Generating Data and Manipulating Objects (Part2)

Use dplyr to manipulate data dplyr is primarily a set of functions designed to enable dataframe manipulation in an intuitive, user-friendly way. Data analysts typically use dplyr in order to transform existing datasets into a format better suited for some particular type of analysis, or data visualization.

"tibble" refers to a data frame that has the "tbl_df" class. Tibble is the central data structure for the set of packages known as the tidyverse, including dplyr, ggplot2, tidyr, and readr.

```
library(readr)
BirdNest <- read_csv("BirdNest.csv")  # read data from csv file
```

0. import data

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Species = col_character(),
##   Common = col_character(),
##   Page = col_double(),
##   Length = col_double(),
##   Nesttype = col_character(),
##   Location = col_character(),
##   No.eggs = col_double(),
##   Color = col_double(),
##   Incubate = col_double(),
##   Nestling = col_double(),
##   Totcare = col_double(),
##   Closed. = col_double()
## )
```

```
# If you did not set work directory or would like to read in file from other folder instead of working
```

1. select To select columns or drop columns of a data frame, use select().

- Select desired variables

```
# select four columns: Length, Nesttype, Location, No.eggs from original data, return the first six rows
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
select <- select(BirdNest, Length, Nesttype, Location, No.eggs)
head(select)
```

```
## # A tibble: 6 x 4
##   Length Nesttype Location No.eggs
##    <dbl> <chr>    <chr>      <dbl>
## 1   20   cup      decid        3.5
## 2   20   cavity   decid        3.5
## 3   20   cavity   decid        4.5
## 4   22.5 cavity   decid        4.5
## 5   17   cavity   decid        4.5
## 6   17   cup      bridge       4.5
```

- Drop undesired variables

```
# remove two columns: Species and Common from original data
drop <- select(BirdNest,-(Species:Common))
head(drop)
```

```
## # A tibble: 6 x 10
##    Page Length Nesttype Location No.eggs Color Incubate Nestling Totcare Closed.
##   <dbl>  <dbl> <chr>    <chr>      <dbl> <dbl>    <dbl>    <dbl>   <dbl>   <dbl>
## 1   360   20   cup      decid        3.5     1     17       17      34         0
## 2   368   20   cavity   decid        3.5     1     15.5     17      32.5       1
## 3   372   20   cavity   decid        4.5     1     15       15      30         1
## 4   372   22.5 cavity   decid        4.5     1     14       16.5    30.5       1
## 5   374   17   cavity   decid        4.5     1     14       14      28         1
## 6   378   17   cup      bridge       4.5     0     16       15.5    31.5       0
```

- Note: Other functions for variable selection:

usage	summary
-	Select everything but
:	Select range
contains()	Select columns whose name contains a character string
start_with()	Select columns whose name starts with a string
ends_with()	Select columns whose name ends with a string
matches()	Select columns whose name matches a regular expression
one_of()	Select columns whose names are in a group of names

```
# select variables contain "nest" (default case insensitive)
head(select(BirdNest, contains("nest")))
```

```
## # A tibble: 6 x 2
##   Nesttype Nestling
##   <chr>       <dbl>
## 1 cup          17
## 2 cavity       17
## 3 cavity       15
## 4 cavity       16.5
## 5 cavity       14
## 6 cup          15.5
```

2. filter Use **filter** to select rows that meet criteria. you may use %in% when you have specified levels that would like the variable to be filtered on, however, when your criteria is blurred, you may use "grepl", to roughly search pattern in the variable and return the rows meet criteria.

```
# select rows with Length more than 30
filter(BirdNest, Length>30)
```

```
## # A tibble: 4 x 12
##   Species Common  Page Length Nesttype Location No.eggs Color Incubate Nestling
##   <chr>   <chr>  <dbl>  <dbl> <chr>    <chr>      <dbl> <dbl>    <dbl>    <dbl>
## 1 Aphelo~ Scrub~   404   30.5 cup      decid        4.5     1       16     18.5
## 2 Nucifr~ Clark~   410   31.5 cup      conif        3       1       17     19.5
## 3 Periso~ Cray ~   410   30.5 cup      decid        3.5     1       17     15
## 4 Toxost~ Calif~   476   30.5 cup      shrub        3.5     1       14     13
## # ... with 2 more variables: Totcare <dbl>, Closed. <dbl>
```

```
# select rows with No.eggs more than 6 and Location in "decid"
filter(BirdNest, No.eggs>6 , Location %in% c("decid")) ## or Location=="decid" if just one level
```

```
## # A tibble: 3 x 12
##   Species Common  Page Length Nesttype Location No.eggs Color Incubate Nestling
##   <chr>   <chr>  <dbl>  <dbl> <chr>    <chr>      <dbl> <dbl>    <dbl>    <dbl>
## 1 Parus ~ Black~   424     13 cavity   decid        7       1       12       16
## 2 Sitta ~ White~   434     14 cavity   decid        6.5     1       12       14
## 3 Troglo~ House~   438     12 cavity   decid        7       1       13       15
## # ... with 2 more variables: Totcare <dbl>, Closed. <dbl>
```

Question: Could you filter the rows with levels contains "Jay" in Common column? (hint: use grepl function)

```
filter(BirdNest ,grepl("Jay",Common))
```

```
## # A tibble: 4 x 12
##   Species Common  Page Length Nesttype Location No.eggs Color Incubate Nestling
##   <chr>   <chr>  <dbl>  <dbl> <chr>    <chr>      <dbl> <dbl>    <dbl>    <dbl>
## 1 Aphelo~ Scrub~   404   30.5 cup      decid        4.5     1     16       18.5
## 2 Gymnor~ Pinyo~   406   26.5 cup      conif        4.5     1     16.5     21
## 3 Cyanoc~ Blue ~   408   29.5 cup      conif        4.5     1     17       19
## 4 Periso~ Cray ~   410   30.5 cup      decid        3.5     1     17       15
## # ... with 2 more variables: Totcare <dbl>, Closed. <dbl>
```

3. mutate When you want to create new columns based on the values in existing columns, for example, do calculation using existing variables, we'll use the dplyr function mutate().

```
head(mutate(BirdNest, ratio = Incubate/Totcare))
```

```
## # A tibble: 6 x 13
##   Species Common  Page Length Nesttype Location No.eggs Color Incubate Nestling
##   <chr>   <chr>  <dbl>  <dbl> <chr>    <chr>      <dbl> <dbl>    <dbl>    <dbl>
## 1 Tyrann~ Easte~   360   20   cup      decid        3.5     1     17       17
## 2 Myiody~ Sulph~   368   20   cavity   decid        3.5     1     15.5     17
## 3 Myiarc~ Ash-t~   372   20   cavity   decid        4.5     1     15       15
## 4 Myiarc~ Brown~   372   22.5 cavity   decid        4.5     1     14       16.5
## 5 Myarch~ Dusky~   374   17   cavity   decid        4.5     1     14       14
## 6 Sayorn~ Easte~   378   17   cup      bridge       4.5     0     16       15.5
## # ... with 3 more variables: Totcare <dbl>, Closed. <dbl>, ratio <dbl>
```

```
head(mutate(BirdNest, ratio = Incubate/Totcare, inverse = 1/ratio))
```

```
## # A tibble: 6 x 14
##   Species Common  Page Length Nesttype Location No.eggs Color Incubate Nestling
##   <chr>   <chr>  <dbl>  <dbl> <chr>    <chr>      <dbl> <dbl>    <dbl>    <dbl>
## 1 Tyrann~ Easte~   360   20   cup      decid        3.5     1     17       17
## 2 Myiody~ Sulph~   368   20   cavity   decid        3.5     1     15.5     17
## 3 Myiarc~ Ash-t~   372   20   cavity   decid        4.5     1     15       15
## 4 Myiarc~ Brown~   372   22.5 cavity   decid        4.5     1     14       16.5
## 5 Myarch~ Dusky~   374   17   cavity   decid        4.5     1     14       14
## 6 Sayorn~ Easte~   378   17   cup      bridge       4.5     0     16       15.5
## # ... with 4 more variables: Totcare <dbl>, Closed. <dbl>, ratio <dbl>,
## #   inverse <dbl>
```

```
head(mutate(BirdNest, cumsum_total = cumsum(Totcare)))
```

```
## # A tibble: 6 x 13
##   Species Common  Page Length Nesttype Location No.eggs Color Incubate Nestling
##   <chr>   <chr>  <dbl>  <dbl> <chr>    <chr>      <dbl> <dbl>    <dbl>    <dbl>
## 1 Tyrann~ Easte~   360   20   cup      decid        3.5     1     17       17
## 2 Myiody~ Sulph~   368   20   cavity   decid        3.5     1     15.5     17
## 3 Myiarc~ Ash-t~   372   20   cavity   decid        4.5     1     15       15
## 4 Myiarc~ Brown~   372   22.5 cavity   decid        4.5     1     14       16.5
## 5 Myarch~ Dusky~   374   17   cavity   decid        4.5     1     14       14
## 6 Sayorn~ Easte~   378   17   cup      bridge       4.5     0     16       15.5
## # ... with 3 more variables: Totcare <dbl>, Closed. <dbl>, cumsum_total <dbl>
```

```
head(mutate(BirdNest, nor_Nest = Nestling/mean(Nestling, na.rm=T))) # na.rm=T removes the missing value
```

```
## # A tibble: 6 x 13
##   Species Common  Page Length Nesttype Location No.eggs Color Incubate Nestling
##   <chr>   <chr>  <dbl>  <dbl> <chr>    <chr>      <dbl> <dbl>    <dbl>    <dbl>
## 1 Tyrann~ Easte~   360   20   cup      decid        3.5     1     17       17
## 2 Myiody~ Sulph~   368   20   cavity   decid        3.5     1     15.5     17
## 3 Myiarc~ Ash-t~   372   20   cavity   decid        4.5     1     15       15
## 4 Myiarc~ Brown~   372   22.5 cavity   decid        4.5     1     14       16.5
## 5 Myarch~ Dusky~   374   17   cavity   decid        4.5     1     14       14
## 6 Sayorn~ Easte~   378   17   cup      bridge       4.5     0     16       15.5
## # ... with 3 more variables: Totcare <dbl>, Closed. <dbl>, nor_Nest <dbl>
```

```
head(mutate(BirdNest, cup_type = case_when(Nesttype == "cup"~ 1, Nesttype != "cup"~ 0)))
```

```
## # A tibble: 6 x 13
##   Species Common  Page Length Nesttype Location No.eggs Color Incubate Nestling
##   <chr>   <chr>  <dbl>  <dbl> <chr>    <chr>      <dbl> <dbl>    <dbl>    <dbl>
## 1 Tyrann~ Easte~   360   20   cup      decid        3.5     1     17       17
## 2 Myiody~ Sulph~   368   20   cavity   decid        3.5     1     15.5     17
## 3 Myiarc~ Ash-t~   372   20   cavity   decid        4.5     1     15       15
## 4 Myiarc~ Brown~   372   22.5 cavity   decid        4.5     1     14       16.5
## 5 Myarch~ Dusky~   374   17   cavity   decid        4.5     1     14       14
## 6 Sayorn~ Easte~   378   17   cup      bridge       4.5     0     16       15.5
## # ... with 3 more variables: Totcare <dbl>, Closed. <dbl>, cup_type <dbl>
```

```
head(mutate(BirdNest, cup_type = ifelse(Nesttype == "cup", 1,0)))
```

```
## # A tibble: 6 x 13
##   Species Common  Page Length Nesttype Location No.eggs Color Incubate Nestling
##   <chr>   <chr>  <dbl>  <dbl> <chr>    <chr>      <dbl> <dbl>    <dbl>    <dbl>
## 1 Tyrann~ Easte~   360   20   cup      decid        3.5     1     17       17
## 2 Myiody~ Sulph~   368   20   cavity   decid        3.5     1     15.5     17
## 3 Myiarc~ Ash-t~   372   20   cavity   decid        4.5     1     15       15
## 4 Myiarc~ Brown~   372   22.5 cavity   decid        4.5     1     14       16.5
## 5 Myarch~ Dusky~   374   17   cavity   decid        4.5     1     14       14
## 6 Sayorn~ Easte~   378   17   cup      bridge       4.5     0     16       15.5
## # ... with 3 more variables: Totcare <dbl>, Closed. <dbl>, cup_type <dbl>
```

```
a <- mutate(BirdNest, totcare_gt27 = case_when(Totcare >27 ~ 1, Totcare <=27~ 0))
table(a$totcare_gt27) # missing value is NA
```

```
##
##  0  1
## 41 42
```

```
b <- mutate(BirdNest, totcare_gt27 = case_when(Totcare >27 ~ 1, Totcare <=27~ 0, TRUE ~999))
table(b$totcare_gt27) # missing value is defined
```

```
##
##   0   1 999
##  41  42   1
```

```
c <- mutate(BirdNest, totcare_mul = case_when(Totcare <27 ~ 0, Totcare >=30~ 2, Totcare >= 27 & Totcare
table(c$totcare_mul) # missing value is defined
```

```
##
##  0  1  2
## 36 20 27
```

```
d <- mutate(BirdNest, totcare_gt27 = ifelse(Totcare >27 ,1, 0)) # become complex if multiple crtieria
table(d$totcare_gt27)
```

```
## 
##  0  1 
## 41 42
```

Note: create variables with criteria

usage	summary
pmin(), pmax()	Elementwise minimum or maximum
cummin(), cummax()	Cumulative minimum and maximum
cumsum(), cumprod()	Cumulative sum and product
ifelse	Conditioning on less criteria
case_when	Conditioning on more criteria

4. arrange Arrange the rows of your data based according to the preferred order in the specified variable. Default ascending, use "desc" for descending.

```
# Order the data by ascending length
head(arrange(BirdNest, Length))
```

```
## # A tibble: 6 x 12
##   Species Common  Page Length Nesttype Location No.eggs Color Incubate Nestling
##   <chr>   <chr>  <dbl>  <dbl> <chr>    <chr>      <dbl> <dbl>    <dbl>    <dbl>
## 1 Regulu~ Golde~   448    9   pendant  conif        8.5     1     14.5     16.5
## 2 Sitta ~ Pygmy~   436   10   cavity   conif        7       1     15.5     21  
## 3 Regulu~ Ruby-~   450   10   pendant  conif        8       1     12       12  
## 4 Auripa~ Verdin   432   10.5 spheric~ shrub        4.5     1     10       21  
## 5 Sitta ~ Red-b~   436   11   cavity   conif        5.5     1     12       17.5
## 6 Poliop~ Black~   452   11   cup      shrub        4       1     14       12  
## # ... with 2 more variables: Totcare <dbl>, Closed. <dbl>
```

```
head(arrange(BirdNest, Length, No.eggs))
```

```
## # A tibble: 6 x 12
##   Species Common  Page Length Nesttype Location No.eggs Color Incubate Nestling
##   <chr>   <chr>  <dbl>  <dbl> <chr>    <chr>      <dbl> <dbl>    <dbl>    <dbl>
## 1 Regulu~ Golde~   448    9   pendant  conif        8.5     1     14.5     16.5
## 2 Sitta ~ Pygmy~   436   10   cavity   conif        7       1     15.5     21  
## 3 Regulu~ Ruby-~   450   10   pendant  conif        8       1     12       12  
## 4 Auripa~ Verdin   432   10.5 spheric~ shrub        4.5     1     10       21  
## 5 Poliop~ Black~   452   11   cup      shrub        4       1     14       12  
## 6 Sitta ~ Red-b~   436   11   cavity   conif        5.5     1     12       17.5
## # ... with 2 more variables: Totcare <dbl>, Closed. <dbl>
```

```
# Order Common by descending
head(arrange(BirdNest, desc(Common)))
```

```
## # A tibble: 6 x 12
##   Species Common  Page Length Nesttype Location No.eggs Color Incubate Nestling
##   <chr>   <chr>  <dbl>  <dbl> <chr>    <chr>      <dbl> <dbl>    <dbl>    <dbl>
## 1 Icteri~ Yello~   548   18   cup      shrub        3.5     1     11        8
## 2 Motaci~ Yello~   482   16   cup      ground       5.5     1     11.5     15.5
## 3 Helmit~ Worm-~   540   13.5 cup      ground       4.5     1     13       10
## 4 Hyloci~ Wood ~   456   20   cup      decid        3.5     0     13.5     12
## 5 Sitta ~ White~   434   14   cavity   decid        6.5     1     12       14
## 6 Motaci~ White~   480   18   crevice  bank         5.5     1     13       14.5
## # ... with 2 more variables: Totcare <dbl>, Closed. <dbl>
```

5. pipes Pipes can be used when you want to many things to the same data set. It takes the output of one function and send it directly to the next. Different layes can be added in the pipes. You will need to consider the order of adding the layers, because it needs to execute the analysis you plan and satisfy pipe logic.

Note: ctrl+shift+m for the %>% symbol for Mac.

```
head(select(BirdNest, Length, Nesttype, Location, No.eggs))
```

```
## # A tibble: 6 x 4
##   Length Nesttype Location No.eggs
##    <dbl> <chr>    <chr>      <dbl>
## 1   20   cup      decid        3.5
## 2   20   cavity   decid        3.5
## 3   20   cavity   decid        4.5
## 4   22.5 cavity   decid        4.5
## 5   17   cavity   decid        4.5
## 6   17   cup      bridge       4.5
```

```
# equals to
head(BirdNest %>% select(Length, Nesttype, Location, No.eggs))
```

```
## # A tibble: 6 x 4
##   Length Nesttype Location No.eggs
##    <dbl> <chr>    <chr>      <dbl>
## 1   20   cup      decid        3.5
## 2   20   cavity   decid        3.5
## 3   20   cavity   decid        4.5
## 4   22.5 cavity   decid        4.5
## 5   17   cavity   decid        4.5
## 6   17   cup      bridge       4.5
```

```
head(filter(BirdNest, No.eggs>6 , Location %in% c("decid")))
```

```
## # A tibble: 3 x 12
##   Species Common  Page Length Nesttype Location No.eggs Color Incubate Nestling
##   <chr>   <chr>  <dbl>  <dbl> <chr>    <chr>      <dbl> <dbl>    <dbl>    <dbl>
## 1 Parus ~ Black~   424     13 cavity   decid        7       1       12       16
## 2 Sitta ~ White~   434     14 cavity   decid        6.5     1       12       14
## 3 Troglo~ House~   438     12 cavity   decid        7       1       13       15
## # ... with 2 more variables: Totcare <dbl>, Closed. <dbl>
```

```
# equals to
head(BirdNest %>% filter( No.eggs>6 , Location %in% c("decid")))
```

```
## # A tibble: 3 x 12
##   Species Common  Page Length Nesttype Location No.eggs Color Incubate Nestling
##   <chr>   <chr>  <dbl>  <dbl> <chr>    <chr>      <dbl> <dbl>    <dbl>    <dbl>
## 1 Parus ~ Black~   424     13 cavity   decid        7       1       12       16
## 2 Sitta ~ White~   434     14 cavity   decid        6.5     1       12       14
## 3 Troglo~ House~   438     12 cavity   decid        7       1       13       15
## # ... with 2 more variables: Totcare <dbl>, Closed. <dbl>
```

```
# add layer
BirdNest %>%
  filter( No.eggs>6 , Location %in% c("decid")) %>%
  select(Species, Common, No.eggs, Location)
```

```
## # A tibble: 3 x 4
##   Species            Common                  No.eggs Location
##   <chr>              <chr>                     <dbl> <chr>
## 1 Parus atricapillus Black-capped Chickadee      7   decid
## 2 Sitta carolinensis White-breasted Nuthatch     6.5 decid
## 3 Troglodytes aedon  House Wren                  7   decid
```

Question: Could you output top 10 observations with largest length?

```
BirdNest %>%
  arrange(desc(Length) ) %>%
  head(n = 10)
```

```
## # A tibble: 10 x 12
##    Species Common  Page Length Nesttype Location No.eggs Color Incubate Nestling
##    <chr>   <chr>  <dbl>  <dbl> <chr>    <chr>      <dbl> <dbl>    <dbl>    <dbl>
##  1 Nucifr~ Clark~   410   31.5 cup      conif        3       1     17       19.5
##  2 Aphelo~ Scrub~   404   30.5 cup      decid        4.5     1     16       18.5
##  3 Periso~ Cray ~   410   30.5 cup      decid        3.5     1     17       15
##  4 Toxost~ Calif~   476   30.5 cup      shrub        3.5     1     14       13
##  5 Cyanoc~ Blue ~   408   29.5 cup      conif        4.5     1     17       19
##  6 Toxost~ Brown~   470   29   cup      shrub        4.5     1     12.5     11
##  7 Gymnor~ Pinyo~   406   26.5 cup      conif        4.5     1     16.5     21
##  8 Toxost~ Curve~   472   26.5 cup      shrub        3.5     1     13.5     14.5
##  9 Turdus~ Ameri~   462   25.5 cup      decid        4       0     13       15
## 10 Mimus ~ North~   468   25.5 cup      shrub        1       1     12.5     12
## # ... with 2 more variables: Totcare <dbl>, Closed. <dbl>
```

6. summarise When you want to create a summary across the data, summarise() function in dplyr package can be used. Generally, it often combines with group_by, which creates a summary by subgroups.

```
# to create a summery about average, variance of length, and count distinct egg color
BirdNest %>% summarise(mean_length = mean(Length), var_Length = var(Length), n_distict_color = n_distin
## # A tibble: 1 x 3
##   mean_length var_Length n_distict_color
##         <dbl>      <dbl>           <int>
## 1        17.6       27.6               2
```

```
# to create a summery respective to same fields above within each egg color.
BirdNest %>%
  group_by(Color) %>%
  summarise(mean_length = mean(Length), var_Length = var(Length), n_distict_color = n_distinct(Color))
```

```
## # A tibble: 2 x 4
##   Color mean_length var_Length n_distict_color
##   <dbl>       <dbl>      <dbl>           <int>
## 1     0        17.6       13.6               1
## 2     1        17.6       30.6               1
```

7. group by group_by() and summarise together can create a split-apply-combine analysis. group_by() splits the data into groups, summarise() provides summary function in each group and the summary for each subgroups are combined and returned.

Note:
Adding multiple variables in group_by() will return a summary with grouping by adding order.

```
# create summary about mean and variance of legnth, number of distinct location by color and nesttype
BirdNest %>%
  group_by(Color,Nesttype) %>%
  summarise(mean_length = mean(Length), num_obs = n(), n_distict_location = n_distinct(Location))
```

```
## ‘summarise()‘ has grouped output by ’Color’. You can override using the ‘.groups‘ argument.
```

```
## # A tibble: 10 x 5
## # Groups:   Color [2]
##    Color Nesttype  mean_length num_obs n_distict_location
##    <dbl> <chr>           <dbl>   <int>              <int>
##  1     0 burrow           13         2                  1
##  2     0 cavity           18         2                  1
##  3     0 crevice          19.5       1                  1
##  4     0 cup              18.3       9                  4
##  5     1 cavity           15.3      15                  4
##  6     1 crevice          17.5       2                  2
##  7     1 cup              19.0      44                  5
##  8     1 pendant           9.5       2                  1
##  9     1 saucer           16.9       4                  3
## 10     1 spherical        15.5       3                  2
```

Take home question:

Could you create a summary about the largest ratio (ratio = Nestling / Totcare) by nest type (excluding “cavity” category) and present the result in a descending order?

```
## # A tibble: 6 x 2
##   Nesttype  max_ratio
##   <chr>         <dbl>
## 1 spherical     0.677
## 2 burrow        0.625
## 3 crevice       0.589
## 4 cup           0.582
## 5 pendant       0.532
## 6 saucer        0.527
```