Cheatsheet for dplyr join functions

Jenny Bryan

Why the cheatsheet

Examples for those of us who don't speak SQL so good. There are lots of Venn diagrams re: SQL joins on the internet, but I wanted R examples. Those diagrams also utterly fail to show what's really going on vis-a-vis rows AND columns.

Other great places to read about joins:

- The dplyr vignette on Two-table verbs
- The Relational data chapter in R for Data Science. Excellent diagrams.

The data

Working with two small data frames, `superheroes` and `publishers`.

```
library(tidyverse) ## dplyr provides the join functions

superheroes <- "
    name, alignment, gender,         publisher
 Magneto,       bad,   male,            Marvel
   Storm,      good, female,            Marvel
Mystique,       bad, female,            Marvel
  Batman,      good,   male,                DC
   Joker,       bad,   male,                DC
Catwoman,       bad, female,                DC
 Hellboy,      good,   male, Dark Horse Comics
"
superheroes <- read_csv(superheroes, skip = 1)

publishers <- "
  publisher, yr_founded
         DC,       1934
     Marvel,       1939
      Image,       1992
"
publishers <- read_csv(publishers, skip = 1)
```

Sorry, cheat sheet does not illustrate "multiple match" situations terribly well.

Sub-plot: watch the row and variable order of the join results for a healthy reminder of why it's dangerous to rely on any of that in an analysis.

inner_join(superheroes, publishers)

> inner_join(x, y): Return all rows from x where there are matching values in y, and all columns from x and y. If there are multiple matches between x and y, all combination of the matches are returned. This is a mutating join.

```
(ijsp <- inner_join(superheroes, publishers))
#> Joining, by = "publisher"
#> # A tibble: 6 x 5
#>   name     alignment gender publisher yr_founded
#>   <chr>    <chr>     <chr>  <chr>          <dbl>
#> 1 Magneto  bad       male   Marvel          1939
#> 2 Storm    good      female Marvel          1939
#> 3 Mystique bad       female Marvel          1939
#> 4 Batman   good      male   DC              1934
#> 5 Joker    bad       male   DC              1934
#> 6 Catwoman bad       female DC              1934
```

We lose Hellboy in the join because, although he appears in `x = superheroes`, his publisher Dark Horse Comics does not appear in `y = publishers`. The join result has all variables from `x = superheroes` plus `yr_founded`, from `y`.

superheroes

name	alignment	gender	publisher
Magneto	bad	male	Marvel
Storm	good	female	Marvel
Mystique	bad	female	Marvel
Batman	good	male	DC
Joker	bad	male	DC
Catwoman	bad	female	DC
Hellboy	good	male	Dark Horse Comics

publishers

publisher	yr_founded
DC	1934
Marvel	1939
Image	1992

inner_join(x = superheroes, y = publishers)

name	alignment	gender	publisher	yr_founded
Magneto	bad	male	Marvel	1939
Storm	good	female	Marvel	1939
Mystique	bad	female	Marvel	1939
Batman	good	male	DC	1934
Joker	bad	male	DC	1934
Catwoman	bad	female	DC	1934

semi_join(superheroes, publishers)

semi_join(x, y): Return all rows from x where there are matching values in y, keeping just columns from x. A semi join differs from an inner join because an inner join will return one

row of x for each matching row of y, where a semi join will never duplicate rows of x. This is a filtering join.

```
(sjsp <- semi_join(superheroes, publishers))
#> Joining, by = "publisher"
#> # A tibble: 6 x 4
#>   name     alignment gender publisher
#>   <chr>    <chr>     <chr>  <chr>
#> 1 Magneto  bad       male   Marvel
#> 2 Storm    good      female Marvel
#> 3 Mystique bad       female Marvel
#> 4 Batman   good      male   DC
#> 5 Joker    bad       male   DC
#> 6 Catwoman bad       female DC
```

We get a similar result as with `inner_join()` but the join result contains only the variables originally found in `x = superheroes`. But note the row order has changed.

superheroes

name	alignment	gender	publisher
Magneto	bad	male	Marvel
Storm	good	female	Marvel
Mystique	bad	female	Marvel
Batman	good	male	DC
Joker	bad	male	DC
Catwoman	bad	female	DC
Hellboy	good	male	Dark Horse Comics

publishers

publisher	yr_founded
DC	1934
Marvel	1939
Image	1992

semi-join(x = superheroes, y = publishers)

name	alignment	gender	publisher
Magneto	bad	male	Marvel
Storm	good	female	Marvel
Mystique	bad	female	Marvel
Batman	good	male	DC
Joker	bad	male	DC
Catwoman	bad	female	DC

left_join(superheroes, publishers)

left_join(x, y): Return all rows from x, and all columns from x and y. If there are multiple matches between x and y, all combination of the matches are returned. This is a mutating join.

```
(ljsp <- left_join(superheroes, publishers))
#> Joining, by = "publisher"
#> # A tibble: 7 x 5
#>   name     alignment gender publisher         yr_founded
#>   <chr>    <chr>     <chr>  <chr>                  <dbl>
#> 1 Magneto  bad       male   Marvel                  1939
#> 2 Storm    good      female Marvel                  1939
#> 3 Mystique bad       female Marvel                  1939
#> 4 Batman   good      male   DC                      1934
#> 5 Joker    bad       male   DC                      1934
#> 6 Catwoman bad       female DC                      1934
#> 7 Hellboy  good      male   Dark Horse Comics         NA
```

We basically get `x = superheroes` back, but with the addition of variable `yr_founded`, which is unique to `y = publishers`. Hellboy, whose publisher does not appear in `y = publishers`, has an `NA` for `yr_founded`.

superheroes

name	alignment	gender	publisher
Magneto	bad	male	Marvel
Storm	good	female	Marvel
Mystique	bad	female	Marvel
Batman	good	male	DC
Joker	bad	male	DC
Catwoman	bad	female	DC
Hellboy	good	male	Dark Horse Comics

publishers

publisher	yr_founded
DC	1934
Marvel	1939
Image	1992

left_join(x = superheroes, y = publishers)

name	alignment	gender	publisher	yr_founded
Magneto	bad	male	Marvel	1939
Storm	good	female	Marvel	1939
Mystique	bad	female	Marvel	1939
Batman	good	male	DC	1934
Joker	bad	male	DC	1934
Catwoman	bad	female	DC	1934
Hellboy	good	male	Dark Horse Comics	NA

anti_join(superheroes, publishers)

anti_join(x, y): Return all rows from x where there are not matching values in y, keeping just columns from x. This is a filtering join.

```
(ajsp <- anti_join(superheroes, publishers))
#> Joining, by = "publisher"
#> # A tibble: 1 x 4
#>   name    alignment gender publisher
#>   <chr>   <chr>     <chr>  <chr>
#> 1 Hellboy good      male   Dark Horse Comics
```

We keep **only** Hellboy now (and do not get `yr_founded`).

superheroes

name	alignment	gender	publisher
Magneto	bad	male	Marvel
Storm	good	female	Marvel
Mystique	bad	female	Marvel
Batman	good	male	DC
Joker	bad	male	DC
Catwoman	bad	female	DC
Hellboy	good	male	Dark Horse Comics

publishers

publisher	yr_founded
DC	1934
Marvel	1939
Image	1992

anti_join(x = superheroes, y = publishers)

name	alignment	gender	publisher
Hellboy	good	male	Dark Horse Comics

inner_join(publishers, superheroes)

inner_join(x, y): Return all rows from x where there are matching values in y, and all columns from x and y. If there are multiple matches between x and y, all combination of the matches are returned. This is a mutating join.

```
(ijps <- inner_join(publishers, superheroes))
#> Joining, by = "publisher"
#> # A tibble: 6 x 5
#>   publisher yr_founded name     alignment gender
#>   <chr>          <dbl> <chr>    <chr>     <chr>
```

```
#> 1 DC             1934 Batman   good      male
#> 2 DC             1934 Joker    bad       male
#> 3 DC             1934 Catwoman bad       female
#> 4 Marvel         1939 Magneto  bad       male
#> 5 Marvel         1939 Storm    good      female
#> 6 Marvel         1939 Mystique bad       female
```

In a way, this does illustrate multiple matches, if you think about it from the `x = publishers` direction. Every publisher that has a match in `y = superheroes` appears multiple times in the result, once for each match. In fact, we're getting the same result as with `inner_join(superheroes, publishers)`, up to variable order (which you should also never rely on in an analysis).

publishers

publisher	yr_founded
DC	1934
Marvel	1939
Image	1992

superheroes

name	alignment	gender	publisher
Magneto	bad	male	Marvel
Storm	good	female	Marvel
Mystique	bad	female	Marvel
Batman	good	male	DC
Joker	bad	male	DC
Catwoman	bad	female	DC
Hellboy	good	male	Dark Horse Comics

inner_join(x = publishers, y = superheroes)

publisher	yr_founded	name	alignment	gender
DC	1934	Batman	good	male
DC	1934	Joker	bad	male
DC	1934	Catwoman	bad	female
Marvel	1939	Magneto	bad	male
Marvel	1939	Storm	good	female
Marvel	1939	Mystique	bad	female

semi_join(publishers, superheroes)

semi_join(x, y): Return all rows from x where there are matching values in y, keeping just columns from x. A semi join differs from an inner join because an inner join will return one row of x for each matching row of y, where a semi join will never duplicate rows of x. This is a filtering join.

```
(sjps <- semi_join(x = publishers, y = superheroes))
#> Joining, by = "publisher"
#> # A tibble: 2 x 2
#>   publisher yr_founded
#>   <chr>          <dbl>
#> 1 DC              1934
#> 2 Marvel          1939
```

Now the effects of switching the `x` and `y` roles is more clear. The result resembles `x = publishers`, but the publisher Image is lost, because there are no observations where `publisher == "Image"` in `y = superheroes`.

publishers

publisher	yr_founded
DC	1934
Marvel	1939
Image	1992

superheroes

name	alignment	gender	publisher
Magneto	bad	male	Marvel
Storm	good	female	Marvel
Mystique	bad	female	Marvel
Batman	good	male	DC
Joker	bad	male	DC
Catwoman	bad	female	DC
Hellboy	good	male	Dark Horse Comics

semi-join(x = publishers, y = superheroes)

publisher	yr_founded
DC	1934
Marvel	1939

left_join(publishers, superheroes)

> left_join(x, y): Return all rows from x, and all columns from x and y. If there are multiple matches between x and y, all combination of the matches are returned. This is a mutating join.

```
(ljps <- left_join(publishers, superheroes))
#> Joining, by = "publisher"
#> # A tibble: 7 x 5
#>   publisher yr_founded name     alignment gender
#>   <chr>          <dbl> <chr>    <chr>     <chr>
#> 1 DC              1934 Batman   good      male
#> 2 DC              1934 Joker    bad       male
```

```
#> 3 DC             1934 Catwoman bad       female
#> 4 Marvel         1939 Magneto  bad       male
#> 5 Marvel         1939 Storm    good      female
#> 6 Marvel         1939 Mystique bad       female
#> 7 Image          1992 <NA>     <NA>      <NA>
```

We get a similar result as with `inner_join()` but the publisher Image survives in the join, even though no superheroes from Image appear in `y = superheroes`. As a result, Image has `NA`s for `name`, `alignment`, and `gender`.

publishers

publisher	yr_founded
DC	1934
Marvel	1939
Image	1992

superheroes

name	alignment	gender	publisher
Magneto	bad	male	Marvel
Storm	good	female	Marvel
Mystique	bad	female	Marvel
Batman	good	male	DC
Joker	bad	male	DC
Catwoman	bad	female	DC
Hellboy	good	male	Dark Horse Comics

left_join(x = publishers, y = superheroes)

publisher	yr_founded	name	alignment	gender
DC	1934	Batman	good	male
DC	1934	Joker	bad	male
DC	1934	Catwoman	bad	female
Marvel	1939	Magneto	bad	male
Marvel	1939	Storm	good	female
Marvel	1939	Mystique	bad	female
Image	1992	NA	NA	NA

anti_join(publishers, superheroes)

anti_join(x, y): Return all rows from x where there are not matching values in y, keeping just columns from x. This is a filtering join.

```
(ajps <- anti_join(publishers, superheroes))
#> Joining, by = "publisher"
#> # A tibble: 1 x 2
#>   publisher yr_founded
```

```
#>   <chr>          <dbl>
#> 1 Image           1992
```

We keep **only** publisher Image now (and the variables found in `x = publishers`).

publishers

publisher	yr_founded
DC	1934
Marvel	1939
Image	1992

superheroes

name	alignment	gender	publisher
Magneto	bad	male	Marvel
Storm	good	female	Marvel
Mystique	bad	female	Marvel
Batman	good	male	DC
Joker	bad	male	DC
Catwoman	bad	female	DC
Hellboy	good	male	Dark Horse Comics

anti_join(x = publishers, y = superheroes)

publisher	yr_founded
Image	1992

full_join(superheroes, publishers)

full_join(x, y): Return all rows and all columns from both x and y. Where there are not matching values, returns NA for the one missing. This is a mutating join.

```
(fjsp <- full_join(superheroes, publishers))
#> Joining, by = "publisher"
#> # A tibble: 8 x 5
#>   name     alignment gender publisher         yr_founded
#>   <chr>    <chr>     <chr>  <chr>                  <dbl>
#> 1 Magneto  bad       male   Marvel                  1939
#> 2 Storm    good      female Marvel                  1939
#> 3 Mystique bad       female Marvel                  1939
#> 4 Batman   good      male   DC                      1934
#> 5 Joker    bad       male   DC                      1934
#> 6 Catwoman bad       female DC                      1934
#> 7 Hellboy  good      male   Dark Horse Comics         NA
#> 8 <NA>     <NA>      <NA>   Image                   1992
```

We get all rows of `x = superheroes` plus a new row from `y = publishers`, containing the publisher Image.

We get all variables from `x = superheroes` AND all variables from `y = publishers`. Any row that derives solely from one table or the other carries `NA`s in the variables found only in the other table.

superheroes

name	alignment	gender	publisher
Magneto	bad	male	Marvel
Storm	good	female	Marvel
Mystique	bad	female	Marvel
Batman	good	male	DC
Joker	bad	male	DC
Catwoman	bad	female	DC
Hellboy	good	male	Dark Horse Comics

publishers

publisher	yr_founded
DC	1934
Marvel	1939
Image	1992

full_join(x = superheroes, y = publishers)

name	alignment	gender	publisher	yr_founded
Magneto	bad	male	Marvel	1939
Storm	good	female	Marvel	1939
Mystique	bad	female	Marvel	1939
Batman	good	male	DC	1934
Joker	bad	male	DC	1934
Catwoman	bad	female	DC	1934
Hellboy	good	male	Dark Horse Comics	NA
NA	NA	NA	Image	1992