vars)) %>% head(1)
```

```
## # A tibble: 1 x 2
##   dep_time dep_delay
##      <int>     <dbl>
## 1      517         2
```

```
# vars에 일부만 존재한다면 출력(Warning과 함께 출력됨)
flights %>% select(one_of(vars)) %>% head(1)
```

```
## Warning: Unknown columns: `aa`
```

```
## # A tibble: 1 x 2
##   dep_time dep_delay
##      <int>     <dbl>
## 1      517         2
```

mutate()

데이터에 새로운 변수(열)을 생성할 때 사용하며, 함수내에서 바로 사용가능한 장점이 있다.

```
# 3개의 새로운 열을 생성, 생성된 변수가 함수내에서 바로 적용됨
flights %>% select(dep_delay, arr_delay, air_time) %>%
  mutate(gain = dep_delay - arr_delay, hours = air_time / 60, gain_per_hour = gain / hours)
```

```
## # A tibble: 336,776 x 6
##    dep_delay arr_delay air_time  gain hours gain_per_hour
##        <dbl>     <dbl>    <dbl> <dbl> <dbl>         <dbl>
##  1         2        11      227    -9 3.78          -2.38
##  2         4        20      227   -16 3.78          -4.23
##  3         2        33      160   -31 2.67         -11.6
##  4        -1       -18      183    17 3.05           5.57
##  5        -6       -25      116    19 1.93           9.83
##  6        -4        12      150   -16 2.5           -6.4
##  7        -5        19      158   -24 2.63          -9.11
##  8        -3       -14       53    11 0.883         12.5
##  9        -3        -8      140     5 2.33           2.14
## 10        -2         8      138   -10 2.3           -4.35
## # ... with 336,766 more rows
```

만약 새로운 변수만 포함된 데이터를 만들고 싶을 경우, `transmute()` 를 사용한다.

```
flights %>%
  transmute(gain = dep_delay - arr_delay, hours = air_time / 60, gain_per_hour = gain / hours)
```

```
## # A tibble: 336,776 x 3
##     gain hours gain_per_hour
##    <dbl> <dbl>         <dbl>
##  1    -9 3.78          -2.38
##  2   -16 3.78          -4.23
##  3   -31 2.67         -11.6
##  4    17 3.05           5.57
##  5    19 1.93           9.83
##  6   -16 2.5           -6.4
##  7   -24 2.63          -9.11
##  8    11 0.883         12.5
##  9     5 2.33           2.14
## 10   -10 2.3           -4.35
## # ... with 336,766 more rows
```

`row_rank()` , `min_rank()` 와 같은 **window function** 은 링크 참고

summarise()

`summarise()` 는 `mean()` , `sd()` , `var()` , `median()` 등의 함수를 지정하여 기초 통계량을 구할 수 있다. 결과값은 데이터 프레임 형식으로 출력된다. 주로 `group_by()` 와 함께 사용된다.

```
# 날짜별로 출발지연시간 평균을 구한다.
flights %>% group_by(year, month, day) %>% summarise(delay=mean(dep_delay, na.rm=TRUE))
```

```
## `summarise()` regrouping output by 'year', 'month' (override with `.groups` argument)
```

```
## # A tibble: 365 x 4
## # Groups:   year, month [12]
##     year month   day delay
##    <int> <int> <int> <dbl>
##  1  2013     1     1 11.5
##  2  2013     1     2 13.9
##  3  2013     1     3 11.0
##  4  2013     1     4  8.95
##  5  2013     1     5  5.73
##  6  2013     1     6  7.15
##  7  2013     1     7  5.42
##  8  2013     1     8  2.55
##  9  2013     1     9  2.28
## 10  2013     1    10  2.84
## # ... with 355 more rows
```

```
flights %>%
  group_by(dest) %>%
  summarise(count = n(),
            dist = mean(distance, na.rm = TRUE),
            delay = mean(arr_delay, na.rm = TRUE)) %>%
  filter(count > 20, dest != "HNL")
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 96 x 4
##    dest  count  dist delay
##    <chr> <int> <dbl> <dbl>
##  1 ABQ     254 1826   4.38
##  2 ACK     265  199   4.85
##  3 ALB     439  143  14.4
##  4 ATL   17215  757. 11.3
##  5 AUS    2439 1514.  6.02
##  6 AVL     275  584.  8.00
##  7 BDL     443  116   7.05
##  8 BGR     375  378   8.03
##  9 BHM     297  866. 16.9
## 10 BNA    6333  758. 11.8
## # ... with 86 more rows
```