# Relational Data

Jarred Robidoux

2023-02-19

## Relational Data

### Introduction

It's rare that a data analysis involves only a single table of data. Typically you have many tables of data, and you must combine them to answer the questions that you're interested in. Collectively, multiple tables of data are called *relational data* because it is the relations, not just the individual datasets, that are important.

To work with relational data you need verbs that work with pairs of tables. There are three families of verbs designed to work with relational data:

- *Mutating Joins*, which add new variables to one data frame from matching observations in another.
- *Filtering Joins*, which filter observations from one data frame based on whether or not they match an observation in the other table.
- *Set operations*, which treat observations as if they were a set of elements.

#### Prerequisites

We will explore relational data from *nycflights13* using the two-table verbs from dplyr.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0     v purrr   1.0.1
## v tibble  3.1.8     v dplyr   1.1.0
## v tidyr   1.3.0     v stringr 1.5.0
## v readr   2.1.3     v forcats 1.0.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(nycflights13)
```

## nycflights13

We will use the nycflights13 package to learn about relational data. nycflights13 contains four tibbles that are related to the *flights* table that you used in *data transformation:*

*airlines* lets you look up the full carrier name from its abbreviated code

```
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* gives information about each airport, identified by the *faa* airport code:

```
airports
```

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

*planes* gives information about each plane, identified by its *tailnum*:

```
planes
```

```
## # A tibble: 3,322 x 9
##    tailnum  year type                  manuf~1 model engines seats speed engine
##    <chr>   <int> <chr>                 <chr>   <chr>   <int> <int> <int> <chr>
##  1 N10156   2004 Fixed wing multi engi~ EMBRAER EMB-~       2    55    NA Turbo~
##  2 N102UW   1998 Fixed wing multi engi~ AIRBUS~ A320~       2   182    NA Turbo~
##  3 N103US   1999 Fixed wing multi engi~ AIRBUS~ A320~       2   182    NA Turbo~
##  4 N104UW   1999 Fixed wing multi engi~ AIRBUS~ A320~       2   182    NA Turbo~
##  5 N10575   2002 Fixed wing multi engi~ EMBRAER EMB-~       2    55    NA Turbo~
##  6 N105UW   1999 Fixed wing multi engi~ AIRBUS~ A320~       2   182    NA Turbo~
##  7 N107US   1999 Fixed wing multi engi~ AIRBUS~ A320~       2   182    NA Turbo~
##  8 N108UW   1999 Fixed wing multi engi~ AIRBUS~ A320~       2   182    NA Turbo~
##  9 N109UW   1999 Fixed wing multi engi~ AIRBUS~ A320~       2   182    NA Turbo~
## 10 N110UW   1999 Fixed wing multi engi~ AIRBUS~ A320~       2   182    NA Turbo~
## # ... with 3,312 more rows, and abbreviated variable name 1: manufacturer
```

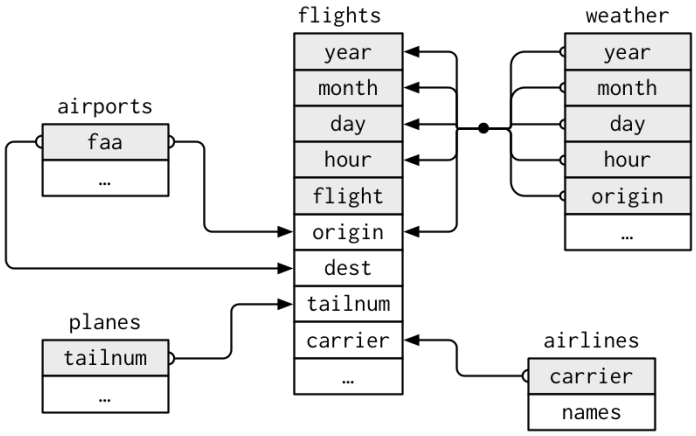
*weather* gives the weather at each NYC airport for each hour:

```
weather
```

```
## # A tibble: 26,115 x 15
##    origin  year month   day  hour  temp  dewp humid wind_dir wind_speed wind_g~1
##    <chr>  <int> <int> <int> <int> <dbl> <dbl> <dbl>    <dbl>      <dbl>    <dbl>
##  1 EWR     2013     1     1     1  39.0  26.1  59.4      270      10.4        NA
##  2 EWR     2013     1     1     2  39.0  27.0  61.6      250       8.06       NA
##  3 EWR     2013     1     1     3  39.0  28.0  64.4      240      11.5        NA
##  4 EWR     2013     1     1     4  39.9  28.0  62.2      250      12.7        NA
##  5 EWR     2013     1     1     5  39.0  28.0  64.4      260      12.7        NA
##  6 EWR     2013     1     1     6  37.9  28.0  67.2      240      11.5        NA
##  7 EWR     2013     1     1     7  39.0  28.0  64.4      240      15.0        NA
##  8 EWR     2013     1     1     8  39.9  28.0  62.2      250      10.4        NA
##  9 EWR     2013     1     1     9  39.9  28.0  62.2      260      15.0        NA
## 10 EWR     2013     1     1    10  41    28.0  59.6      260      13.8        NA
## # ... with 26,105 more rows, 4 more variables: precip <dbl>, pressure <dbl>,
## #   visib <dbl>, time_hour <dttm>, and abbreviated variable name 1: wind_gust
```

One way to show the relationships between the different tables is with a drawing:

```
knitr::include_graphics("relational-nycflights.png")
```




For nycflights13:

- *flights* connects *planes* via a single variable, *tailnum*
- *flights* connects to *airlines* through the *carrier* variable
- *flights* connects to *airports* in two ways: via the *origin* and *dest* variables
- *flights* connects to *weather* via *origin* (the location), and *year*, *month*, *day* and *hour* (the time)

## Keys

The variables used to connect each pair of tables are called *keys*. A key is a variable (or set of variables) that uniquely identifies an observation.

There are two types of keys:

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

A **foreign key** uniquely identifies 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 and the corresponding foreign key in another table form a **relation**. Relations are typically one-to-many. For example, each flight has one plance, but each plane has many flights. In other data, you'll occasionally see a 1-to-1 relationship. You can think of this as a special case of 1-to-many. You can model many-to-many relations with many-to-1 plus a 1-to-many relation. For example, in this data there's a many-to-many relationship between airlines and airports: each airline flies to many airports; each airport hosts many airlines.

## Mutating Joins

The first tool that we'll look at for combining a pair of tables is the **mutating join**. 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.

- **Left Join** Keeps all observations in X
- **Right Join** Keeps all observations in Y
- **Full Join** Keeps all observations in X and Y
- **Inner Join** An inner join matches pairs of observations whenever keys are equal. Unmatched rows are NOT included in the result.