Assignment 2 (Session 7, 8 & 9)

Kaushal Khatiwada

2024-03-08

Reading JSON file from URL

JSON (JavaScript Object Notation) is a text-based format for storing and transporting data.

```
library(jsonlite)
Raw<- fromJSON("https://data.ny.gov/api/views/9a8c-vfzj/rows.json?accessType=DOWNLOAD")
food_market<-Raw[['data']]    #Fetch data
Names<-food_market[,14]     #Get 14th Column only
head(Names)
```

```
## [1] "CA FOOD MART" "1307 CORP"    "GNP SUNIL"    "BRONX BAZAR"  "TOPS 754"
## [6] "MAY STORE"
```

Representation in Table format with frequency

```
head(table(food_market[,19]))
```

```
##
##       ACCORD        ADAMS ADAMS CENTER      ADDISON   ADIRONDACK        AFTON
##            5            9            2            8            1            4
```

HTML Scrapping with R

We can get the information from the internet using web scrapping but might be illegal if done without authorization

```
library(rvest)
simple<-read_html("https://dataquestio.github.io/web-scraping-pages/simple.html")
simple %>%
  html_nodes("p") %>%   #Get <p> HTML tag
  html_text()           #Extract text content
```

```
## [1] "Here is some simple content for this page."
```

Scrape Covid Table from wikipedia

Scrapping using html elements.

```
library(rvest)
wiki_link<-"https://en.wikipedia.org/wiki/COVID-19_pandemic_in_Nepal"
wiki_page<-read_html(wiki_link)
covid_table<-wiki_page %>%
  html_elements(".COVID-19_pandemic_data_Nepal_medical_cases") %>%    #Div class
  html_nodes("table") %>%                                         #Get table tag
  html_table() %>%.[[1]]                                          #Get table content
tail(covid_table)
```

```
## # A tibble: 6 x 14
##   Date   ‘Confirmed cases‘ ‘Confirmed cases‘ ‘Confirmed cases‘ Recoveries
##   <chr>  <chr>             <chr>             <chr>             <chr>
## 1 26 May 535,525           +6,677            117,077           411,603
## 2 27 May 542,256           +6,731            116,476           418,829
## 3 28 May 549,111           +6,855            113,394           428,670
## 4 29 May 553,422           +4,311            111,509           434,750
## 5 30 May 557,124           +3,702            108,897           440,955
## 6 31 May 561,302           +4,178            106,470           447,446
## # i 9 more variables: Recoveries <chr>, Deaths <chr>, Deaths <chr>,
## #   ‘RT-PCR tests‘ <chr>, ‘RT-PCR tests‘ <chr>, TPR <chr>, RR <chr>, CFR <chr>,
## #   Ref. <chr>
```

Data Wrangling

Previously, Covid Table contains sub column [Total,New,Active]. so, we need to remove the sub column heading after concatenating the column with sub column using “_” separator.

```
names(covid_table)<- paste(names(covid_table),covid_table[1,], sep="_")
covid_table<-covid_table[-1,]
tail(covid_table)
```

```
## # A tibble: 6 x 14
##   Date_Date ‘Confirmed cases_Total‘ ‘Confirmed cases_New‘ Confirmed cases_Acti~1
##   <chr>     <chr>                   <chr>                 <chr>
## 1 26 May    535,525                 +6,677                117,077
## 2 27 May    542,256                 +6,731                116,476
## 3 28 May    549,111                 +6,855                113,394
## 4 29 May    553,422                 +4,311                111,509
## 5 30 May    557,124                 +3,702                108,897
## 6 31 May    561,302                 +4,178                106,470
## # i abbreviated name: 1: ‘Confirmed cases_Active‘
## # i 10 more variables: Recoveries_Total <chr>, Recoveries_New <chr>,
## #   Deaths_Total <chr>, Deaths_New <chr>, ‘RT-PCR tests_Total‘ <chr>,
## #   ‘RT-PCR tests_New‘ <chr>, TPR_TPR <chr>, RR_RR <chr>, CFR_CFR <chr>,
## #   Ref._Ref. <chr>
```

Renaming Column names

```
colnames(covid_table) <- c("Date", "Confirmed_Cases_Total",
                           "Confirmed_Cases_New", "Confirmed_Cases_Active",
                           "Recoveries_Total", "Recoveries_New", "Deaths_Total",
                           "Deaths_New", "PCR_Total", "PCR_New", "TPR", "RR", "CFR", "Ref")
tail(covid_table)
```

```
## # A tibble: 6 x 14
##   Date   Confirmed_Cases_Total Confirmed_Cases_New Confirmed_Cases_Active
##   <chr>  <chr>                 <chr>               <chr>
## 1 26 May 535,525               +6,677              117,077
## 2 27 May 542,256               +6,731              116,476
## 3 28 May 549,111               +6,855              113,394
## 4 29 May 553,422               +4,311              111,509
## 5 30 May 557,124               +3,702              108,897
## 6 31 May 561,302               +4,178              106,470
## # i 10 more variables: Recoveries_Total <chr>, Recoveries_New <chr>,
## #   Deaths_Total <chr>, Deaths_New <chr>, PCR_Total <chr>, PCR_New <chr>,
## #   TPR <chr>, RR <chr>, CFR <chr>, Ref <chr>
```

Remove "+" , "%" and ","

Cleaning the data using gsub()function to replace the pattern. gsub(PATTERN,REPLACEMENT,COLUMN)

```
covid_table$Confirmed_Cases_New <- gsub('[+]', '', covid_table$Confirmed_Cases_New)
covid_table$Recoveries_New <- gsub('[+]', '', covid_table$Recoveries_New)
covid_table$Deaths_New <- gsub('[+]', '', covid_table$Deaths_New)
covid_table$PCR_New <- gsub('[+]', '', covid_table$PCR_New)

covid_table$TPR <- gsub('[%]', '', covid_table$TPR)
covid_table$RR <- gsub('[%]', '', covid_table$RR)
covid_table$CFR <- gsub('[%]', '', covid_table$CFR)

covid_table$Confirmed_Cases_Total <- gsub('[,]', '', covid_table$Confirmed_Cases_Total)   #Remove "," f
covid_table$Confirmed_Cases_New <- gsub('[,]', '', covid_table$Confirmed_Cases_New)
covid_table$Confirmed_Cases_Active <- gsub('[,]', '', covid_table$Confirmed_Cases_Active)
covid_table$Recoveries_Total <- gsub('[,]', '', covid_table$Recoveries_Total)
covid_table$Recoveries_New <- gsub('[,]', '', covid_table$Recoveries_New)
covid_table$Deaths_Total <- gsub('[,]', '', covid_table$Deaths_Total)
covid_table$Deaths_New <- gsub('[,]', '', covid_table$Deaths_New)
covid_table$PCR_Total <- gsub('[,]', '', covid_table$PCR_Total)
covid_table$PCR_New <- gsub('[,]', '', covid_table$PCR_New)

tail(covid_table)
```

```
## # A tibble: 6 x 14
##   Date   Confirmed_Cases_Total Confirmed_Cases_New Confirmed_Cases_Active
##   <chr>  <chr>                 <chr>               <chr>
## 1 26 May 535525                6677                117077
## 2 27 May 542256                6731                116476
## 3 28 May 549111                6855                113394
## 4 29 May 553422                4311                111509
## 5 30 May 557124                3702                108897
## 6 31 May 561302                4178                106470
## # i 10 more variables: Recoveries_Total <chr>, Recoveries_New <chr>,
## #   Deaths_Total <chr>, Deaths_New <chr>, PCR_Total <chr>, PCR_New <chr>,
## #   TPR <chr>, RR <chr>, CFR <chr>, Ref <chr>
```

Convert data type from characters to integers and numeric

```r
covid_table$Confirmed_Cases_Total <- suppressWarnings(as.integer(covid_table$Confirmed_Cases_Total))
covid_table$Confirmed_Cases_New <- suppressWarnings(as.integer(covid_table$Confirmed_Cases_New))
covid_table$Confirmed_Cases_Active <- suppressWarnings(as.integer(covid_table$Confirmed_Cases_Active))
covid_table$Recoveries_Total <- suppressWarnings(as.integer(covid_table$Recoveries_Total))
covid_table$Recoveries_New <- suppressWarnings(as.integer(covid_table$Recoveries_New))
covid_table$Deaths_Total <- suppressWarnings(as.integer(covid_table$Deaths_Total))
covid_table$Deaths_New <- suppressWarnings(as.integer(covid_table$Deaths_New))
covid_table$PCR_Total <- suppressWarnings(as.integer(covid_table$PCR_Total))
covid_table$PCR_New <- suppressWarnings(as.integer(covid_table$PCR_New))

covid_table$TPR <- as.numeric(covid_table$TPR)
covid_table$RR <- as.numeric(covid_table$RR)
covid_table$CFR <- as.numeric(covid_table$CFR)
str(covid_table)
```

```
## tibble [495 x 14] (S3: tbl_df/tbl/data.frame)
##  $ Date                  : chr [1:495] "23 Jan" "24 Jan" "25 Jan" "26 Jan" ...
##  $ Confirmed_Cases_Total : int [1:495] 1 1 1 1 1 1 1 1 1 1 ...
##  $ Confirmed_Cases_New   : int [1:495] 1 0 0 0 0 0 0 0 0 0 ...
##  $ Confirmed_Cases_Active: int [1:495] 1 1 1 1 1 1 0 0 0 0 ...
##  $ Recoveries_Total      : int [1:495] 0 0 0 0 0 0 1 1 1 1 ...
##  $ Recoveries_New        : int [1:495] 0 0 0 0 0 0 1 0 0 0 ...
##  $ Deaths_Total          : int [1:495] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Deaths_New            : int [1:495] 0 0 0 0 0 0 0 0 0 0 ...
##  $ PCR_Total             : int [1:495] NA NA NA NA NA 3 4 5 5 NA ...
##  $ PCR_New               : int [1:495] NA NA NA NA NA NA 1 1 0 NA ...
##  $ TPR                   : num [1:495] NA NA NA NA NA ...
##  $ RR                    : num [1:495] 0 0 0 0 0 0 100 100 100 100 ...
##  $ CFR                   : num [1:495] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Ref                   : chr [1:495] "[175]" "" "" "" ...
```

Convert Date Format

Date was in the format of %d %b. So it is converted to %m %d.

```r
date_format <- as.Date(covid_table$Date,format = "%d %b")
final_format <- format(date_format,"%m-%d")
covid_table$Date <- final_format
tail(covid_table)
```

```
## # A tibble: 6 x 14
##   Date  Confirmed_Cases_Total Confirmed_Cases_New Confirmed_Cases_Active
##   <chr>                 <int>               <int>                  <int>
## 1 05-26                535525                6677                 117077
## 2 05-27                542256                6731                 116476
## 3 05-28                549111                6855                 113394
## 4 05-29                553422                4311                 111509
## 5 05-30                557124                3702                 108897
## 6 05-31                561302                4178                 106470
## # i 10 more variables: Recoveries_Total <int>, Recoveries_New <int>,
## #   Deaths_Total <int>, Deaths_New <int>, PCR_Total <int>, PCR_New <int>,
## #   TPR <dbl>, RR <dbl>, CFR <dbl>, Ref <chr>
```

Tibble

A tibble is a type of data frame.

Create Tibble

```
library(tidyr)
table1 <- tibble(
  country = c("Afghanistan","Afghanistan","Brazil","Brazil","China","China"),
  year = c(1999,2000,1999,2000,1999,2000),
  cases = c(745,2666,37737,80488,212258,213766),
  population = c(19987071,20595360,172006362, 174504898, 1272915272,1280428583)
)
table1
```

```
## # A tibble: 6 x 4
##   country      year  cases population
##   <chr>       <dbl>  <dbl>      <dbl>
## 1 Afghanistan  1999    745   19987071
## 2 Afghanistan  2000   2666   20595360
## 3 Brazil       1999  37737  172006362
## 4 Brazil       2000  80488  174504898
## 5 China        1999 212258 1272915272
## 6 China        2000 213766 1280428583
```

Pivoting [Longer to Wider]

Best for standard statistical analysis

```
table2
```

```
## # A tibble: 12 x 4
##    country      year type            count
##    <chr>       <dbl> <chr>           <dbl>
##  1 Afghanistan  1999 cases             745
##  2 Afghanistan  1999 population   19987071
##  3 Afghanistan  2000 cases            2666
##  4 Afghanistan  2000 population   20595360
##  5 Brazil       1999 cases           37737
##  6 Brazil       1999 population  172006362
##  7 Brazil       2000 cases           80488
##  8 Brazil       2000 population  174504898
##  9 China        1999 cases          212258
## 10 China        1999 population 1272915272
## 11 China        2000 cases          213766
## 12 China        2000 population 1280428583
```

```
table2 %>%pivot_wider(names_from = type, values_from = count)
```

```
## # A tibble: 6 x 4
##   country      year  cases population
##   <chr>       <dbl>  <dbl>      <dbl>
## 1 Afghanistan  1999    745   19987071
## 2 Afghanistan  2000   2666   20595360
## 3 Brazil       1999  37737  172006362
## 4 Brazil       2000  80488  174504898
## 5 China        1999 212258 1272915272
## 6 China        2000 213766 1280428583
```

Pivoting [Wider to Longer]

Best for variance components analysis

```
table4a
```

```
## # A tibble: 3 x 3
##   country     ‘1999‘ ‘2000‘
##   <chr>        <dbl>  <dbl>
## 1 Afghanistan    745   2666
## 2 Brazil       37737  80488
## 3 China       212258 213766
```

```
table4a %>% pivot_longer(c('1999','2000'),names_to = "year",values_to="cases")
```

```
## # A tibble: 6 x 3
##   country     year   cases
##   <chr>       <chr>  <dbl>
## 1 Afghanistan 1999     745
## 2 Afghanistan 2000    2666
## 3 Brazil      1999   37737
## 4 Brazil      2000   80488
## 5 China       1999  212258
## 6 China       2000  213766
```

Separate

Separate the single column into multiple column

```
table3
```

```
## # A tibble: 6 x 3
##   country      year rate
##   <chr>       <dbl> <chr>
## 1 Afghanistan  1999 745/19987071
## 2 Afghanistan  2000 2666/20595360
## 3 Brazil       1999 37737/172006362
## 4 Brazil       2000 80488/174504898
## 5 China        1999 212258/1272915272
## 6 China        2000 213766/1280428583
```

```
table3 %>% separate(rate,into = c("cases","population"))
```

```
## # A tibble: 6 x 4
##   country      year cases  population
##   <chr>       <dbl> <chr>  <chr>
## 1 Afghanistan  1999 745    19987071
## 2 Afghanistan  2000 2666   20595360
## 3 Brazil       1999 37737  172006362
## 4 Brazil       2000 80488  174504898
## 5 China        1999 212258 1272915272
## 6 China        2000 213766 1280428583
```

Unite

Combine multiple columns into single column

```
table5
```

```
## # A tibble: 6 x 4
##   country     century year  rate
##   <chr>       <chr>   <chr> <chr>
## 1 Afghanistan 19      99    745/19987071
## 2 Afghanistan 20      00    2666/20595360
## 3 Brazil      19      99    37737/172006362
## 4 Brazil      20      00    80488/174504898
## 5 China       19      99    212258/1272915272
## 6 China       20      00    213766/1280428583
```

```
table5 %>% unite(new,century,year)
```

```
## # A tibble: 6 x 3
##   country     new   rate
##   <chr>       <chr> <chr>
## 1 Afghanistan 19_99 745/19987071
## 2 Afghanistan 20_00 2666/20595360
## 3 Brazil      19_99 37737/172006362
## 4 Brazil      20_00 80488/174504898
## 5 China       19_99 212258/1272915272
## 6 China       20_00 213766/1280428583
```

Missing values in tibble

Explicitly missing: 4th Quarter of 2015 Implicitly missing: 1st Quarter of 2016

```
stocks <- tibble(
  year = c(2015, 2015, 2015, 2015, 2016, 2016, 2016),
  qtr = c( 1, 2, 3, 4, 2, 3, 4),
  return = c(1.88, 0.59, 0.35, NA, 0.92, 0.17, 2.66)
)
stocks
```

```
## # A tibble: 7 x 3
##    year   qtr return
##   <dbl> <dbl>  <dbl>
## 1  2015     1   1.88
## 2  2015     2   0.59
## 3  2015     3   0.35
## 4  2015     4  NA
## 5  2016     2   0.92
## 6  2016     3   0.17
## 7  2016     4   2.66
```

Pivot and changing the data set to handle the missing values

```
stocks
```

```
## # A tibble: 7 x 3
##    year   qtr return
##   <dbl> <dbl>  <dbl>
## 1  2015     1   1.88
## 2  2015     2   0.59
## 3  2015     3   0.35
## 4  2015     4  NA
## 5  2016     2   0.92
## 6  2016     3   0.17
## 7  2016     4   2.66
```

```
stocks %>% pivot_wider(names_from = year,values_from = return)
```

```
## # A tibble: 4 x 3
##     qtr `2015` `2016`
##   <dbl>  <dbl>  <dbl>
## 1     1   1.88  NA
## 2     2   0.59   0.92
## 3     3   0.35   0.17
## 4     4  NA      2.66
```

Handle the missing value and drop the NA value.

```
stocks %>%
  pivot_wider(names_from = year, values_from = return) %>%
  pivot_longer(
    cols = c(`2015`, `2016`),
    names_to = "year",
    values_to = "return",
    values_drop_na = TRUE
  )
```

```
## # A tibble: 6 x 3
##     qtr year  return
##   <dbl> <chr>  <dbl>
## 1     1 2015    1.88
## 2     2 2015    0.59
## 3     2 2016    0.92
## 4     3 2015    0.35
## 5     3 2016    0.17
## 6     4 2016    2.66
```

Using “complete” function to handle missing value

```
stocks %>% complete(year,qtr)
```

```
## # A tibble: 8 x 3
##    year   qtr return
##   <dbl> <dbl>  <dbl>
## 1  2015     1   1.88
## 2  2015     2   0.59
## 3  2015     3   0.35
## 4  2015     4  NA
## 5  2016     1  NA
## 6  2016     2   0.92
## 7  2016     3   0.17
## 8  2016     4   2.66
```

Tibble by row

```
treatment <- tribble(
  ~ person, ~ treatment, ~response,
  "Derrick Whitmore", 1, 7,
  NA, 2, 10,
  NA, 3, 9,
  "Katherine Burke", 1, 4
)
treatment
```

```
## # A tibble: 4 x 3
##   person           treatment response
##   <chr>                <dbl>    <dbl>
## 1 Derrick Whitmore         1        7
## 2 <NA>                     2       10
## 3 <NA>                     3        9
## 4 Katherine Burke          1        4
```

Use fill function to handle missing values

```
treatment %>% fill(person)
```

```
## # A tibble: 4 x 3
##   person           treatment response
##   <chr>                <dbl>    <dbl>
## 1 Derrick Whitmore         1        7
## 2 Derrick Whitmore         2       10
## 3 Derrick Whitmore         3        9
## 4 Katherine Burke          1        4
```

Data Manipulation

filter() function is used to pick the observation by their values. arrange() function is used to reorder the rows. select() function is used to pick the variables by their names. mutate() function is used to create new variable with respect to existing variables summarise() function is used to merge down the values to single summary. Use the NYC flight on year 2013 data set.

Filter

Filter flights for 25th December 2013

```
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
```

```
library(nycflights13)                 #nycflights13 data
filter(flights, month == 12,day == 25)  #filter fights on Christmas day
```

```
## # A tibble: 719 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    25      456            500        -4      649            651
##  2  2013    12    25      524            515         9      805            814
##  3  2013    12    25      542            540         2      832            850
##  4  2013    12    25      546            550        -4     1022           1027
##  5  2013    12    25      556            600        -4      730            745
##  6  2013    12    25      557            600        -3      743            752
##  7  2013    12    25      557            600        -3      818            831
##  8  2013    12    25      559            600        -1      855            856
##  9  2013    12    25      559            600        -1      849            855
## 10  2013    12    25      600            600         0      850            846
## # i 709 more rows
## # i 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>
```

Using Operator to filter the flight on Either November or December

```
filter(flights,month==11 | month==12)
```

```
## # A tibble: 55,403 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
##  4  2013    11     1      539            545        -6      856            827
##  5  2013    11     1      542            545        -3      831            855
##  6  2013    11     1      549            600       -11      912            923
##  7  2013    11     1      550            600       -10      705            659
##  8  2013    11     1      554            600        -6      659            701
##  9  2013    11     1      554            600        -6      826            827
## 10  2013    11     1      554            600        -6      749            751
## # i 55,393 more rows
## # i 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>
```

```
#          OR
#filter(flights,month==11 | 12)
#filter(flights,month %in% c(11,12))
```

De Morgan's Law

To filter the arrival delay and departure delay less than 120 minutes.

```
filter(flights,!(arr_delay>120 | dep_delay>120))
```

```
## # A tibble: 316,050 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
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # i 316,040 more rows
## # i 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>
```

```
#         OR
#filter(flights,arr_delay<=120 , dep_delay<=120)
```

Arrange

To reorder the data set in year,month,day in descending order

```
arrange(flights,year,month,day)
```

```
## # A tibble: 336,776 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
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # i 336,766 more rows
## # i 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>
```

Most delayed Airlines arrange departure delay in descending order Get Top 20 Represent the carrier and its dep_delay frequency in a table

```
delay<-arrange(flights,desc(dep_delay))
dep_delay_20<- head(delay,n=20)
most_delayed_airlines_table<-table(dep_delay_20$carrier)
most_delayed_airlines_table
```

```
## 
## AA DL F9 HA MQ 
##  5  8  1  1  5
```

Average departed delayed time

```
average_departed_delay<-mean(dep_delay_20$dep_delay)
average_departed_delay
```

```
## [1] 925.9
```

Select

```
select(flights,year,month:day)    #Column between month and day
```

```
## # A tibble: 336,776 x 3
##     year month   day
##    <int> <int> <int>
##  1  2013     1     1
##  2  2013     1     1
##  3  2013     1     1
##  4  2013     1     1
##  5  2013     1     1
##  6  2013     1     1
##  7  2013     1     1
##  8  2013     1     1
##  9  2013     1     1
## 10  2013     1     1
## # i 336,766 more rows
```

```
select(flights,-(year:day))     #Exclude column between year and day
```

```
## # A tibble: 336,776 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     
##  4      544            545        -1     1004           1022       -18 B6     
##  5      554            600        -6      812            837       -25 DL     
##  6      554            558        -4      740            728        12 UA     
##  7      555            600        -5      913            854        19 B6     
##  8      557            600        -3      709            723       -14 EV     
##  9      557            600        -3      838            846        -8 B6     
## 10      558            600        -2      753            745         8 AA     
## # i 336,766 more rows
## # i 9 more variables: flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

```
select(flights,time_hour,air_time)
```

```
## # A tibble: 336,776 x 2
##    time_hour           air_time
##    <dttm>                 <dbl>
##  1 2013-01-01 05:00:00      227
##  2 2013-01-01 05:00:00      227
##  3 2013-01-01 05:00:00      160
##  4 2013-01-01 05:00:00      183
##  5 2013-01-01 06:00:00      116
##  6 2013-01-01 05:00:00      150
##  7 2013-01-01 06:00:00      158
##  8 2013-01-01 06:00:00       53
##  9 2013-01-01 06:00:00      140
## 10 2013-01-01 06:00:00      138
## # i 336,766 more rows
```

Mutate

Adds new column to existing column Mutate gain,speed,hours and gain_per_hour column in existing.

```
flights_sml<-select(flights,
                    year:day,
                    ends_with("delay"),
                    distance,
                    air_time
)
flights_sml
```

```
## # A tibble: 336,776 x 7
##     year month   day dep_delay arr_delay distance air_time
##    <int> <int> <int>     <dbl>     <dbl>    <dbl>    <dbl>
##  1  2013     1     1         2        11     1400      227
##  2  2013     1     1         4        20     1416      227
##  3  2013     1     1         2        33     1089      160
##  4  2013     1     1        -1       -18     1576      183
##  5  2013     1     1        -6       -25      762      116
##  6  2013     1     1        -4        12      719      150
##  7  2013     1     1        -5        19     1065      158
##  8  2013     1     1        -3       -14      229       53
##  9  2013     1     1        -3        -8      944      140
## 10  2013     1     1        -2         8      733      138
## # i 336,766 more rows
```

```
mutate(flights_sml,
       gain=dep_delay - arr_delay,
       speed= distance / air_time * 60,
       hours = air_time /60,
       gain_per_hour = gain /hours
)
```

```
## # A tibble: 336,776 x 11
##     year month   day dep_delay arr_delay distance air_time  gain speed hours
##    <int> <int> <int>     <dbl>     <dbl>    <dbl>    <dbl> <dbl> <dbl> <dbl>
##  1  2013     1     1         2        11     1400      227    -9  370. 3.78
##  2  2013     1     1         4        20     1416      227   -16  374. 3.78
##  3  2013     1     1         2        33     1089      160   -31  408. 2.67
##  4  2013     1     1        -1       -18     1576      183    17  517. 3.05
##  5  2013     1     1        -6       -25      762      116    19  394. 1.93
##  6  2013     1     1        -4        12      719      150   -16  288. 2.5
##  7  2013     1     1        -5        19     1065      158   -24  404. 2.63
##  8  2013     1     1        -3       -14      229       53    11  259. 0.883
##  9  2013     1     1        -3        -8      944      140     5  405. 2.33
## 10  2013     1     1        -2         8      733      138   -10  319. 2.3
## # i 336,766 more rows
## # i 1 more variable: gain_per_hour <dbl>
```

Transmute

It is used only to keep the new variables

```
transmute(flights,
          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
## # i 336,766 more rows
```

```
transmute(flights,
          dep_time,
          hour =dep_time %/% 100,       #Modular
          minute =dep_time %% 100,      #Remainder
)
```

```
## # A tibble: 336,776 x 3
##    dep_time  hour minute
##       <int> <dbl>  <dbl>
##  1      517     5     17
##  2      533     5     33
##  3      542     5     42
##  4      544     5     44
##  5      554     5     54
##  6      554     5     54
##  7      555     5     55
##  8      557     5     57
##  9      557     5     57
## 10      558     5     58
## # i 336,766 more rows
```

Summarise

To group the variable by value

```
summarise(flights,delay=mean(dep_delay,na.rm=TRUE))
```

```
## # A tibble: 1 x 1
##   delay
##   <dbl>
## 1  12.6
```

```
by_day<-group_by(flights,year,month,day)
summarise(by_day,delay=mean(dep_delay,na.rm=TRUE), .groups = 'drop')
```

```
## # A tibble: 365 x 4
##     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
## # i 355 more rows
```

MULTIPLE OPERATION WITH PIPES

Displays the mean distance of flights and mean arrival delay of flights to each destination with more than 20 flights excluding destination “HNL”

```
delays<-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")
delays
```

```
## # 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
## # i 86 more rows
```

Remove Cancelled flights First, Filters only those rows where departure delay and arrival delay is not missing which means the flight has take-off and landed.

```
not_cancelled<- flights %>%
  filter(!is.na(dep_delay),!is.na(arr_delay))
not_cancelled
```

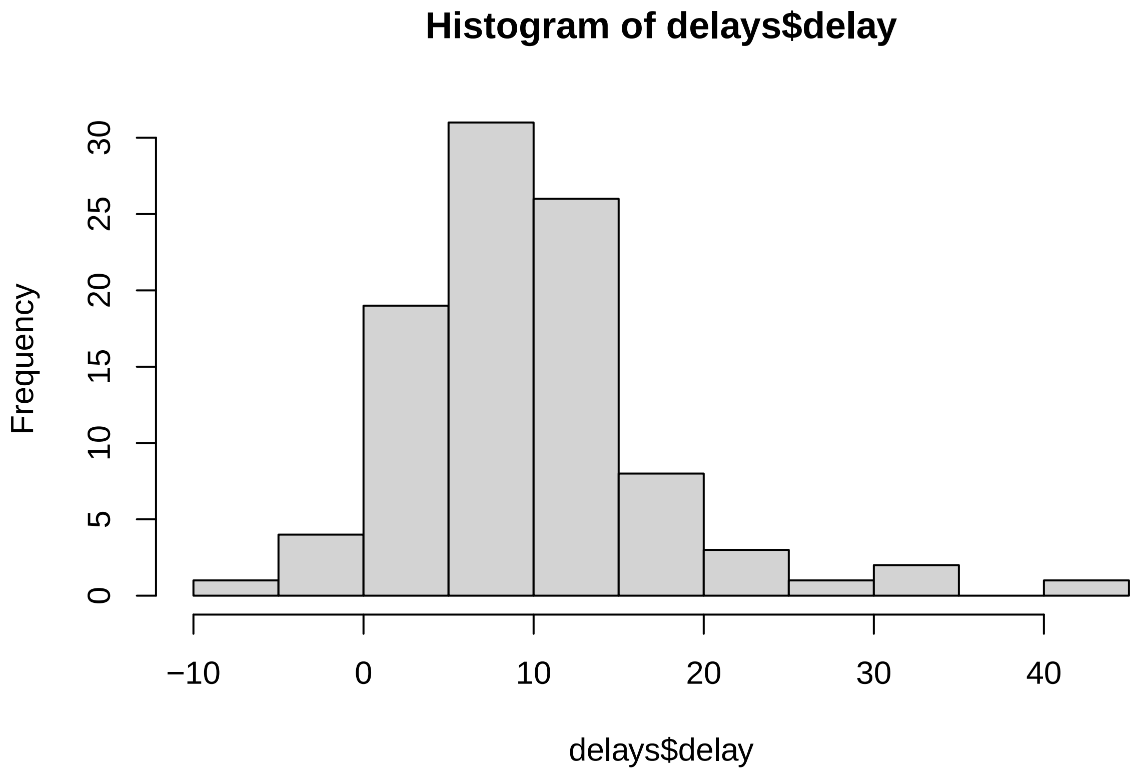
```
## # A tibble: 327,346 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      542            540         2      923            850
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # i 327,336 more rows
## # i 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>
```

Secondly, Calculate the mean departure delay for each day.

```
not_cancelled %>%
  group_by(year,month,day) %>%
  summarise(mean=mean(dep_delay), .groups = 'drop')
```

```
## # A tibble: 365 x 4
##     year month   day  mean
##    <int> <int> <int> <dbl>
##  1  2013     1     1 11.4
##  2  2013     1     2 13.7
##  3  2013     1     3 10.9
##  4  2013     1     4  8.97
##  5  2013     1     5  5.73
##  6  2013     1     6  7.15
##  7  2013     1     7  5.42
##  8  2013     1     8  2.56
##  9  2013     1     9  2.30
## 10  2013     1    10  2.84
## # i 355 more rows
```

```
  hist(delays$delay)
```



Summary Function

(A) When do the first and last flights leave each day? Calculate the minimum departure time and maximum departure time every day using min and max value of departure time

```
not_cancelled %>%
  group_by(year,month,day) %>%
  summarise(
    first= min(dep_time),
    last = max(dep_time), .groups = 'drop'
  )
```

```
## # A tibble: 365 x 5
##     year month   day first  last
##    <int> <int> <int> <int> <int>
##  1  2013     1     1   517  2356
##  2  2013     1     2    42  2354
##  3  2013     1     3    32  2349
##  4  2013     1     4    25  2358
##  5  2013     1     5    14  2357
##  6  2013     1     6    16  2355
##  7  2013     1     7    49  2359
##  8  2013     1     8   454  2351
##  9  2013     1     9     2  2252
## 10  2013     1    10     3  2320
## # i 355 more rows
```

On Jan 1 first departure time is 5:17 AM and last departure time is 11:56 PM.

(B) Why is distance to some destinations more variable than to others? Calculates the mean distance and standard deviation distance(in descending order)

```
not_cancelled %>%
  group_by(dest) %>%
  summarise(
    distance_mean=mean(distance),distance_sd=sd(distance)) %>%
  arrange(desc(distance_sd))
```

```
## # A tibble: 104 x 3
##    dest  distance_mean distance_sd
##    <chr>         <dbl>       <dbl>
##  1 EGE           1736.       10.5
##  2 SAN           2437.       10.4
##  3 SFO           2578.       10.2
##  4 HNL           4973.       10.0
##  5 SEA           2413.        9.98
##  6 LAS           2241.        9.91
##  7 PDX           2446.        9.87
##  8 PHX           2141.        9.86
##  9 LAX           2469.        9.66
## 10 IND            652.        9.46
## # i 94 more rows
```

(C) Which destination have the most carries?

```
not_cancelled %>%
  group_by(dest) %>%
  summarise(carriers=n_distinct(carrier)) %>%
  arrange(desc(carriers))
```

```
## # A tibble: 104 x 2
##    dest  carriers
##    <chr>    <int>
##  1 ATL          7
##  2 BOS          7
##  3 CLT          7
##  4 ORD          7
##  5 TPA          7
##  6 AUS          6
##  7 DCA          6
##  8 DTW          6
##  9 IAD          6
## 10 MSP          6
## # i 94 more rows
```

"ATL","BOS","CLT","ORD","TPA" has the most carries

(D) How many flights left before 5am daily? Shows the number of flights departing earlier than 5:00 AM everyday

```
not_cancelled %>%
  group_by(year,month,day) %>%
  summarise(n_early=sum(dep_time<500), .groups = 'drop')
```

```
## # A tibble: 365 x 4
##     year month   day n_early
##    <int> <int> <int>   <int>
##  1  2013     1     1       0
##  2  2013     1     2       3
##  3  2013     1     3       4
##  4  2013     1     4       3
##  5  2013     1     5       3
##  6  2013     1     6       2
##  7  2013     1     7       2
##  8  2013     1     8       1
##  9  2013     1     9       3
## 10  2013     1    10       3
## # i 355 more rows
```

On Jan 1 no flight has departed earlier than 5:00 AM

(E) what proportion of flight are delayed by more than an hour?

```
not_cancelled %>%
  group_by(year,month,day) %>%
  summarise(hour_prop=mean(arr_delay>60),.groups = 'drop')
```

```
## # A tibble: 365 x 4
##     year month   day hour_prop
##    <int> <int> <int>     <dbl>
##  1  2013     1     1    0.0722
##  2  2013     1     2    0.0851
##  3  2013     1     3    0.0567
##  4  2013     1     4    0.0396
##  5  2013     1     5    0.0349
##  6  2013     1     6    0.0470
##  7  2013     1     7    0.0333
##  8  2013     1     8    0.0213
##  9  2013     1     9    0.0202
## 10  2013     1    10    0.0183
## # i 355 more rows
```

(F) Find all groups bigger than a threshold? Display all the flights only to the destination that have more than flights in total.

```
popular_dests<- flights %>%
  group_by(dest) %>%
  filter(n()>365)
popular_dests
```

```
## # A tibble: 332,577 x 19
## # Groups:   dest [77]
##     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
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # i 332,567 more rows
## # i 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>
```

(G) Popular destination Group By destination and get the total flights and arrange in descending order.

```
popular_destination <- flights %>%
  group_by(dest) %>%
  summarise(total_flights = n()) %>%
  filter(total_flights > 365) %>%
  arrange(desc(total_flights))
head(popular_destination$dest)
```

```
## [1] "ORD" "ATL" "LAX" "BOS" "MCO" "CLT"
```

“ORD” is the popular destination

Slice function

```
flights %>% slice_sample(n=5,replace = TRUE)
```

```
## # A tibble: 5 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     3    20      851            845         6     1037           1035
## 2  2013     3     2     1840           1840         0     1951           2020
## 3  2013    12    20     2059           2027        32     2356           2344
## 4  2013     8    31     1524           1529        -5     1805           1821
## 5  2013     9    22     1806           1740        26     2047           2035
## # i 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>
```

```
set.seed(123)
train_data <- flights %>% slice_sample(prop = 0.8)
train_data
```

```
## # A tibble: 269,420 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     4    26      730            655        35      901            820
##  2  2013     2    26     1714           1720        -6     2031           2040
##  3  2013     2    15     1442           1445        -3     1634           1647
##  4  2013     3    27     1444           1445        -1     1556           1604
##  5  2013     6     5     1428           1430        -2     1537           1555
##  6  2013     2    16      613            600        13      731            735
##  7  2013     5     1      848            850        -2      947           1014
##  8  2013    10    20     1918           1855        23     2132           2130
##  9  2013    11    11      825            835       -10      942           1000
## 10  2013     8    31      656            700        -4      904            929
## # i 269,410 more rows
## # i 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>
```

```
test_data <- flights %>% slice_sample(prop = 0.2)
test_data
```

```
## # A tibble: 67,355 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     8    23     1605           1559         6     1700           1721
##  2  2013     5     2     1222           1205        17     1444           1425
##  3  2013     4     7     1957           2000        -3     2204           2208
##  4  2013     4    16      639            640        -1      837            851
##  5  2013     4     2     1132           1135        -3     1311           1255
##  6  2013     8     7     1006            959         7     1116           1114
##  7  2013     9    26     1826           1829        -3     2014           2033
##  8  2013     5    19     1945           1925        20     2252           2250
##  9  2013     7    15     1654           1659        -5     1843           1905
## 10  2013     4    13      741            745        -4      959           1012
## # i 67,345 more rows
## # i 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>
```