Untitled

Michael Rose

January 9, 2018

There are 3 families of verbs that are designed to work with relational data:

- Mutating Joins: Adds new variables to one data frame from matching observations in another
- Filtering Joins: Filters observations from one data frame based on whether or not they match an observation in the other table
- Set Operations: Treats observations as if they were set elements

```
# tables
airlines
```

```
## # A tibble: 16 x 2
##    carrier name
##    <chr>   <chr>
##  1 9E      Endeavor Air Inc.
##  2 AA      American Airlines Inc.
##  3 AS      Alaska Airlines Inc.
##  4 B6      JetBlue Airways
##  5 DL      Delta Air Lines Inc.
##  6 EV      ExpressJet Airlines Inc.
##  7 F9      Frontier Airlines Inc.
##  8 FL      AirTran Airways Corporation
##  9 HA      Hawaiian Airlines Inc.
## 10 MQ      Envoy Air
## 11 OO      SkyWest Airlines Inc.
## 12 UA      United Air Lines Inc.
## 13 US      US Airways Inc.
## 14 VX      Virgin America
## 15 WN      Southwest Airlines Co.
## 16 YV      Mesa Airlines Inc.
```

```
airports
```

```
## # A tibble: 1,458 x 8
##    faa   name                   lat    lon   alt    tz dst   tzone
##    <chr> <chr>                <dbl>  <dbl> <int> <dbl> <chr> <chr>
##  1 04G   Lansdowne Airport     41.1  -80.6  1044    -5 A     America/New_~
##  2 06A   Moton Field Municip~  32.5  -85.7   264    -6 A     America/Chic~
##  3 06C   Schaumburg Regional   42.0  -88.1   801    -6 A     America/Chic~
##  4 06N   Randall Airport       41.4  -74.4   523    -5 A     America/New_~
##  5 09J   Jekyll Island Airpo~  31.1  -81.4    11    -5 A     America/New_~
##  6 0A9   Elizabethton Munici~  36.4  -82.2  1593    -5 A     America/New_~
##  7 0G6   Williams County Air~  41.5  -84.5   730    -5 A     America/New_~
##  8 0G7   Finger Lakes Region~  42.9  -76.8   492    -5 A     America/New_~
##  9 0P2   Shoestring Aviation~  39.8  -76.6  1000    -5 U     America/New_~
## 10 0S9   Jefferson County In~  48.1 -123.    108    -8 A     America/Los_~
## # ... with 1,448 more rows
```

```
planes
```

```
## # A tibble: 3,322 x 9
```

```
##    tailnum  year type      manufacturer  model  engines seats speed engine
##    <chr>   <int> <chr>     <chr>         <chr>    <int> <int> <int> <chr>
##  1 N10156   2004 Fixed wi~ EMBRAER       EMB-1~       2    55    NA Turbo~
##  2 N102UW   1998 Fixed wi~ AIRBUS INDUS~ A320-~       2   182    NA Turbo~
##  3 N103US   1999 Fixed wi~ AIRBUS INDUS~ A320-~       2   182    NA Turbo~
##  4 N104UW   1999 Fixed wi~ AIRBUS INDUS~ A320-~       2   182    NA Turbo~
##  5 N10575   2002 Fixed wi~ EMBRAER       EMB-1~       2    55    NA Turbo~
##  6 N105UW   1999 Fixed wi~ AIRBUS INDUS~ A320-~       2   182    NA Turbo~
##  7 N107US   1999 Fixed wi~ AIRBUS INDUS~ A320-~       2   182    NA Turbo~
##  8 N108UW   1999 Fixed wi~ AIRBUS INDUS~ A320-~       2   182    NA Turbo~
##  9 N109UW   1999 Fixed wi~ AIRBUS INDUS~ A320-~       2   182    NA Turbo~
## 10 N110UW   1999 Fixed wi~ AIRBUS INDUS~ A320-~       2   182    NA Turbo~
## # ... with 3,312 more rows
```

```
weather
```

```
## # A tibble: 26,130 x 15
##    origin  year month   day  hour  temp  dewp humid wind_dir wind_speed
##    <chr>  <dbl> <dbl> <int> <int> <dbl> <dbl> <dbl>    <dbl>      <dbl>
##  1 EWR     2013     1     1     0  37.0  21.9  54.0      230      10.4
##  2 EWR     2013     1     1     1  37.0  21.9  54.0      230      13.8
##  3 EWR     2013     1     1     2  37.9  21.9  52.1      230      12.7
##  4 EWR     2013     1     1     3  37.9  23    54.5      230      13.8
##  5 EWR     2013     1     1     4  37.9  24.1  57.0      240      15.0
##  6 EWR     2013     1     1     6  39.0  26.1  59.4      270      10.4
##  7 EWR     2013     1     1     7  39.0  27.0  61.6      250       8.06
##  8 EWR     2013     1     1     8  39.0  28.0  64.4      240      11.5
##  9 EWR     2013     1     1     9  39.9  28.0  62.2      250      12.7
## 10 EWR     2013     1     1    10  39.0  28.0  64.4      260      12.7
## # ... with 26,120 more rows, and 5 more variables: wind_gust <dbl>,
## #   precip <dbl>, pressure <dbl>, visib <dbl>, time_hour <dttm>
```

```
# Imagine you wanted to draw the route each plane flies from its origin to its destination. What variab

# From the flights table we would use origin and dest. From the airports table we would use the longitu

# I forgot to draw the relationship between weather and airports. What is the relationshop and how shou

# The variable origin in weather is matched with faa in airports

# Weather only contains information for the origin NYC airports. If it contained weather records for al

# dest

# We know that some days of the year are "special," and fewer people than usual fly on them. How might

# One could add a table of special dates, where they are classified as days in which < x people flew. T
```

Keys

Two types of keys:

- Primary Keys uniquely identify an observation in its own table. For example, planes$tailnum is a primary key because it uniquely identifies each plane in the planes table.

- Foreign Keys uniquely identify an observation in another table. For example, flights$tailnum is a foreign key because it appears in the flights table where it matches each flight to a unique plane.

```
# A primary key should consist of only unique values. One way to test if something is a primary key is

planes %>%
  count(tailnum) %>%
  filter(n > 1)
```

```
## # A tibble: 0 x 2
## # ... with 2 variables: tailnum <chr>, n <int>
```

```
weather %>%
  count(year, month, day, hour, origin) %>%
  filter(n > 1)
```

```
## # A tibble: 0 x 6
## # ... with 6 variables: year <dbl>, month <dbl>, day <int>, hour <int>,
## #   origin <chr>, n <int>
```

```
# sometimes a table doesn't have an explicit primary key

flights %>%
  count(year, month, day, flight) %>%
  filter(n > 1)
```

```
## # A tibble: 29,768 x 5
##     year month   day flight     n
##    <int> <int> <int>  <int> <int>
##  1  2013     1     1      1     2
##  2  2013     1     1      3     2
##  3  2013     1     1      4     2
##  4  2013     1     1     11     3
##  5  2013     1     1     15     2
##  6  2013     1     1     21     2
##  7  2013     1     1     27     4
##  8  2013     1     1     31     2
##  9  2013     1     1     32     2
## 10  2013     1     1     35     2
## # ... with 29,758 more rows
```

```
flights %>%
  count(year, month, day, tailnum) %>%
  filter(n > 1)
```

```
## # A tibble: 64,928 x 5
##     year month   day tailnum     n
##    <int> <int> <int> <chr>   <int>
##  1  2013     1     1 NOEGMQ      2
##  2  2013     1     1 N11189      2
##  3  2013     1     1 N11536      2
##  4  2013     1     1 N11544      3
##  5  2013     1     1 N11551      2
##  6  2013     1     1 N12540      2
##  7  2013     1     1 N12567      2
##  8  2013     1     1 N13123      2
##  9  2013     1     1 N13538      3
```

```
## 10  2013     1     1 N13566      3
## # ... with 64,918 more rows
```

```
# If a table lacks a primary key, sometimes it is useful to add one using mutate and row_number(). This

# adding a surrogate key to flights

flights %>%
  arrange(year, month, day, sched_dep_time, carrier, flight) %>%
  mutate(flight_id = row_number()) %>%
  glimpse()
```

```
## Observations: 336,776
## Variables: 20
## $ year           <int> 2013, 2013, 2013, 2013, 2013, 2013, 2013, 2013,...
## $ month          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ day            <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ dep_time       <int> 517, 533, 542, 544, 554, 559, 558, 559, 558, 55...
## $ sched_dep_time <int> 515, 529, 540, 545, 558, 559, 600, 600, 600, 60...
## $ dep_delay      <dbl> 2, 4, 2, -1, -4, 0, -2, -1, -2, -2, -3, NA, 1, ...
## $ arr_time       <int> 830, 850, 923, 1004, 740, 702, 753, 941, 849, 8...
## $ sched_arr_time <int> 819, 830, 850, 1022, 728, 706, 745, 910, 851, 8...
## $ arr_delay      <dbl> 11, 20, 33, -18, 12, -4, 8, 31, -2, -3, -8, NA,...
## $ carrier        <chr> "UA", "UA", "AA", "B6", "UA", "B6", "AA", "AA",...
## $ flight         <int> 1545, 1714, 1141, 725, 1696, 1806, 301, 707, 49...
## $ tailnum        <chr> "N14228", "N24211", "N619AA", "N804JB", "N39463...
## $ origin         <chr> "EWR", "LGA", "JFK", "JFK", "EWR", "JFK", "LGA"...
## $ dest           <chr> "IAH", "IAH", "MIA", "BQN", "ORD", "BOS", "ORD"...
## $ air_time       <dbl> 227, 227, 160, 183, 150, 44, 138, 257, 149, 158...
## $ distance       <dbl> 1400, 1416, 1089, 1576, 719, 187, 733, 1389, 10...
## $ hour           <dbl> 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,...
## $ minute         <dbl> 15, 29, 40, 45, 58, 59, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ time_hour      <dttm> 2013-01-01 05:00:00, 2013-01-01 05:00:00, 2013...
## $ flight_id      <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, ...
```

```
# Identify the keys in the following sets

# playerID, yearID, stint

Lahman::Batting %>%
  group_by(playerID, yearID, stint) %>%
  filter(n() > 1) %>%
  nrow()
```

```
## [1] 0
```

```
# year, sex, name

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

```
## [1] 0
```

```
# location and time of the measurement, lat, long, year, month
```

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

```
## [1] 0
```

```
# id

fueleconomy::vehicles %>%
  group_by(id) %>%
  filter(n() > 1) %>%
  nrow()
```

```
## [1] 0
```

```
# There is no primary key for diamonds. Since there are less distinct rows than there are number of row

ggplot2::diamonds %>%
  distinct() %>%
  nrow()
```

```
## [1] 53794
```

```
nrow(ggplot2::diamonds)
```

```
## [1] 53940
```

Mutating Joins

A mutating join allows you to combine variables from two tables. It first matches observations by their keys, then copies across variables from one table to the other.

```
# to see whats going on, we make a narrower data set

flights2 <- flights %>%
  select(year:day, hour, origin, dest, tailnum, carrier)

flights2
```

```
## # A tibble: 336,776 x 8
##     year month   day  hour origin dest  tailnum carrier
##    <int> <int> <int> <dbl> <chr>  <chr> <chr>   <chr>
##  1  2013     1     1     5 EWR    IAH   N14228  UA
##  2  2013     1     1     5 LGA    IAH   N24211  UA
##  3  2013     1     1     5 JFK    MIA   N619AA  AA
##  4  2013     1     1     5 JFK    BQN   N804JB  B6
##  5  2013     1     1     6 LGA    ATL   N668DN  DL
##  6  2013     1     1     5 EWR    ORD   N39463  UA
##  7  2013     1     1     6 EWR    FLL   N516JB  B6
##  8  2013     1     1     6 LGA    IAD   N829AS  EV
##  9  2013     1     1     6 JFK    MCO   N593JB  B6
## 10  2013     1     1     6 LGA    ORD   N3ALAA  AA
## # ... with 336,766 more rows
```

```
# add the full airline name to the flights 2 data
```

```
flights2 %>%
  select(-origin, -dest) %>%
  left_join(airlines, by = "carrier")
```

```
## # A tibble: 336,776 x 7
##     year month   day  hour tailnum carrier name
##    <int> <int> <int> <dbl> <chr>   <chr>   <chr>
##  1  2013     1     1     5 N14228  UA      United Air Lines Inc.
##  2  2013     1     1     5 N24211  UA      United Air Lines Inc.
##  3  2013     1     1     5 N619AA  AA      American Airlines Inc.
##  4  2013     1     1     5 N804JB  B6      JetBlue Airways
##  5  2013     1     1     6 N668DN  DL      Delta Air Lines Inc.
##  6  2013     1     1     5 N39463  UA      United Air Lines Inc.
##  7  2013     1     1     6 N516JB  B6      JetBlue Airways
##  8  2013     1     1     6 N829AS  EV      ExpressJet Airlines Inc.
##  9  2013     1     1     6 N593JB  B6      JetBlue Airways
## 10  2013     1     1     6 N3ALAA  AA      American Airlines Inc.
## # ... with 336,766 more rows
```

```
# We added the name variable above. We can do the same thing with the following

flights2 %>%
  select(-origin, -dest) %>%
  mutate(name = airlines$name[match(carrier, airlines$carrier)])
```

```
## # A tibble: 336,776 x 7
##     year month   day  hour tailnum carrier name
##    <int> <int> <int> <dbl> <chr>   <chr>   <chr>
##  1  2013     1     1     5 N14228  UA      United Air Lines Inc.
##  2  2013     1     1     5 N24211  UA      United Air Lines Inc.
##  3  2013     1     1     5 N619AA  AA      American Airlines Inc.
##  4  2013     1     1     5 N804JB  B6      JetBlue Airways
##  5  2013     1     1     6 N668DN  DL      Delta Air Lines Inc.
##  6  2013     1     1     5 N39463  UA      United Air Lines Inc.
##  7  2013     1     1     6 N516JB  B6      JetBlue Airways
##  8  2013     1     1     6 N829AS  EV      ExpressJet Airlines Inc.
##  9  2013     1     1     6 N593JB  B6      JetBlue Airways
## 10  2013     1     1     6 N3ALAA  AA      American Airlines Inc.
## # ... with 336,766 more rows
```

Understanding Joins

```
x <- tribble(
  ~key, ~val_x,
  1, "x1",
  2, "x2",
  3, "x3"
)

y <- tribble(
  ~key, ~val_y,
  1, "y1",
  2, "y2",
```

```
  4, "y3"
)
```

Inner Join

```
# an inner join matches 2 tables wherever their keys are equal. The most important property of an inner

x %>%
  inner_join(y, by = "key")
```

```
## # A tibble: 2 x 3
##     key val_x val_y
##   <dbl> <chr> <chr>
## 1     1 x1    y1
## 2     2 x2    y2
```

Outer Joins

An outer join keeps observations that appear in at least one of the tables.

- A left join keeps all observations in x
- a right join keeps all observations in y
- a full join keeps all observations in x and y

Duplicate Keys

```
x <- tribble(
  ~key, ~val_x,
  1, "x1",
  2, "x2",
  2, "x3",
  1, "x4"
)

x
```

```
## # A tibble: 4 x 2
##     key val_x
##   <dbl> <chr>
## 1     1 x1
## 2     2 x2
## 3     2 x3
## 4     1 x4
```

```
y <- tribble(
  ~key, ~val_y,
  1, "y1",
  2, "y2"
)

y
```

```
## # A tibble: 2 x 2
##     key val_y
##   <dbl> <chr>
```

```
## 1     1 y1
## 2     2 y2
```

```
# join when one table has duplicate keys

left_join(x, y, by = "key")
```

```
## # A tibble: 4 x 3
##     key val_x val_y
##   <dbl> <chr> <chr>
## 1     1 x1    y1
## 2     2 x2    y2
## 3     2 x3    y2
## 4     1 x4    y1
```

```
# When both tables have duplicate keys, this results in all possible combinations (the cartesian produc

x2 <- tribble(
  ~key, ~val_x,
  1, "x1",
  2, "x2",
  2, "x3",
  3, "x4"
)


y2 <- tribble(
  ~key, ~val_y,
  1, "y1",
  2, "y2",
  2, "y3",
  3, "y4"
)

left_join(x, y, by = "key")
```

```
## # A tibble: 4 x 3
##     key val_x val_y
##   <dbl> <chr> <chr>
## 1     1 x1    y1
## 2     2 x2    y2
## 3     2 x3    y2
## 4     1 x4    y1
```

Defining the Key Columns

We can use other variables besides key to join.

By not defining b = “column”, it defaults to by = “NULL” and uses all the variables that occur in both tables.

```
# by = "NULL"

flights2 %>%
  left_join(weather)
```

```
## Joining, by = c("year", "month", "day", "hour", "origin")
```

```
## # A tibble: 336,776 x 18
##     year month   day  hour origin dest  tailnum carrier  temp  dewp humid
##    <dbl> <dbl> <int> <dbl> <chr>  <chr> <chr>   <chr>   <dbl> <dbl> <dbl>
##  1  2013     1     1     5 EWR    IAH   N14228  UA       NA    NA    NA
##  2  2013     1     1     5 LGA    IAH   N24211  UA       NA    NA    NA
##  3  2013     1     1     5 JFK    MIA   N619AA  AA       NA    NA    NA
##  4  2013     1     1     5 JFK    BQN   N804JB  B6       NA    NA    NA
##  5  2013     1     1     6 LGA    ATL   N668DN  DL       39.9  26.1  57.3
##  6  2013     1     1     5 EWR    ORD   N39463  UA       NA    NA    NA
##  7  2013     1     1     6 EWR    FLL   N516JB  B6       39.0  26.1  59.4
##  8  2013     1     1     6 LGA    IAD   N829AS  EV       39.9  26.1  57.3
##  9  2013     1     1     6 JFK    MCO   N593JB  B6       39.0  26.1  59.4
## 10  2013     1     1     6 LGA    ORD   N3ALAA  AA       39.9  26.1  57.3
## # ... with 336,766 more rows, and 7 more variables: wind_dir <dbl>,
## #   wind_speed <dbl>, wind_gust <dbl>, precip <dbl>, pressure <dbl>,
## #   visib <dbl>, time_hour <dttm>
```

```
# by = character vector

flights2 %>%
  left_join(planes, by = "tailnum")
```

```
## # A tibble: 336,776 x 16
##    year.x month   day  hour origin dest  tailnum carrier year.y type
##     <int> <int> <int> <dbl> <chr>  <chr> <chr>   <chr>    <int> <chr>
##  1   2013     1     1     5 EWR    IAH   N14228  UA        1999 Fixed win~
##  2   2013     1     1     5 LGA    IAH   N24211  UA        1998 Fixed win~
##  3   2013     1     1     5 JFK    MIA   N619AA  AA        1990 Fixed win~
##  4   2013     1     1     5 JFK    BQN   N804JB  B6        2012 Fixed win~
##  5   2013     1     1     6 LGA    ATL   N668DN  DL        1991 Fixed win~
##  6   2013     1     1     5 EWR    ORD   N39463  UA        2012 Fixed win~
##  7   2013     1     1     6 EWR    FLL   N516JB  B6        2000 Fixed win~
##  8   2013     1     1     6 LGA    IAD   N829AS  EV        1998 Fixed win~
##  9   2013     1     1     6 JFK    MCO   N593JB  B6        2004 Fixed win~
## 10   2013     1     1     6 LGA    ORD   N3ALAA  AA          NA <NA>
## # ... with 336,766 more rows, and 6 more variables: manufacturer <chr>,
## #   model <chr>, engines <int>, seats <int>, speed <int>, engine <chr>
```

```
# by a named character vector, by = c("a" = "b"). The column name from x will be taken by default

flights2 %>%
  left_join(airports, c("dest" = "faa"))
```

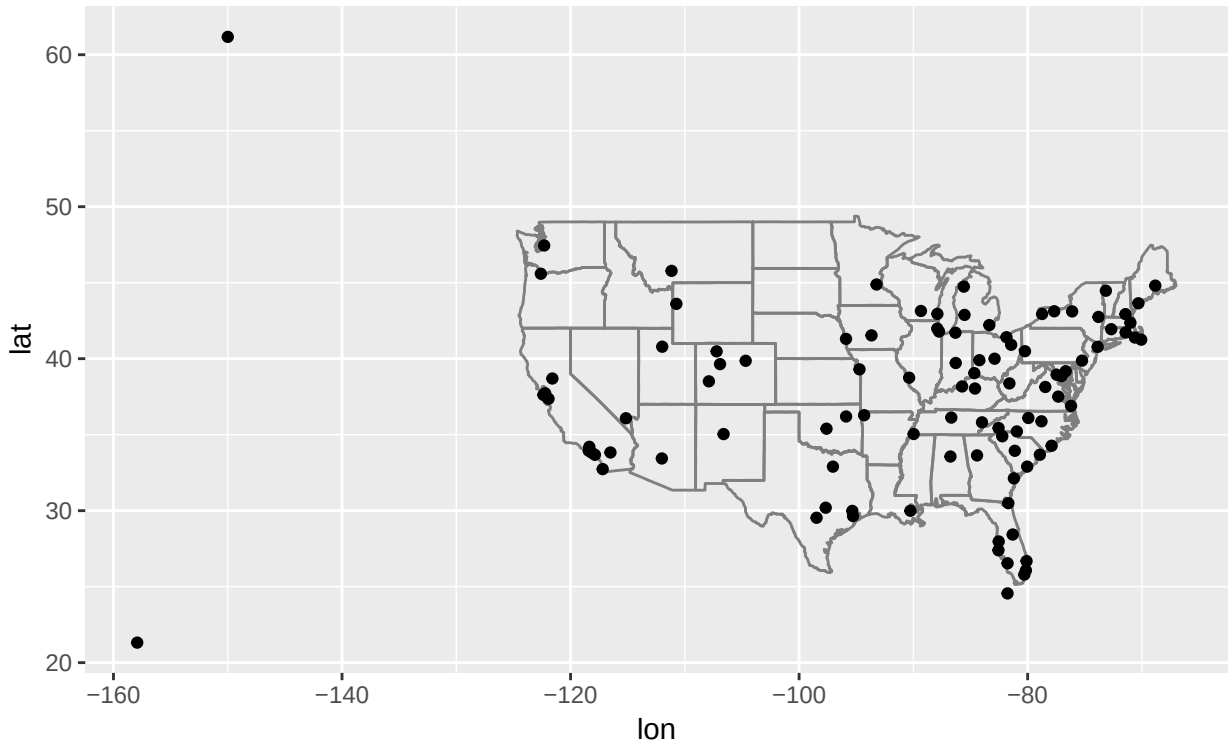
```
## # A tibble: 336,776 x 15
##     year month   day  hour origin dest  tailnum carrier name     lat   lon
##    <int> <int> <int> <dbl> <chr>  <chr> <chr>   <chr>   <chr>  <dbl> <dbl>
##  1  2013     1     1     5 EWR    IAH   N14228  UA      Georg~  30.0 -95.3
##  2  2013     1     1     5 LGA    IAH   N24211  UA      Georg~  30.0 -95.3
##  3  2013     1     1     5 JFK    MIA   N619AA  AA      Miami~  25.8 -80.3
##  4  2013     1     1     5 JFK    BQN   N804JB  B6      <NA>    NA    NA
##  5  2013     1     1     6 LGA    ATL   N668DN  DL      Harts~  33.6 -84.4
##  6  2013     1     1     5 EWR    ORD   N39463  UA      Chica~  42.0 -87.9
##  7  2013     1     1     6 EWR    FLL   N516JB  B6      Fort ~  26.1 -80.2
##  8  2013     1     1     6 LGA    IAD   N829AS  EV      Washi~  38.9 -77.5
```

```
##  9  2013     1     1     6 JFK    MCO   N593JB  B6      Orlan~  28.4 -81.3
## 10  2013     1     1     6 LGA    ORD   N3ALAA  AA      Chica~  42.0 -87.9
## # ... with 336,766 more rows, and 4 more variables: alt <int>, tz <dbl>,
## #   dst <chr>, tzone <chr>
```

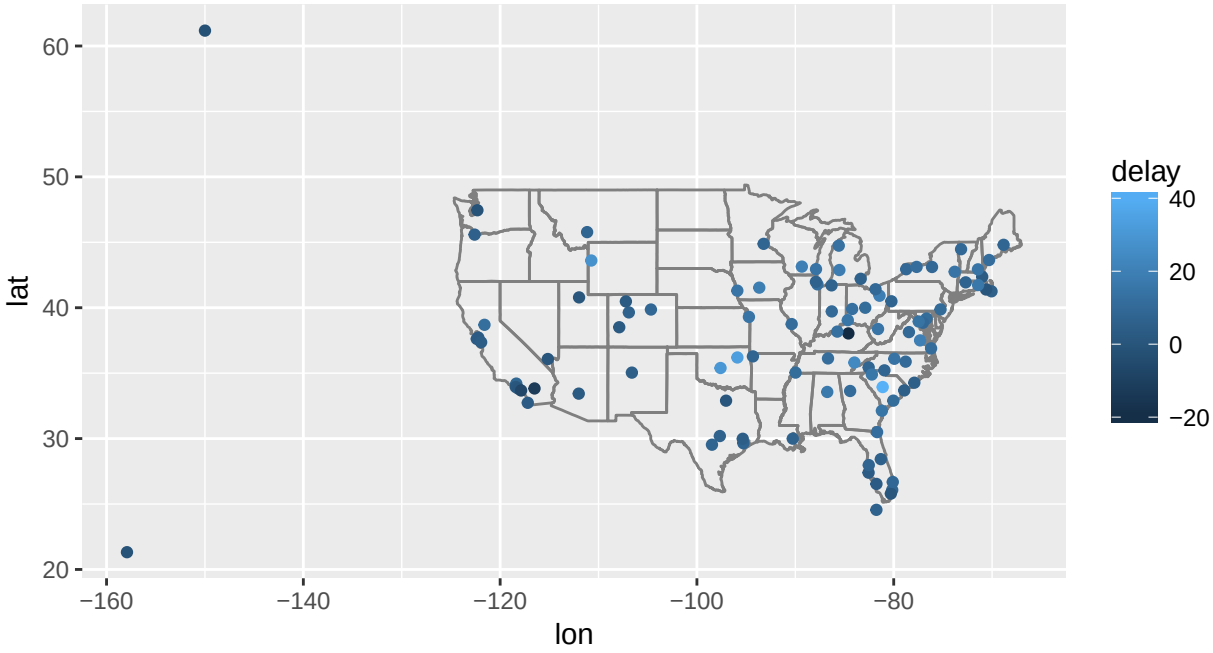
```
flights2 %>%
  left_join(airports, c("origin" = "faa"))
```

```
## # A tibble: 336,776 x 15
##     year month   day  hour origin dest  tailnum carrier name     lat   lon
##    <int> <int> <int> <dbl> <chr>  <chr> <chr>   <chr>   <chr>  <dbl> <dbl>
##  1  2013     1     1     5 EWR    IAH   N14228  UA      Newar~  40.7 -74.2
##  2  2013     1     1     5 LGA    IAH   N24211  UA      La Gu~  40.8 -73.9
##  3  2013     1     1     5 JFK    MIA   N619AA  AA      John ~  40.6 -73.8
##  4  2013     1     1     5 JFK    BQN   N804JB  B6      John ~  40.6 -73.8
##  5  2013     1     1     6 LGA    ATL   N668DN  DL      La Gu~  40.8 -73.9
##  6  2013     1     1     5 EWR    ORD   N39463  UA      Newar~  40.7 -74.2
##  7  2013     1     1     6 EWR    FLL   N516JB  B6      Newar~  40.7 -74.2
##  8  2013     1     1     6 LGA    IAD   N829AS  EV      La Gu~  40.8 -73.9
##  9  2013     1     1     6 JFK    MCO   N593JB  B6      John ~  40.6 -73.8
## 10  2013     1     1     6 LGA    ORD   N3ALAA  AA      La Gu~  40.8 -73.9
## # ... with 336,766 more rows, and 4 more variables: alt <int>, tz <dbl>,
## #   dst <chr>, tzone <chr>
```

```
# Compute the avg delay by destination, then join on the airports data frame so you can show the spatia

airports %>%
  semi_join(flights, c("faa" = "dest")) %>%
  ggplot(aes(lon, lat)) +
    borders("state") +
    geom_point() +
    coord_quickmap()
```




```
avg_dest_delays <- flights %>%
  group_by(dest) %>%
  summarize(delay = mean(arr_delay, na.rm = TRUE)) %>%
  inner_join(airports, by = c(dest = "faa"))

avg_dest_delays %>%
  ggplot(aes(lon, lat, color = delay)) +
    borders("state") +
    geom_point() +
    coord_quickmap()
```




```
# Add the location of the origin and destination (i.e. the lat and lon) to flights

flights %>%
  left_join(airports, by = c(dest = "faa")) %>%
  left_join(airports, by = c(origin = "faa")) %>%
  head()
```

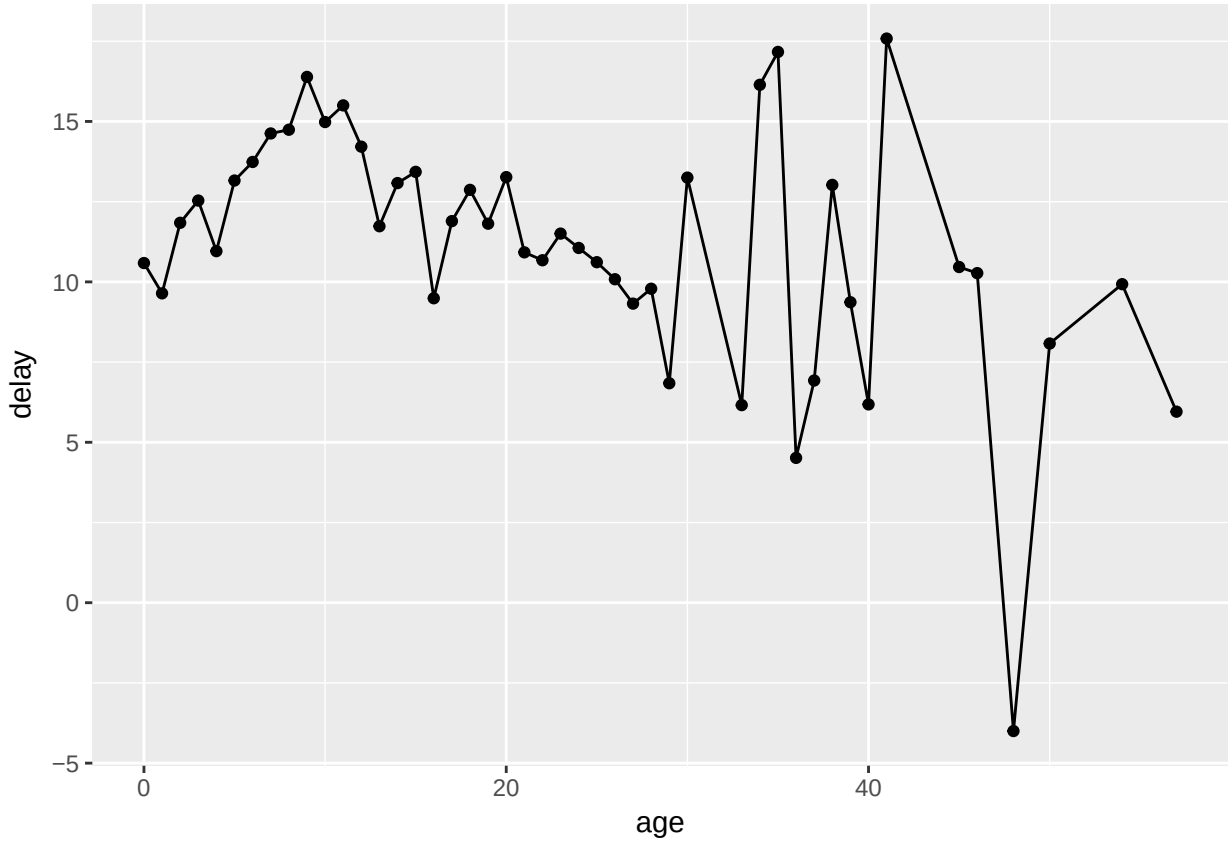
```
## # A tibble: 6 x 33
##    year month   day dep_time sched_dep_time dep_delay arr_time
##   <int> <int> <int>    <int>          <int>     <dbl>    <int>
## 1  2013     1     1      517            515         2      830
## 2  2013     1     1      533            529         4      850
## 3  2013     1     1      542            540         2      923
## 4  2013     1     1      544            545        -1     1004
## 5  2013     1     1      554            600        -6      812
## 6  2013     1     1      554            558        -4      740
## # ... with 26 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>, name.x <chr>, lat.x <dbl>, lon.x <dbl>, alt.x <int>,
## #   tz.x <dbl>, dst.x <chr>, tzone.x <chr>, name.y <chr>, lat.y <dbl>,
## #   lon.y <dbl>, alt.y <int>, tz.y <dbl>, dst.y <chr>, tzone.y <chr>
```

```
# Is there a relationship between the age of a plane and its delays?

plane_ages <- planes %>%
  mutate(age = 2013 - year) %>%
  select(tailnum, age)

flights %>%
  inner_join(plane_ages, by = "tailnum") %>%
  group_by(age) %>%
  filter(!is.na(dep_delay)) %>%
  summarize(delay = mean(dep_delay)) %>%
```

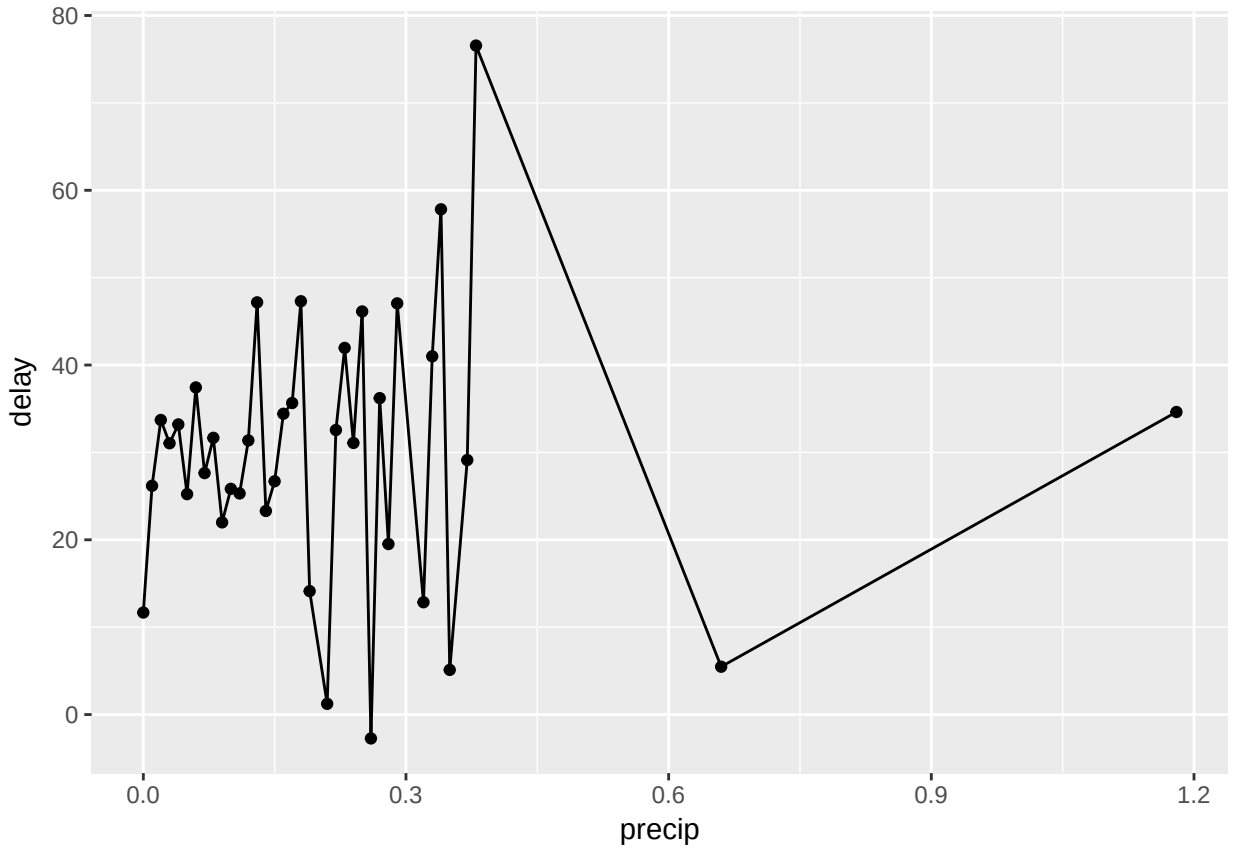
```r
  ggplot(aes(x = age, y = delay)) +
    geom_point() +
    geom_line()
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```

```
## Warning: Removed 1 rows containing missing values (geom_path).
```

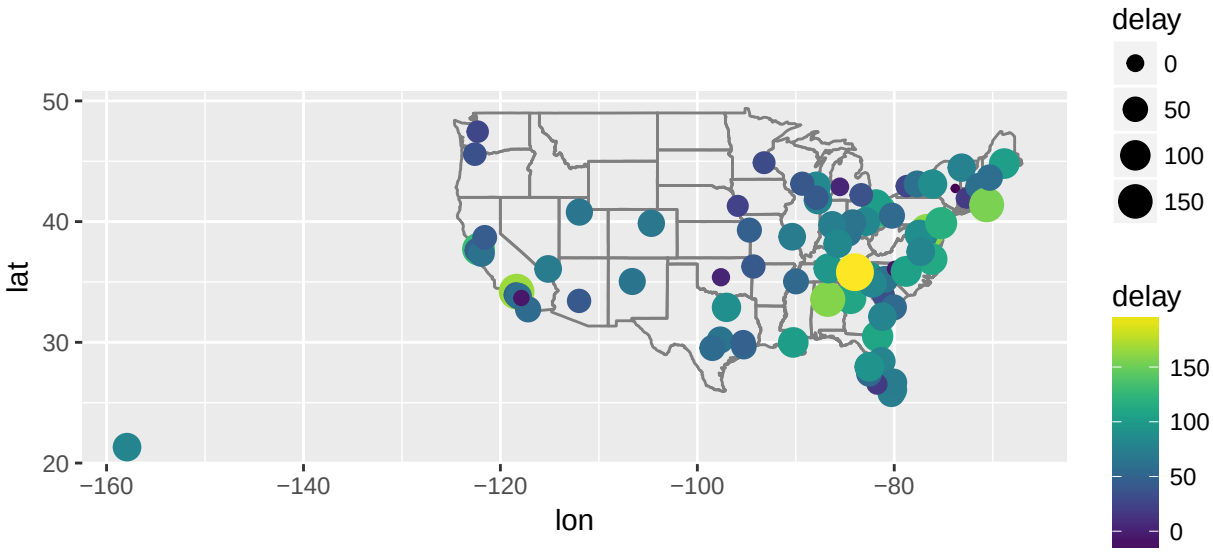



```r
# What weather conditions make it more likely to see a delay?

flight_weather <- flights %>%
  inner_join(weather, by = c("origin" = "origin",
                             "year" = "year",
                             "month" = "month",
                             "day" = "day",
                             "hour" = "hour"))

flight_weather %>%
  group_by(precip) %>%
  summarize(delay = mean(dep_delay, na.rm = TRUE)) %>%
  ggplot(aes(x = precip, y = delay)) +
    geom_line() +
    geom_point()
```




```
# What happened on June 13, 2013? Display the spatial pattern of delays, and then use google to cross r

# there was a large series of storms in the southeastern us

flights %>%
  filter(year == 2013, month == 6, day == 13) %>%
  group_by(dest) %>%
  summarize(delay = mean(arr_delay, na.rm = TRUE)) %>%
  inner_join(airports, by = c("dest" = "faa")) %>%
  ggplot(aes(y = lat, x = lon, size = delay, color = delay)) +
    borders("state") +
    geom_point() +
    coord_quickmap() +
    scale_color_viridis()
```

```
## Warning: Removed 3 rows containing missing values (geom_point).
```



Other Implementations

```
# base R merge can perform all 4 types of mutating join

# inner_join
merge(x, y)
```

```
##   key val_x val_y
## 1   1    x1    y1
## 2   1    x4    y1
## 3   2    x2    y2
## 4   2    x3    y2
```

```
# left_join
merge(x, y, all.x = TRUE)
```

```
##   key val_x val_y
## 1   1    x1    y1
## 2   1    x4    y1
## 3   2    x2    y2
## 4   2    x3    y2
```

```
# right_join
merge(x, y, all.y = TRUE)
```

```
##   key val_x val_y
## 1   1    x1    y1
## 2   1    x4    y1
## 3   2    x2    y2
## 4   2    x3    y2
```

```
#full_join
merge(x, y, all.x = TRUE, all.y = TRUE)
```

```
##   key val_x val_y
## 1   1    x1    y1
## 2   1    x4    y1
```

```
## 3   2    x2    y2
## 4   2    x3    y2
```

Filtering Joins

Filtering joins match observations in the same way as mutating joins, but affect the observations, not the variables.

semi_join(x, y) keeps all observations in x that have a match in y anti_join(x, y) drops all observations in x that have a match in y

```
# top 10 most popular destinations

top_dest <- flights %>%
  count(dest, sort = TRUE) %>%
  head(10)

top_dest
```

```
## # A tibble: 10 x 2
##    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
```

```
# can be done with filter, but inefficient for multiple variables

flights %>%
  filter(dest %in% top_dest$dest)
```

```
## # A tibble: 141,145 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>
##  1  2013     1     1      542            540         2      923
##  2  2013     1     1      554            600        -6      812
##  3  2013     1     1      554            558        -4      740
##  4  2013     1     1      555            600        -5      913
##  5  2013     1     1      557            600        -3      838
##  6  2013     1     1      558            600        -2      753
##  7  2013     1     1      558            600        -2      924
##  8  2013     1     1      558            600        -2      923
##  9  2013     1     1      559            559         0      702
## 10  2013     1     1      600            600         0      851
## # ... with 141,135 more rows, and 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>
```

```r
# semi-join connects 2 tables like a mutating join, but instead of adding new columns, it only keeps th

flights %>%
  semi_join(top_dest)
```

```
## Joining, by = "dest"
```

```
## # A tibble: 141,145 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>
##  1  2013     1     1      542            540         2      923
##  2  2013     1     1      554            600        -6      812
##  3  2013     1     1      554            558        -4      740
##  4  2013     1     1      555            600        -5      913
##  5  2013     1     1      557            600        -3      838
##  6  2013     1     1      558            600        -2      753
##  7  2013     1     1      558            600        -2      924
##  8  2013     1     1      558            600        -2      923
##  9  2013     1     1      559            559         0      702
## 10  2013     1     1      600            600         0      851
## # ... with 141,135 more rows, and 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>
```

```r
# only the existence of a match is important. It doesn't matter which observation is matched.

# the inverse of a semi_join is an anti_join. anti_join only keeps the rows which don't have a match. A

# flights without a match in planes

flights %>%
  anti_join(planes, by = "tailnum") %>%
  count(tailnum, sort = TRUE)
```

```
## # A tibble: 722 x 2
##    tailnum     n
##    <chr>   <int>
##  1 <NA>     2512
##  2 N725MQ    575
##  3 N722MQ    513
##  4 N723MQ    507
##  5 N713MQ    483
##  6 N735MQ    396
##  7 NOEGMQ    371
##  8 N534MQ    364
##  9 N542MQ    363
## 10 N531MQ    349
## # ... with 712 more rows
```

```r
# What does it mean for a flight to have a missing tailnum? What do the tail numbers that don't have a

# American Airlines and Envoy Airlines don't report tail numbers

flights %>%
  anti_join(planes, by = "tailnum") %>%
```

```
  count(carrier, sort = TRUE)
```

```
## # A tibble: 10 x 2
##    carrier     n
##    <chr>   <int>
##  1 MQ      25397
##  2 AA      22558
##  3 UA       1693
##  4 9E       1044
##  5 B6        830
##  6 US        699
##  7 FL        187
##  8 DL        110
##  9 F9         50
## 10 WN         38
```

```
# Filter flights to only show flights with planes that have flown at least 100 flights

planes_atleast100 <- flights %>%
  group_by(tailnum) %>%
  count() %>%
  filter(n > 100)

flights %>%
  semi_join(planes_atleast100, by = "tailnum")
```

```
## # A tibble: 229,202 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>
##  1  2013     1     1      517            515         2      830
##  2  2013     1     1      533            529         4      850
##  3  2013     1     1      544            545        -1     1004
##  4  2013     1     1      554            558        -4      740
##  5  2013     1     1      555            600        -5      913
##  6  2013     1     1      557            600        -3      709
##  7  2013     1     1      557            600        -3      838
##  8  2013     1     1      558            600        -2      849
##  9  2013     1     1      558            600        -2      853
## 10  2013     1     1      558            600        -2      923
## # ... with 229,192 more rows, and 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>
```

```
# Combine fueleconomy::vehicles and fueleconomy::common to find the only the records for the most commo

fueleconomy::vehicles %>%
  semi_join(fueleconomy::common, by = c("make", "model"))
```

```
## # A tibble: 14,531 x 12
##       id make  model  year class trans drive   cyl displ fuel    hwy   cty
##    <int> <chr> <chr> <int> <chr> <chr> <chr> <int> <dbl> <chr> <int> <int>
##  1  1833 Acura Inte~  1986 Subc~ Auto~ Fron~     4   1.6 Regu~    28    22
##  2  1834 Acura Inte~  1986 Subc~ Manu~ Fron~     4   1.6 Regu~    28    23
##  3  3037 Acura Inte~  1987 Subc~ Auto~ Fron~     4   1.6 Regu~    28    22
```

```
##  4  3038 Acura Inte~  1987 Subc~ Manu~ Fron~     4   1.6 Regu~    28    23
##  5  4183 Acura Inte~  1988 Subc~ Auto~ Fron~     4   1.6 Regu~    27    22
##  6  4184 Acura Inte~  1988 Subc~ Manu~ Fron~     4   1.6 Regu~    28    23
##  7  5303 Acura Inte~  1989 Subc~ Auto~ Fron~     4   1.6 Regu~    27    22
##  8  5304 Acura Inte~  1989 Subc~ Manu~ Fron~     4   1.6 Regu~    28    23
##  9  6442 Acura Inte~  1990 Subc~ Auto~ Fron~     4   1.8 Regu~    24    20
## 10  6443 Acura Inte~  1990 Subc~ Manu~ Fron~     4   1.8 Regu~    26    21
## # ... with 14,521 more rows
```

```
# What does anti_join(flights, airports, by = c("dest" = "faa")) tell you?

# These are flights which go to an airport that is not in FAA list of destinations - likely foreign airp

# What about anti_join(airports, flights, by = c("faa" = "dest"))?

# These are airports where there are no flights from NY in 2013
```

Join Problems

To avoid problems:

- Identify the variables that form the primary key in each table
- Check that none of the variables in the primary key are missing
- Check that foreign keys match primary keys in another table with an anti_join

Set Operations

intersect(x,y) - return only observations in both x and y union(x.y) - return unique observations in x and y setdiff(x,y) - return observations in x, but not in y

```
df1 <- tribble(
  ~x, ~y,
  1, 1,
  2, 1
)

df2 <- tribble(
  ~x, ~y,
  1, 1,
  1, 2
)

# intersect

intersect(df1, df2)
```

```
## # A tibble: 1 x 2
##       x     y
##   <dbl> <dbl>
## 1     1     1
```

```
# union - note that we get 3 rows, not 4

union(df1, df2)
```

```
## # A tibble: 3 x 2
##       x     y
##   <dbl> <dbl>
## 1     1     2
## 2     2     1
## 3     1     1
```

```
# setdiff

setdiff(df1, df2)
```

```
## # A tibble: 1 x 2
##       x     y
##   <dbl> <dbl>
## 1     2     1
```

```
# setdiff

setdiff(df2, df1)
```

```
## # A tibble: 1 x 2
##       x     y
##   <dbl> <dbl>
## 1     1     2
```