W3 Notes

K Arnold

Hotel Bookings Dataset

See Application Exercise for details

```
hotels <- paste0(
  "https://raw.githubusercontent.com/",
  "rfordatascience/tidytuesday/",
  "master/data/2020/2020-02-11/hotels.csv"
  ) %>%
  read_csv()
```

Inline R code

R Markdown input

```
The `hotels` dataset has data about `r nrow(hotels)` bookings.
```

🔽

> The `hotels` dataset has data about 119390 bookings.

select to keep variables

```
hotels %>%
  select(hotel, lead_time)
```

`select` to exclude variables

```r
hotels %>%
  select(-agent)
```

`select` variables with certain characteristics

```
hotels %>%
  select(starts_with("arrival"))
```

`arrange` in ascending / descending order

```
hotels %>%
  select(adults, children, babies) %>%
  arrange(babies)
```

```
hotels %>%
  select(adults, children, babies) %>%
  arrange(desc(babies))
```

`slice` for certain row numbers

```
# first five
hotels %>%
  slice(1:5)
```

Alternative:

```
hotels %>%
  slice_head(5)
```

Comments

In R, as in Python, # can be used to comment a line to describe it or to (temporarily) disable it.
(Don't leave commented-out code in your reports.)

```
hotels %>%
  # slice the first five rows  # this line is a comment
  #select(hotel) %>%           # this one doesn't run
  slice(1:5)                   # this line runs
```

`slice` for certain row numbers

```r
# last five
last_row <- nrow(hotels)         # nrow() gives the number of rows in a data frame
hotels %>%
  slice((last_row - 4):last_row)
```

(but `slice_tail(5)` would be easier.)

`filter` to select a subset of rows

```r
# bookings in City Hotels
hotels %>%
  filter(hotel == "City Hotel")
```

`filter` for many conditions at once

```
hotels %>%
  filter(
    adults == 0,
    children >= 1
    ) %>%
  select(adults, babies, children)
```

`filter` for more complex conditions

```
# bookings with no adults and some children or babies in the room
hotels %>%
  filter(
    adults == 0,
    children >= 1 | babies >= 1     # | means or
    ) %>%
  select(adults, babies, children)
```

Logical operators in R

operator	definition	operator	definition
`<`	less than	`x | y`	`x` OR `y`
`<=`	less than or equal to	`is.na(x)`	test if `x` is `NA`
`>`	greater than	`!is.na(x)`	test if `x` is not `NA`
`>=`	greater than or equal to	`x %in% y`	test if `x` is in `y`
`==`	exactly equal to	`!(x %in% y)`	test if `x` is not in `y`
`!=`	not equal to	`!x`	not `x`
`x & y`	`x` AND `y`		

mutate to add a new variable

```r
hotels %>%
  mutate(kids = children + babies) %>%
  select(children, babies, kids) %>%
  arrange(desc(kids))
```

Kids in resort and city hotels

```r
# Resort Hotel
hotels %>%
  mutate(kids = children + babies) %>%
  filter(
    kids >= 1,
    hotel == "Resort Hotel"
    ) %>%
  select(hotel, kids)
```

```r
# City Hotel
hotels %>%
  mutate(kids = children + babies) %>%
  filter(
    kids >= 1,
    hotel == "City Hotel"
    )  %>%
  select(hotel, kids)
```

What is happening in the following chunk?

```
hotels %>%
  mutate(kids = children + babies) %>%
  count(hotel, kids) %>%
  mutate(prop = n / sum(n))
```

```
## # A tibble: 12 x 4
##    hotel         kids     n       prop
##    <chr>        <dbl> <int>      <dbl>
##  1 City Hotel       0 73923 0.619
##  2 City Hotel       1  3263 0.0273
##  3 City Hotel       2  2056 0.0172
##  4 City Hotel       3    82 0.000687
##  5 City Hotel       9     1 0.00000838
##  6 City Hotel      10     1 0.00000838
##  7 City Hotel      NA     4 0.0000335
##  8 Resort Hotel     0 36131 0.303
##  9 Resort Hotel     1  2183 0.0183
## 10 Resort Hotel     2  1716 0.0144
## 11 Resort Hotel     3    29 0.000243
## 12 Resort Hotel    10     1 0.00000838
```

summarise for summary stats

```
# mean average daily rate for all bookings
hotels %>%
  summarise(mean_adr = mean(adr))
```

`summarise` for summary stats

```r
# mean average daily rate for all bookings
hotels %>%
  summarise(mean_adr = mean(adr))
```

`summarise()` changes the data frame entirely, it collapses rows down to a single summary statistics, and removes all columns that are irrelevant to the calculation.

`summarise()` also lets you get away with being sloppy and not naming your new column, but that's not recommended!

❌

```
hotels %>%
  summarise(mean(adr))
```

✅

```
hotels %>%
  summarise(mean_adr = mean(adr))
```

`group_by` for grouped operations

```r
# mean average daily rate for all booking at city and resort hotels
hotels %>%
  group_by(hotel) %>%
  summarise(mean_adr = mean(adr))
```

Calculating frequencies

The following two give the same result, so `count` is simply short for `group_by` then determine frequencies

```
hotels %>%
  group_by(hotel) %>%
  summarise(n = n())
```

```
hotels %>%
  count(hotel)
```

Multiple summary statistics

`summarise` can be used for multiple summary statistics as well

```
hotels %>%
  summarise(
    min_adr = min(adr),
    mean_adr = mean(adr),
    median_adr = median(adr),
    max_adr = max(adr)
    )
```