Data Manipulation 2: Exercises

These exercises accompany the Data Manipulation Part 2 tutorial: http://rpubs.com/NateByers/DataManip2. These exercises use data frames from the `region5air` library. Run the following code to clean out your global environment and load the data you need:

```
rm(list = ls())
library(dplyr)
library(region5air)
data(airdata)
data(chicago_air)
```

Exercises

1. Use the `mutate()` function to add a column named "violation" to the `chicago_air` dataset that indicates when the ozone value was above 0.070 ppm. **Hint**: you will make the new column using a single equal sign, `violation =`. The value you want is a logical comparison: `violation =` ozone values greater than 0.07.

Solution 1

2. Use the `group_by()` function to group the `airdata` data frame by site, parameter, and duration. Then use `filter()` to make a data frame called `max_params` with maximum values for each monitor/parameter/duration combination. **Hint**: in `filter()` you will use `value ==`. You want to keep numbers in the "value" column that equal the maximum for each group (use `max()`, and don't forget `na.rm = TRUE`).

Solution 2

3. In exercise 2 we grouped the `airdata` data frame by site, parameter, and duration. Use `summarize()` to create a data frame called `mean_params` with mean values for each monitor/parameter/duration combination. **Hint**: you will use a single `=` symbol, `mean_value =`. You want to use the `mean()` function on numbers in the "value"" column for each group (don't forget `na.rm = TRUE`).

Solution 3

Advanced Exercise

4. Use the pipe, `%>%`, to string together this series of `dplyr` operations: subset `airdata` down to ozone (44201); group by site, datetime, and poc; use `summarize` to find the mean value for each site/datetime combination; and filter down to the top 4 values for each year. *Note*: this data set only has 1 hour values. This exercise would make more sense with 8 hour data, for comparison to the NAAQS.

Solution 4

Solutions

```
chicago_air <- mutate(chicago_air, violation = ozone > 0.070)
head(chicago_air)
```

Solution 1

```
##         date ozone temp solar month weekday violation
## 1 2013-01-01 0.032   17  0.65     1       3     FALSE
## 2 2013-01-02 0.020   15  0.61     1       4     FALSE
## 3 2013-01-03 0.021   28  0.17     1       5     FALSE
## 4 2013-01-04 0.028   18  0.62     1       6     FALSE
## 5 2013-01-05 0.025   26  0.48     1       7     FALSE
## 6 2013-01-06 0.026   36  0.47     1       1     FALSE
```

Back to exercises

```
airdata <- group_by(airdata, site, parameter, duration)

max_params <- filter(airdata, value == max(value, na.rm = TRUE))
```

Solution 2 Back to exercises

```
mean_params <- summarize(airdata, mean_value = mean(value, na.rm = TRUE))
```

Solution 3 Back to exercises

Solution 4 Without pipes, this is how we would do it.

```
# subset down to ozone
ozone <- filter(airdata, parameter == "44201")

# group ozone by site and datetime
ozone <- group_by(ozone, site, datetime)

# replace the value column with the mean value for each monitor at a datetime
# --taking the mean when there are multiple pocs
ozone <- summarize(ozone, value = mean(value, na.rm = TRUE))

# create a year column so you can group by year
ozone <- mutate(ozone, year = substr(datetime, start = 1, stop = 2))

# group now on site and year
ozone <- group_by(ozone, site, year)
```

```r
# filter down to the fourth highest value when the value column
# is ranked in descending order per site and year
ozone_4rth_high <- filter(ozone, row_number(desc(value)) == 4)
```

Here are the same operations with pipes.

```r
ozone_4rth_high <- airdata %>%
  filter(parameter == "44201") %>%
  group_by(site, datetime) %>%
  summarize(value = mean(value, na.rm = TRUE)) %>%
  mutate(year = substr(datetime, 1, 4)) %>%
  group_by(site, year) %>%
  filter(row_number(desc(value)) == 4)
```

Back to exercises