Relational Data Assignment

Vickie Gray

2023-01-04

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. Ensure you test your functions with sufficient test cases to identify and correct any potential bugs.

```
library(tidyverse)
```

Required Libraries

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0      v purrr   1.0.0
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.5.0
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(nycflights13)
library(Lahman)
library(babynames)
library(nasaweather)
```

```
##
## Attaching package: 'nasaweather'
##
## The following object is masked from 'package:dplyr':
##
##     storms
```

```
library(maps)
```

```
##
## Attaching package: 'maps'
##
## The following object is masked from 'package:purrr':
##
##     map
```

Question 1. Identify the primary keys in the following datasets. Be sure to show that you have the primary key by showing there are no duplicate entries.

Lahman::Batting babynames::babynames nasaweather::atmos

```
# The primary key for Lahman::Batting is a combination of
# playerID, stint, yearID, and teamID.
# A filter on the count confirms 0 rows

# Lahman::Batting # The full dataset for reference

Lahman::Batting %>%
  count(playerID, stint, yearID, teamID) %>%
  filter(n > 1) # filter to find duplicates; result confirms 0 rows
```

```
## [1] playerID stint    yearID   teamID   n
## <0 rows> (or 0-length row.names)
```

```
# The primary key for babynames::babynames is a combination of
# year, sex, name.
# A filter on the count confirms 0 rows

# babynames::babynames # The full dataset

babynames::babynames %>%
  count(year, sex, name) %>%
  filter(n > 1)
```

```
## # A tibble: 0 x 4
## # ... with 4 variables: year <dbl>, sex <chr>, name <chr>, n <int>
```

```
# The primary key for nasaweather::atmos is a combination of
# lat, long, year, month.
# A filter on the count confirms 0 rows

# nasaweather::atmos # The full dataset

nasaweather::atmos %>%
  count(lat, long, year, month) %>%
  filter(n > 1)
```

```
## # A tibble: 0 x 5
## # ... with 5 variables: lat <dbl>, long <dbl>, year <int>, month <int>, n <int>
```

Question 2. What is the relationship between the "Batting", "Master", and "Salaries" tables in the "Lahman" package? What are the keys for each dataset and how do they relate to each other?

```
# According to Help on "Lahman", each player is assigned a unique code (playerID). All of the informati

# In both Batting and Salaries, the variables "playerID", "yearID", "teamID", "lgID" combine to form a

# When joined on this key, the combined table includes consistent salary information for players after
```

```r
# Batting # Full dataset
# Salaries # Full dataset
# Master # Full dataset
```

```r
Bat_Sal <- Batting %>%
  left_join(Salaries,
            by = c("playerID", "yearID", "teamID", "lgID")) %>%
  select("playerID", "yearID", "teamID", "lgID", "salary")

head(Bat_Sal) # The earliest records are dated 1871
```

```
##    playerID yearID teamID lgID  salary
## 1 aardsda01   2004    SFN   NL  300000
## 2 aardsda01   2006    CHN   NL      NA
## 3 aardsda01   2007    CHA   AL  387500
## 4 aardsda01   2008    BOS   AL  403250
## 5 aardsda01   2009    SEA   AL  419000
## 6 aardsda01   2010    SEA   AL 2750000
```

```r
tail(Bat_Sal) # The most recent records are from 2016
```

```
##        playerID yearID teamID lgID   salary
## 97884 zieglbr01   2013    ARI   NL  3150000
## 97885 zimmejo02   2013    WAS   NL  5350000
## 97886 zimmery01   2013    WAS   NL 14100000
## 97887  zitoba01   2013    SFN   NL 20000000
## 97888 zobribe01   2013    TBA   AL  5687300
## 97889 zuninmi01   2013    SEA   AL       NA
```

```r
Player_Sal <- Master %>%
  full_join(Bat_Sal, by = "playerID") %>%
  select("playerID", "nameLast", "nameGiven", "yearID", "teamID", "lgID","salary") %>%
  arrange("nameLast")

head(Player_Sal)
```

```
##    playerID nameLast   nameGiven yearID teamID lgID  salary
## 1 aardsda01  Aardsma David Allan   2004    SFN   NL  300000
## 2 aardsda01  Aardsma David Allan   2006    CHN   NL      NA
## 3 aardsda01  Aardsma David Allan   2007    CHA   AL  387500
## 4 aardsda01  Aardsma David Allan   2008    BOS   AL  403250
## 5 aardsda01  Aardsma David Allan   2009    SEA   AL  419000
## 6 aardsda01  Aardsma David Allan   2010    SEA   AL 2750000
```

```r
tail(Player_Sal)
```

```
##        playerID nameLast       nameGiven yearID teamID lgID salary
## 98131 zuverge01 Zuverink          George   1958    BAL   AL     NA
## 98132 zuverge01 Zuverink          George   1959    BAL   AL     NA
## 98133 zwilldu01 Zwilling Edward Harrison   1910    CHA   AL     NA
## 98134 zwilldu01 Zwilling Edward Harrison   1914    CHF   FL     NA
## 98135 zwilldu01 Zwilling Edward Harrison   1915    CHF   FL     NA
## 98136 zwilldu01 Zwilling Edward Harrison   1916    CHN   NL     NA
```

Question 3. Load the "nycflights13" library. Use an appropriate join to add a column containing the airline name to the "flights" dataset present in the library. Be sure to put the carrier code and name in the first two columns of the result so we can see them. Save the result as "flights2".

```
flights2 <- flights %>%
  left_join(airlines, by = "carrier") %>%
  select(carrier, name, everything())

head(flights2)
```

```
## # A tibble: 6 x 20
##   carrier name          year month   day dep_t~1 sched~2 dep_d~3 arr_t~4 sched~5
##   <chr>   <chr>        <int> <int> <int>   <int>   <int>   <dbl>   <int>   <int>
## 1 UA      United Air ~  2013     1     1     517     515       2     830     819
## 2 UA      United Air ~  2013     1     1     533     529       4     850     830
## 3 AA      American Ai~  2013     1     1     542     540       2     923     850
## 4 B6      JetBlue Air~  2013     1     1     544     545      -1    1004    1022
## 5 DL      Delta Air L~  2013     1     1     554     600      -6     812     837
## 6 UA      United Air ~  2013     1     1     554     558      -4     740     728
## # ... with 10 more variables: arr_delay <dbl>, flight <int>, tailnum <chr>,
## #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
## #   minute <dbl>, time_hour <dttm>, and abbreviated variable names 1: dep_time,
## #   2: sched_dep_time, 3: dep_delay, 4: arr_time, 5: sched_arr_time
```

```
tail(flights2)
```

```
## # A tibble: 6 x 20
##   carrier name          year month   day dep_t~1 sched~2 dep_d~3 arr_t~4 sched~5
##   <chr>   <chr>        <int> <int> <int>   <int>   <int>   <dbl>   <int>   <int>
## 1 EV      ExpressJet ~  2013     9    30      NA    1842      NA      NA    2019
## 2 9E      Endeavor Ai~  2013     9    30      NA    1455      NA      NA    1634
## 3 9E      Endeavor Ai~  2013     9    30      NA    2200      NA      NA    2312
## 4 MQ      Envoy Air     2013     9    30      NA    1210      NA      NA    1330
## 5 MQ      Envoy Air     2013     9    30      NA    1159      NA      NA    1344
## 6 MQ      Envoy Air     2013     9    30      NA     840      NA      NA    1020
## # ... with 10 more variables: arr_delay <dbl>, flight <int>, tailnum <chr>,
## #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
## #   minute <dbl>, time_hour <dttm>, and abbreviated variable names 1: dep_time,
## #   2: sched_dep_time, 3: dep_delay, 4: arr_time, 5: sched_arr_time
```

Question 4. Use an appropriate join to add the airport name to the "flights2" dataset you got above. The codes and names of the airports are in the "airports" dataset of the "nycflights13" package. Put the carrier and carrier name first followed by the destination and destination name, then everything else.

```
airports2 <- airports %>%
  select(everything()) %>%
  rename("airport_name" = "name") # Rename the airports variable "name" so it doesn't conflict with air

flights2 <- flights2 %>%
  left_join(airports2, by = c("dest" = "faa")) %>%
  select(carrier, name, airport_name, everything())

head(flights2)
```

```
## # A tibble: 6 x 27
##   carrier name         airpo~1  year month   day dep_t~2 sched~3 dep_d~4 arr_t~5
##   <chr>   <chr>        <chr>   <int> <int> <int>   <int>   <int>   <dbl>   <int>
## 1 UA      United Air ~ George~  2013     1     1     517     515       2     830
## 2 UA      United Air ~ George~  2013     1     1     533     529       4     850
## 3 AA      American Ai~ Miami ~  2013     1     1     542     540       2     923
## 4 B6      JetBlue Air~ <NA>     2013     1     1     544     545      -1    1004
## 5 DL      Delta Air L~ Hartsf~  2013     1     1     554     600      -6     812
## 6 UA      United Air ~ Chicag~  2013     1     1     554     558      -4     740
## # ... with 17 more variables: sched_arr_time <int>, arr_delay <dbl>,
## #   flight <int>, tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>,
## #   distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>, lat <dbl>,
## #   lon <dbl>, alt <dbl>, tz <dbl>, dst <chr>, tzone <chr>, and abbreviated
## #   variable names 1: airport_name, 2: dep_time, 3: sched_dep_time,
## #   4: dep_delay, 5: arr_time
```

```
tail(flights2)
```

```
## # A tibble: 6 x 27
##   carrier name         airpo~1  year month   day dep_t~2 sched~3 dep_d~4 arr_t~5
##   <chr>   <chr>        <chr>   <int> <int> <int>   <int>   <int>   <dbl>   <int>
## 1 EV      ExpressJet ~ Nashvi~  2013     9    30      NA    1842      NA      NA
## 2 9E      Endeavor Ai~ Ronald~  2013     9    30      NA    1455      NA      NA
## 3 9E      Endeavor Ai~ Syracu~  2013     9    30      NA    2200      NA      NA
## 4 MQ      Envoy Air    Nashvi~  2013     9    30      NA    1210      NA      NA
## 5 MQ      Envoy Air    Clevel~  2013     9    30      NA    1159      NA      NA
## 6 MQ      Envoy Air    Raleig~  2013     9    30      NA     840      NA      NA
## # ... with 17 more variables: sched_arr_time <int>, arr_delay <dbl>,
## #   flight <int>, tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>,
## #   distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>, lat <dbl>,
## #   lon <dbl>, alt <dbl>, tz <dbl>, dst <chr>, tzone <chr>, and abbreviated
## #   variable names 1: airport_name, 2: dep_time, 3: sched_dep_time,
## #   4: dep_delay, 5: arr_time
```

Question 5. The "nycflights13" library and the code to create spatial map is provided for you. Now compute the average delay by destination, then join on the airports dataframe so you can show the spatial distribution of delays.

- Use the size or colour of the points to display the average delay for each airport.
- Add the location of the origin and destination (i.e. the lat and lon) to flights.
- Compute the average delay by destination.

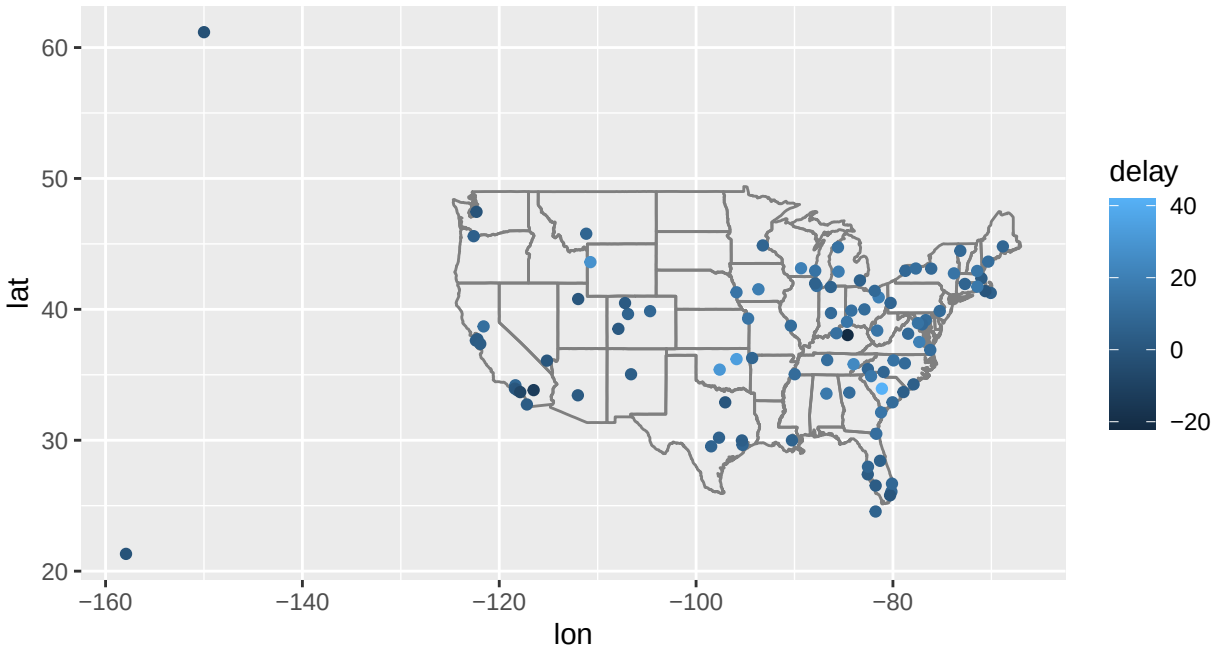
Use the textbook for reference.

```
# First, compute the average delay, grouped by dest
avg_delay <- # Create an object to hold the new dataset
  flights %>% # Pipe flights
  group_by(dest) %>% # group by the destination variable
  summarise(delay = mean(arr_delay, na.rm = TRUE)) %>%
  # arrival delay NA's are cancelled flights, so we remove those
  # calculate the mean arrival delay for each destination, and save that information to the object dela
  inner_join (airports, by = c(dest = "faa")) # Join the avg_delay dataset to the airports dataset on d
```

```
# The airports dataset gives us the lat and lon data required by the question to create the ggplot map
avg_delay # The combined table
```

```
## # A tibble: 101 x 9
##    dest  delay name                           lat    lon   alt    tz dst   tzone
##    <chr> <dbl> <chr>                        <dbl>  <dbl> <dbl> <dbl> <chr> <chr>
##  1 ABQ    4.38 Albuquerque International S~  35.0 -107.   5355    -7 A     Amer~
##  2 ACK    4.85 Nantucket Mem                41.3  -70.1    48    -5 A     Amer~
##  3 ALB   14.4  Albany Intl                  42.7  -73.8   285    -5 A     Amer~
##  4 ANC   -2.5  Ted Stevens Anchorage Intl    61.2 -150.    152    -9 A     Amer~
##  5 ATL   11.3  Hartsfield Jackson Atlanta ~  33.6  -84.4  1026    -5 A     Amer~
##  6 AUS    6.02 Austin Bergstrom Intl        30.2  -97.7   542    -6 A     Amer~
##  7 AVL    8.00 Asheville Regional Airport   35.4  -82.5  2165    -5 A     Amer~
##  8 BDL    7.05 Bradley Intl                 41.9  -72.7   173    -5 A     Amer~
##  9 BGR    8.03 Bangor Intl                  44.8  -68.8   192    -5 A     Amer~
## 10 BHM   16.9  Birmingham Intl              33.6  -86.8   644    -6 A     Amer~
## # ... with 91 more rows
```

```
avg_delay %>% # Now we can pipe the combined table to ggplot
  ggplot(aes(lon, lat, colour = delay)) + # the colour of the airports on the map is going to reflect t
    borders("state") +
    geom_point() +
    coord_quickmap()
```



Question 6. Use a set operation function to find which airport codes from flights are not in the airports dataset.

```
# Setdiff will return the all values from the first set that are not in the second set; There are four

as_tibble(setdiff(flights$dest, airports$faa))
```

```
## # A tibble: 4 x 1
##   value
##   <chr>
## 1 BQN
## 2 SJU
## 3 STT
## 4 PSE
```