Data Analysis Lab

Assignment Instructions Complete all questions below. After completing the assignment, knit your document, and download both your .Rmd and knitted output. Upload your files for peer review.

For each response, include comments detailing your response and what each line does.

```
# loading tidyverse and nycflights
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(nycflights13)
```

Question 1. Using the nycflights13 dataset, find all flights that departed in July, August, or September using the helper function between().

```
# creating a pipe that filters flights between months July to September
flights |>
  filter(between(month, 7, 9))
```

```
## # A tibble: 86,326 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     7     1        1           2029       212      236           2359
##  2  2013     7     1        2           2359         3      344            344
##  3  2013     7     1       29           2245       104      151              1
##  4  2013     7     1       43           2130       193      322             14
##  5  2013     7     1       44           2150       174      300            100
##  6  2013     7     1       46           2051       235      304           2358
##  7  2013     7     1       48           2001       287      308           2305
##  8  2013     7     1       58           2155       183      335             43
##  9  2013     7     1      100           2146       194      327             30
## 10  2013     7     1      100           2245       135      337            135
## # i 86,316 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>
```

Question 2. Using the nycflights13 dataset sort flights to find the 10 flights that flew the furthest. Put them in order of fastest to slowest.

```
flights |>
  # find the flights that flew furthest
  arrange(desc(distance)) |>
  # take the top 10 flights
  slice_head(n = 10) |>
  # sort these flights from fastest to slowest
  arrange(desc(distance / air_time))
```

```
## # A tibble: 10 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     6     1019            900        79     1558           1530
##  2  2013     1     7     1042            900       102     1620           1530
##  3  2013     1     3      914            900        14     1504           1530
##  4  2013     1    10      859            900        -1     1449           1530
##  5  2013     1     5      858            900        -2     1519           1530
##  6  2013     1     2      909            900         9     1525           1530
##  7  2013     1     4      900            900         0     1516           1530
##  8  2013     1     9      641            900      1301     1242           1530
##  9  2013     1     8      901            900         1     1504           1530
## 10  2013     1     1      857            900        -3     1516           1530
## # 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>
```

Question 3. Using the nycflights13 dataset, calculate a new variable called "hr_delay" and arrange the flights dataset in order of the arrival delays in hours (longest delays at the top). Put the new variable you created just before the departure time.Hint: use the experimental argument .before.

```
flights |>
  # create new variable hr_delay, putting this before dep_time
  mutate(hr_delay = arr_delay / 60, .before = dep_time) |>
  # arrange dataset from longest to shortest delays
  arrange(desc(hr_delay))
```

```
## # A tibble: 336,776 x 20
##     year month   day hr_delay dep_time sched_dep_time dep_delay arr_time
##    <int> <int> <int>    <dbl>    <int>          <int>     <dbl>    <int>
##  1  2013     1     9     21.2      641            900      1301     1242
##  2  2013     6    15     18.8     1432           1935      1137     1607
##  3  2013     1    10     18.5     1121           1635      1126     1239
##  4  2013     9    20     16.8     1139           1845      1014     1457
##  5  2013     7    22     16.5      845           1600      1005     1044
##  6  2013     4    10     15.5     1100           1900       960     1342
##  7  2013     3    17     15.2     2321            810       911      135
```

```
##  8  2013     7    22     14.9     2257            759       898      121
##  9  2013    12     5     14.6      756           1700       896     1058
## 10  2013     5     3     14.6     1133           2055       878     1250
## # i 336,766 more rows
## # i 12 more variables: sched_arr_time <int>, 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>
```

Question 4. Using the nycflights13 dataset, find the most popular destinations (those with more than 2000 flights) and show the destination, the date info, the carrier. Then show just the number of flights for each popular destination.

```
# find the most popular destinations with more than 2000 flights
popular_desinations <- flights |>
  # group the dataset by destinations
  group_by(dest) |>
  # filter thos out, with more than 2000 flights
  filter(n() > 2000) |>
  # select destination, date, carrier
  select(dest, year, month, day, carrier)

# show results
popular_desinations
```

```
## # A tibble: 302,969 x 5
## # Groups:   dest [46]
##    dest   year month   day carrier
##    <chr> <int> <int> <int> <chr>
##  1 IAH    2013     1     1 UA
##  2 IAH    2013     1     1 UA
##  3 MIA    2013     1     1 AA
##  4 ATL    2013     1     1 DL
##  5 ORD    2013     1     1 UA
##  6 FLL    2013     1     1 B6
##  7 IAD    2013     1     1 EV
##  8 MCO    2013     1     1 B6
##  9 ORD    2013     1     1 AA
## 10 PBI    2013     1     1 B6
## # i 302,959 more rows
```

```
# show just the numbere of flights for each popular destination
popular_desinations |>
  count(dest) |>
  arrange(desc(n))
```

```
## # A tibble: 46 x 2
## # Groups:   dest [46]
##    dest      n
##    <chr> <int>
##  1 ORD   17283
##  2 ATL   17215
##  3 LAX   16174
```

```
##  4 BOS   15508
##  5 MCO   14082
##  6 CLT   14064
##  7 SFO   13331
##  8 FLL   12055
##  9 MIA   11728
## 10 DCA    9705
## # i 36 more rows
```

Question 5. Using the nycflights13 dataset, find the flight information (flight number, origin, destination, carrier, number of flights in the year, and percent late) for the flight numbers with the highest percentage of arrival delays. Only include the flight numbers that have over 100 flights in the year.