Homework\_one

Brooke Wheeler

1/14/2022

library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.1 ──

## ✓ ggplot2 3.3.5 ✓ purrr 0.3.4  
## ✓ tibble 3.1.6 ✓ dplyr 1.0.7  
## ✓ tidyr 1.1.4 ✓ stringr 1.4.0  
## ✓ readr 2.1.1 ✓ forcats 0.5.1

## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(lubridate)

##   
## Attaching package: 'lubridate'

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

library(dplyr)

# 1 Use and show R coding to find the number of days from June 6th 2020 to July 14th 2021

two\_yearago <- ymd(20200606)  
  
last\_year <- ymd(20210714)  
  
two\_yearago - last\_year

## Time difference of -403 days

# 2 Use and show R coding to confirm that the year 1988 was a leap year.

year\_1988 <- ymd(19880101)  
year\_1989 <- ymd(19890101)  
year\_1988 - year\_1989

## Time difference of -366 days

# the year 1988 was a leap year since it had 366 days rather than 365

# 3 Use and show R coding to confirm that the year 1989 was not a leap year

year\_1990 <- ymd(19900101)  
year\_1989 <- ymd(19890101)  
year\_1990 - year\_1989

## Time difference of 365 days

# the year 1989 was not a leap year since it had 365 days rather than 365

# 4 Why is there months() but no dmonths ? (Answer in 3 to four sentences)

## Using durations such as dhours or dminutes or ddays gives you the exact number in seconds. You can’t use dmonths since each month does that have the same amount of days so theres not a standard that can be multiplied to find the exact number of seconds. When using months() you are just saying the amount of months or which month.

# 5 John was born April 11th, 1962. Use and show R coding to determine how old John is in years

birthday <- ymd(19620411)  
(today() %--% birthday)%/% years(1)

## [1] -59

# he is 59 years old

# 6 Modify the flights\_dt coding in the notes or the book to obtain the following partial data table shown below. Show all required coding. (Most of the coding needed is provided in the notes)

library(nycflights13)  
  
make\_datetime\_100 <- function(year, month, day, time) {  
 make\_datetime(year, month, day, time %/% 100, time %% 100)  
}  
  
flights %>%   
 filter(!is.na(dep\_time), !is.na(arr\_time)) %>%   
 mutate(  
 dep\_time = make\_datetime\_100(year, month, day, dep\_time),  
 arr\_time = make\_datetime\_100(year, month, day, arr\_time),  
 sched\_dep\_time = make\_datetime\_100(year, month, day, sched\_dep\_time),  
 sched\_arr\_time = make\_datetime\_100(year, month, day, sched\_arr\_time)  
 ) ->  
  
flights\_dt  
flights\_dt

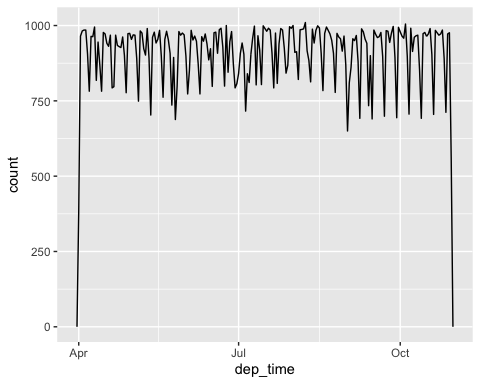
## # A tibble: 328,063 × 19  
## year month day dep\_time sched\_dep\_time dep\_delay  
## <int> <int> <int> <dttm> <dttm> <dbl>  
## 1 2013 1 1 2013-01-01 05:17:00 2013-01-01 05:15:00 2  
## 2 2013 1 1 2013-01-01 05:33:00 2013-01-01 05:29:00 4  
## 3 2013 1 1 2013-01-01 05:42:00 2013-01-01 05:40:00 2  
## 4 2013 1 1 2013-01-01 05:44:00 2013-01-01 05:45:00 -1  
## 5 2013 1 1 2013-01-01 05:54:00 2013-01-01 06:00:00 -6  
## 6 2013 1 1 2013-01-01 05:54:00 2013-01-01 05:58:00 -4  
## 7 2013 1 1 2013-01-01 05:55:00 2013-01-01 06:00:00 -5  
## 8 2013 1 1 2013-01-01 05:57:00 2013-01-01 06:00:00 -3  
## 9 2013 1 1 2013-01-01 05:57:00 2013-01-01 06:00:00 -3  
## 10 2013 1 1 2013-01-01 05:58:00 2013-01-01 06:00:00 -2  
## # … with 328,053 more rows, and 13 more variables: arr\_time <dttm>,  
## # sched\_arr\_time <dttm>, 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>

flights\_dt %>%  
 select(origin, dest, carrier, arr\_time, dep\_time) -> new\_flights  
new\_flights

## # A tibble: 328,063 × 5  
## origin dest carrier arr\_time dep\_time   
## <chr> <chr> <chr> <dttm> <dttm>   
## 1 EWR IAH UA 2013-01-01 08:30:00 2013-01-01 05:17:00  
## 2 LGA IAH UA 2013-01-01 08:50:00 2013-01-01 05:33:00  
## 3 JFK MIA AA 2013-01-01 09:23:00 2013-01-01 05:42:00  
## 4 JFK BQN B6 2013-01-01 10:04:00 2013-01-01 05:44:00  
## 5 LGA ATL DL 2013-01-01 08:12:00 2013-01-01 05:54:00  
## 6 EWR ORD UA 2013-01-01 07:40:00 2013-01-01 05:54:00  
## 7 EWR FLL B6 2013-01-01 09:13:00 2013-01-01 05:55:00  
## 8 LGA IAD EV 2013-01-01 07:09:00 2013-01-01 05:57:00  
## 9 JFK MCO B6 2013-01-01 08:38:00 2013-01-01 05:57:00  
## 10 LGA ORD AA 2013-01-01 07:53:00 2013-01-01 05:58:00  
## # … with 328,053 more rows

# 7 Now, using your table from #6, produce the frequency plot shown which conveys frequency counts for the months of April, July, and October for the year 2013

new\_flights %>%  
 filter(dep\_time >= ymd(20130401) & dep\_time <= ymd(20131030)) %>%  
 ggplot(aes(dep\_time))+  
 geom\_freqpoly(binwidth = 86400)



# 8 Now use dplyr functions to produce a data table that shows arrival times for American Airlines at the Dallas Fort Worth Airport from the LaGuardia airport in New York. Your output should show rows 115 to 125.

new\_flights %>%  
 select(origin, dest, carrier, arr\_time) %>%  
 filter(origin == "LGA",  
 dest == "DFW",  
 carrier == "AA") %>%  
 slice(115:125)

## # A tibble: 11 × 4  
## origin dest carrier arr\_time   
## <chr> <chr> <chr> <dttm>   
## 1 LGA DFW AA 2013-01-09 16:16:00  
## 2 LGA DFW AA 2013-01-09 19:17:00  
## 3 LGA DFW AA 2013-01-09 19:36:00  
## 4 LGA DFW AA 2013-01-09 20:53:00  
## 5 LGA DFW AA 2013-01-09 22:24:00  
## 6 LGA DFW AA 2013-01-10 08:37:00  
## 7 LGA DFW AA 2013-01-10 10:20:00  
## 8 LGA DFW AA 2013-01-10 11:22:00  
## 9 LGA DFW AA 2013-01-10 12:16:00  
## 10 LGA DFW AA 2013-01-10 13:19:00  
## 11 LGA DFW AA 2013-01-10 13:23:00

# 9 Using the first two observational date time designations from your #8 table, Use and show R code to confirm that there are 181 minutes time intervals between them.

row115 <- ymd\_hms("2013-01-09 16:16:00")  
row116 <- ymd\_hms("2013-01-09 19:17:00")  
(row116 -row115) / dminutes(1)

## [1] 181