Data Analysis Lab

Me Myself

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.

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

```
# Opening the libraries that might be used in this assignment.
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- 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(dplyr)
library(nycflights13)
#Creating a filter on the data to extract only flights between month 7 and 9.
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.

```
# Using the arrange function to sort flights.
# Sorting them by the distance column in descending order for greater distances at first,
# then mph in ascending order, for fastest first.
# The mutate function calculates a new column with the flight speeds to find the fastest.
flights %>% mutate(mph = distance / air_time * 60) %>%
            arrange(-distance,mph) %>%
            head(10)
```

```
## # A tibble: 10 x 20
##     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     2     6      853            900        -7     1542           1540
##  2  2013     3    15     1001           1000         1     1551           1530
##  3  2013     3    17     1006           1000         6     1607           1530
##  4  2013     3    16     1001           1000         1     1544           1530
##  5  2013     2     5      900            900         0     1555           1540
##  6  2013     3    14      958           1000        -2     1542           1530
##  7  2013    11    20     1006           1000         6     1639           1555
##  8  2013     4     3      957           1000        -3     1535           1510
##  9  2013    11    11      957           1000        -3     1627           1555
## 10  2013    11    10      957           1000        -3     1625           1555
## # i 12 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>, mph <dbl>
```

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.

```
# Function mutate() creates a new variable; function arrange() sorts it in descending order;
# finally the relocate() function is used with the experimental argument .before to reorganise the colu
flights %>% mutate(hr_delay = dep_delay/60) %>%
            arrange(-hr_delay) %>%
            relocate(hr_delay, .before = dep_time)
```

```
## # 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     21.7      641            900      1301     1242
##  2  2013     6    15     19.0     1432           1935      1137     1607
##  3  2013     1    10     18.8     1121           1635      1126     1239
##  4  2013     9    20     16.9     1139           1845      1014     1457
##  5  2013     7    22     16.8      845           1600      1005     1044
##  6  2013     4    10     16       1100           1900       960     1342
```

```
##  7  2013     3    17     15.2     2321            810       911      135
##  8  2013     6    27     15.0      959           1900       899     1236
##  9  2013     7    22     15.0     2257            759       898      121
## 10  2013    12     5     14.9      756           1700       896     1058
## # 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.

```
# First two variables are created, one that counts destinations and filters for the ones only over 2000
# The second provides the other required information.
count <- flights %>%
                      count(dest) %>%
                      filter(n>2000)
selects <- flights %>%
                      distinct(dest,year,month,day,carrier)
# The two variables are joined in one table to provide all the required info.
df<-left_join(count,selects)
```

```
## Joining with `by = join_by(dest)`
```

```
df
```

```
## # A tibble: 59,206 x 6
##    dest      n  year month   day carrier
##    <chr> <int> <int> <int> <int> <chr>
##  1 ATL   17215  2013     1     1 DL
##  2 ATL   17215  2013     1     1 MQ
##  3 ATL   17215  2013     1     1 FL
##  4 ATL   17215  2013     1     1 EV
##  5 ATL   17215  2013     1     2 DL
##  6 ATL   17215  2013     1     2 FL
##  7 ATL   17215  2013     1     2 MQ
##  8 ATL   17215  2013     1     2 EV
##  9 ATL   17215  2013     1     3 MQ
## 10 ATL   17215  2013     1     3 DL
## # i 59,196 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.

```
#Creating two variables: first with delayed flights; second with a filter for greater than 100 flights.

df1<-flights %>% group_by(flight,year) %>% filter(arr_delay>0) %>% summarise(delayed_flights=n())
```

```
## `summarise()` has grouped output by 'flight'. You can override using the
## `.groups` argument.
```

```r
df2<-flights %>% group_by(flight,year) %>%summarise (no_flights_in_year=n()) %>%filter(no_flights_in_yea
```

```
## `summarise()` has grouped output by 'flight'. You can override using the
## `.groups` argument.
```

```r
# Using a join to combine the two first variables from the data
df<-inner_join(df1,df2)
```

```
## Joining with `by = join_by(flight, year)`
```

```r
# Two new variables where the percent late is created and distinct values are selected to be combined i
df3<- df %>% transmute(percent_late=(delayed_flights/no_flights_in_year)*100,flight,year,no_flights_in_y


df4<-flights %>% distinct(flight,year,origin,dest,carrier)


left_join(df3,df4)
```

```
## Joining with `by = join_by(flight, year)`
```

```
## # A tibble: 4,968 x 7
## # Groups:   flight [1,157]
##    percent_late flight  year no_flights_in_year origin dest  carrier
##           <dbl>  <int> <int>              <int> <chr>  <chr> <chr>
##  1         37.4      1  2013                701 JFK    LAX   AA
##  2         37.4      1  2013                701 JFK    FLL   B6
##  3         37.4      1  2013                701 EWR    PBI   UA
##  4         37.4      1  2013                701 LGA    MDW   WN
##  5         37.4      1  2013                701 JFK    SJU   DL
##  6         37.4      1  2013                701 EWR    ORD   UA
##  7         33.4      3  2013                631 JFK    FLL   B6
##  8         33.4      3  2013                631 JFK    LAX   AA
##  9         33.4      3  2013                631 JFK    SJU   B6
## 10         33.4      3  2013                631 LGA    MDW   WN
## # i 4,958 more rows
```