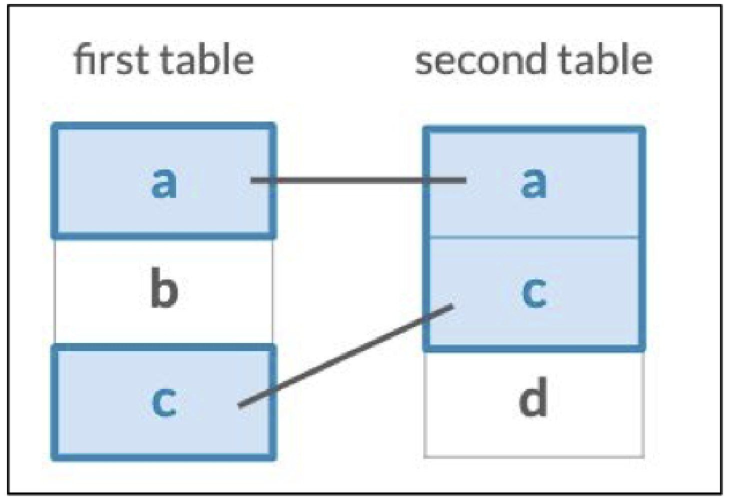
Stack Overflow questions

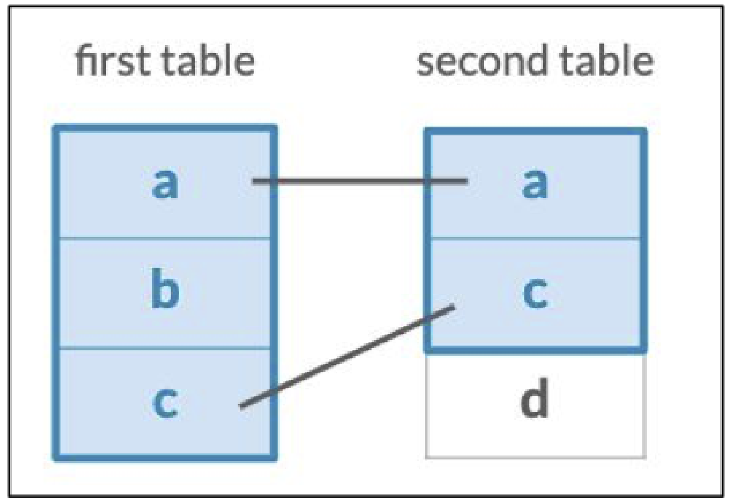
JOINING DATA WITH DPLYR

Chris Cardillo
Data Scientist at DataCamp

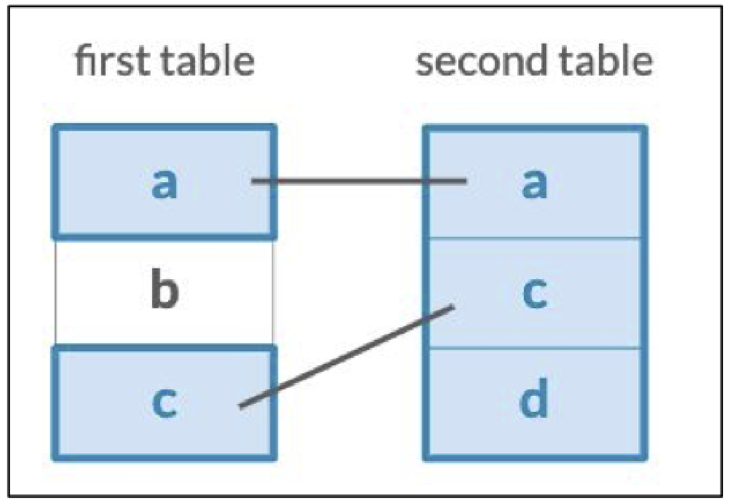
The joining verbs



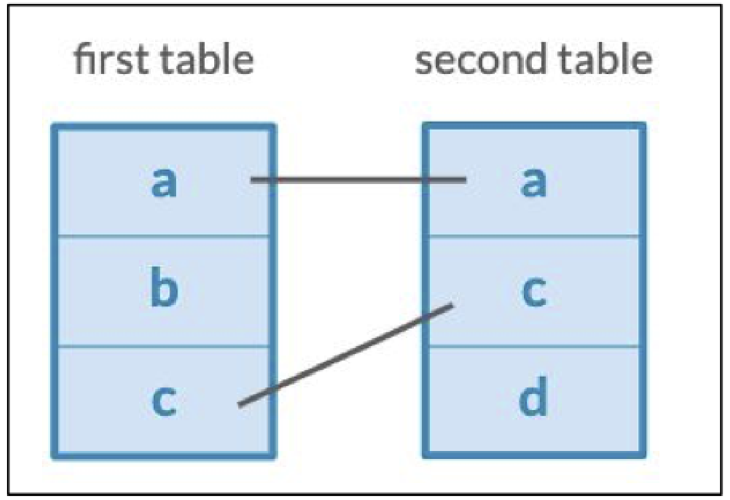
Inner join



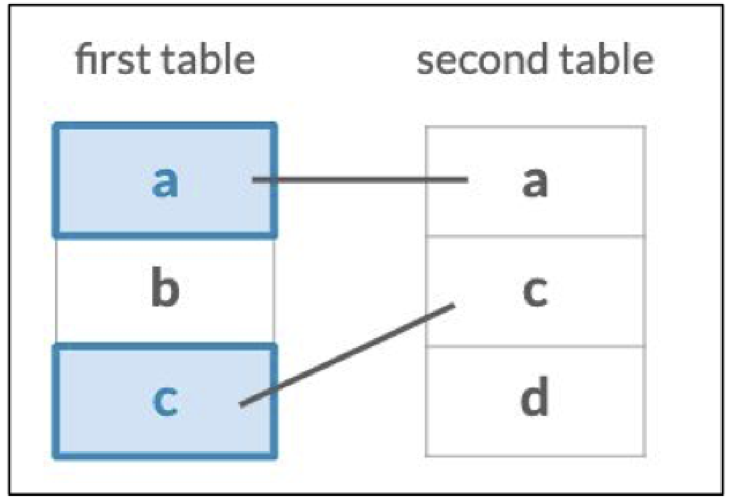
Left join



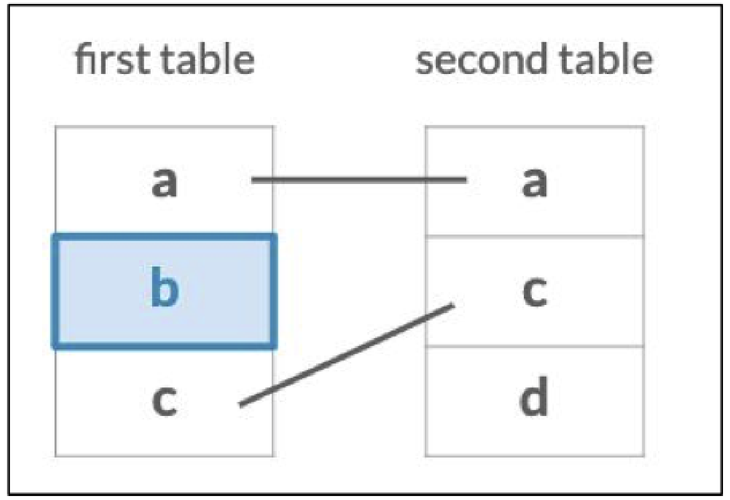
Right join



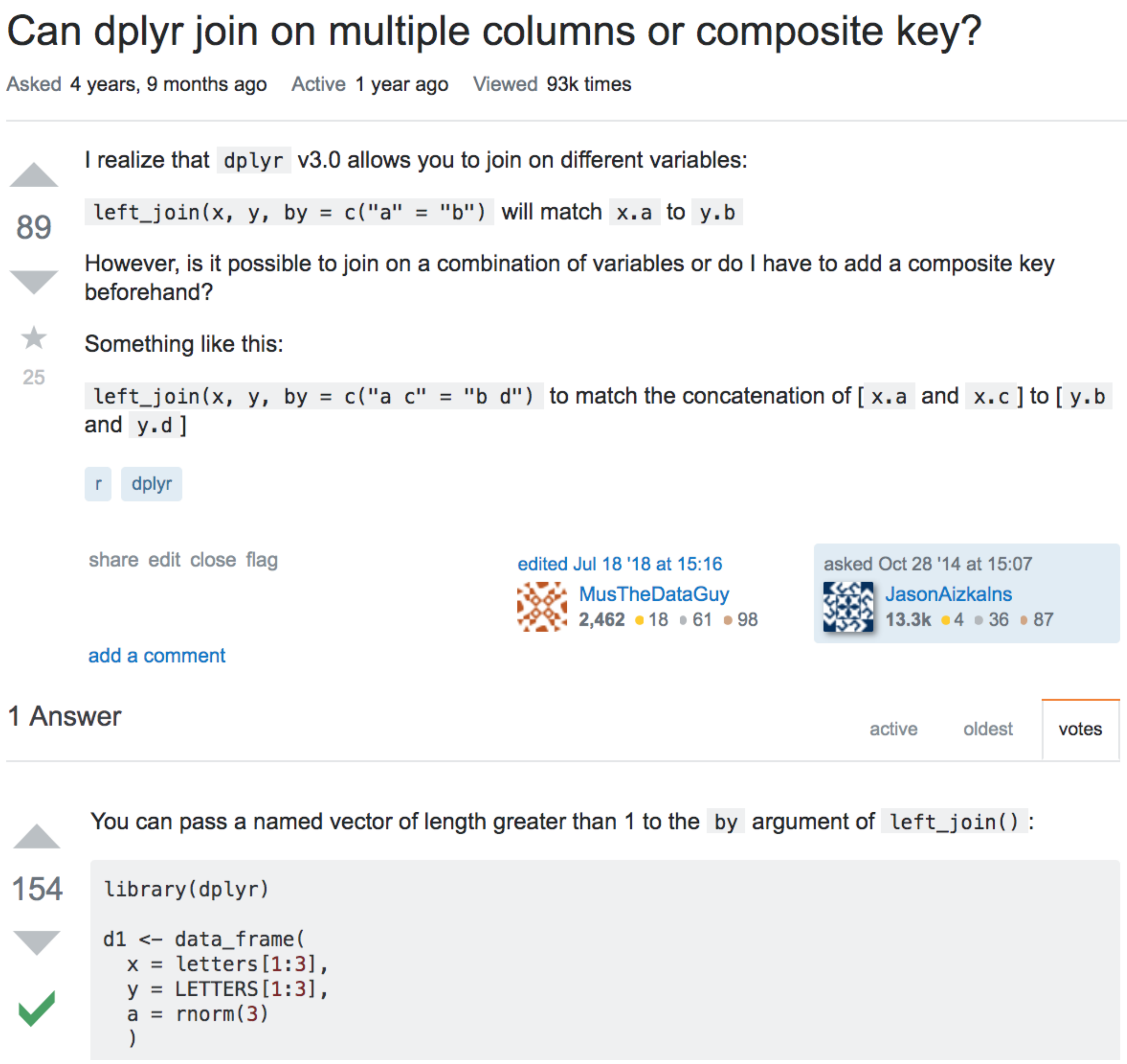
Full join



Semi join



Anti join



Can dplyr join on multiple columns or composite key?

Asked 4 years, 9 months ago Active 1 year ago Viewed 93k times

89

25

I realize that `dplyr` v3.0 allows you to join on different variables:

`left_join(x, y, by = c("a" = "b"))` will match `x.a` to `y.b`

However, is it possible to join on a combination of variables or do I have to add a composite key beforehand?

Something like this:

`left_join(x, y, by = c("a c" = "b d"))` to match the concatenation of [`x.a` and `x.c`] to [`y.b` and `y.d`]

r dplyr

share edit close flag

edited Jul 18 '18 at 15:16
MusTheDataGuy
2,462 ●18 ●61 ●98

asked Oct 28 '14 at 15:07
JasonAizkalns
13.3k ●4 ●36 ●87

add a comment

1 Answer

active oldest votes

154

✓

You can pass a named vector of length greater than 1 to the `by` argument of `left_join()`:

```
library(dplyr)

d1 <- data_frame(
  x = letters[1:3],
  y = LETTERS[1:3],
  a = rnorm(3)
  )
```

The questions table

```
questions
```

```
# A tibble: 294,735 x 3
         id creation_date score
      <int> <date>        <int>
 1 22557677 2014-03-21        1
 2 22557707 2014-03-21        2
 3 22558084 2014-03-21        2
 4 22558395 2014-03-21        2
 5 22558613 2014-03-21        0
 6 22558677 2014-03-21        2
 7 22558887 2014-03-21        8
 8 22559180 2014-03-21        1
 9 22559312 2014-03-21        0
10 22559322 2014-03-21        2
# … with 294,725 more rows
```

The question_tags and tags tables

```
question_tags
```

```
# A tibble: 497,153 x 2
   question_id tag_id
         <int>  <int>
 1    22557677     18
 2    22557677    139
 3    22557677  16088
 4    22557677   1672
 5    22558084   6419
 6    22558084  92764
 7    22558395   5569
 8    22558395    134
 9    22558395   9412
10    22558395  18621
# … with 497,143 more rows
```

```
tags
```

```
# A tibble: 48,299 x 2
       id tag_name
    <dbl> <chr>
 1 124399 laravel-dusk
 2 124402 spring-cloud-vault-config
 3 124404 spring-vault
 4 124405 apache-bahir
 5 124407 astc
 6 124408 simulacrum
 7 124410 angulartics2
 8 124411 django-rest-viewsets
 9 124414 react-native-lightbox
10 124417 java-module
# … with 48,289 more rows
```

Joining question_tags with questions

```r
questions %>%
  inner_join(question_tags, by = c("id" = "question_id"))
```

Joining tags

```
questions_with_tags <- questions %>%
  inner_join(question_tags, by = c("id" = "question_id")) %>%
  inner_join(tags, by = c("tag_id" = "id"))

questions_with_tags
```

```
# A tibble: 497,153 x 5
         id creation_date score tag_id tag_name
      <int> <date>        <int>  <dbl> <chr>
 1 22557677 2014-03-21        1     18 regex
 2 22557677 2014-03-21        1    139 string
 3 22557677 2014-03-21        1  16088 time-complexity
 4 22557677 2014-03-21        1   1672 backreference
 5 22558084 2014-03-21        2   6419 time-series
 6 22558084 2014-03-21        2  92764 panel-data
 7 22558395 2014-03-21        2   5569 function
 8 22558395 2014-03-21        2    134 sorting
 9 22558395 2014-03-21        2   9412 vectorization
10 22558395 2014-03-21        2  18621 operator-precedence
# … with 497,143 more rows
```

Most common tags

```r
questions_with_tags %>%
  count(tag_name, sort = TRUE)
```

```
# A tibble: 7,840 x 2
   tag_name        n
   <chr>       <int>
 1 ggplot2     28228
 2 dataframe   18874
 3 shiny       14219
 4 dplyr       14039
 5 plot        11315
 6 data.table   8809
 7 matrix       6205
 8 loops        5149
 9 regex        4912
10 function     4892
# … with 7,830 more rows
```

Joining questions and answers

JOINING DATA WITH DPLYR

Chris Cardillo
Data Scientist at DataCamp

The answers table

```
answers
```

```
# A tibble: 380,643 x 4
         id creation_date question_id score
      <int> <date>              <int> <int>
 1 39143713 2016-08-25       39143518     3
 2 39143869 2016-08-25       39143518     1
 3 39143935 2016-08-25       39142481     0
 4 39144014 2016-08-25       39024390     0
 5 39144252 2016-08-25       39096741     6
 6 39144375 2016-08-25       39143885     5
 7 39144430 2016-08-25       39144077     0
 8 39144625 2016-08-25       39142728     1
 9 39144794 2016-08-25       39043648     0
10 39145033 2016-08-25       39133170     1
# … with 380,633 more rows
```

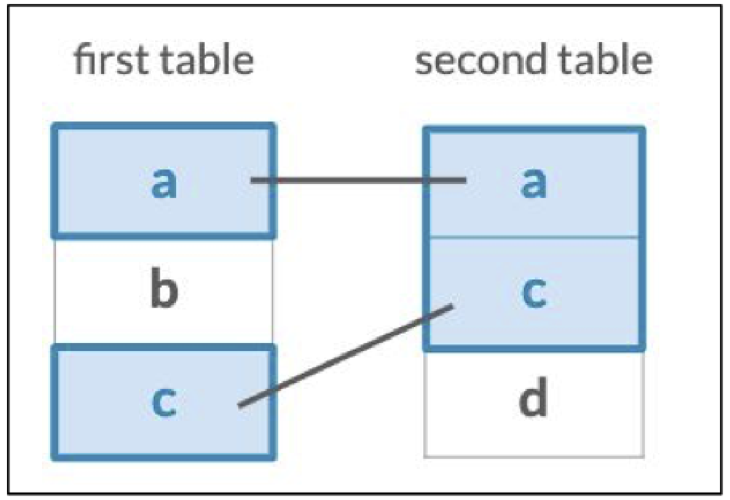
The question ID

```r
questions %>%
  inner_join(answers, by = c("id" = "question_id"))
```

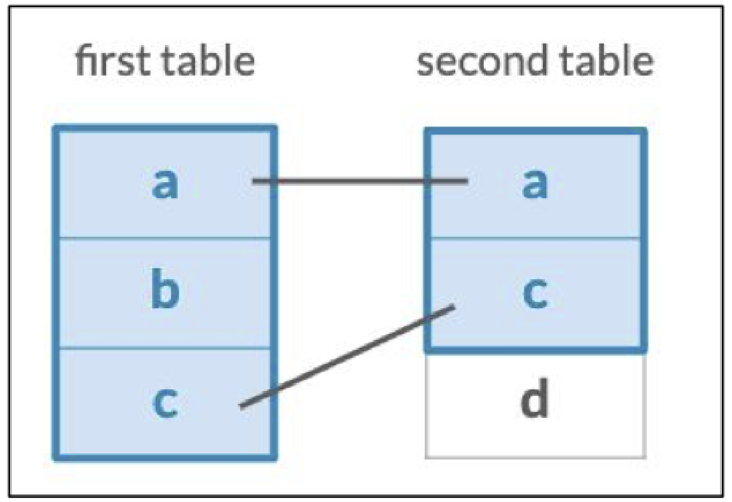
```
# A tibble: 380,643 x 6
         id creation_date.x score.x     id.y creation_date.y score.y
      <int> <date>             <int>    <int> <date>            <int>
 1 22557677 2014-03-21             1 22560670 2014-03-21            2
 2 22557707 2014-03-21             2 22558516 2014-03-21            1
 3 22557707 2014-03-21             2 22558726 2014-03-21            4
 4 22558084 2014-03-21             2 22558085 2014-03-21            0
 5 22558084 2014-03-21             2 22606545 2014-03-24            1
 6 22558084 2014-03-21             2 22610396 2014-03-24            5
 7 22558084 2014-03-21             2 34374729 2015-12-19            0
 8 22558395 2014-03-21             2 22559327 2014-03-21            1
 9 22558395 2014-03-21             2 22560102 2014-03-21            2
10 22558395 2014-03-21             2 22560288 2014-03-21            2
# … with 380,633 more rows
```

The joining verbs

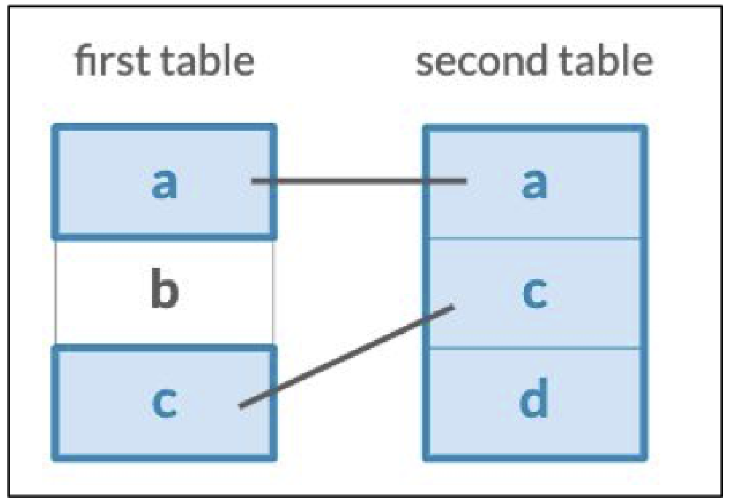
Inner join



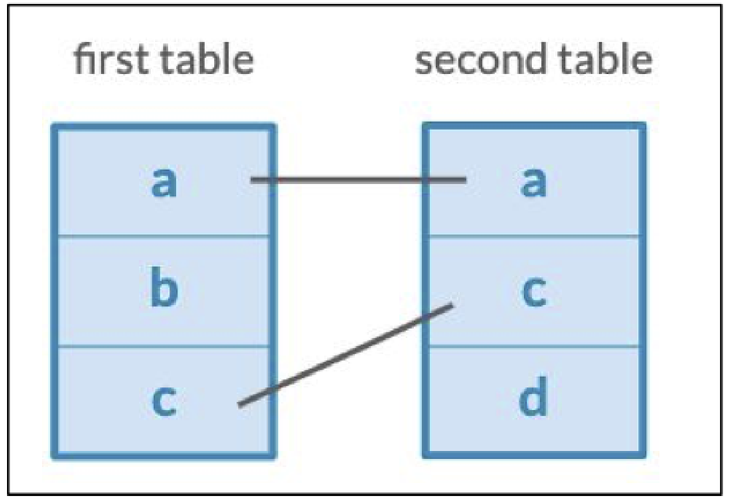
Left join



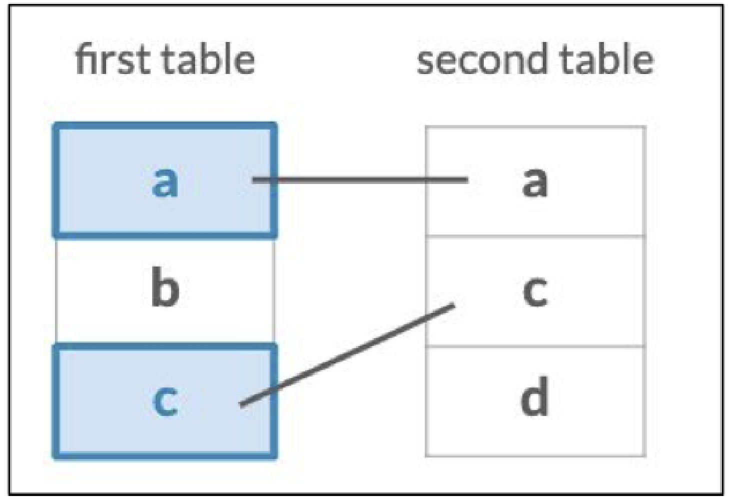
Right join



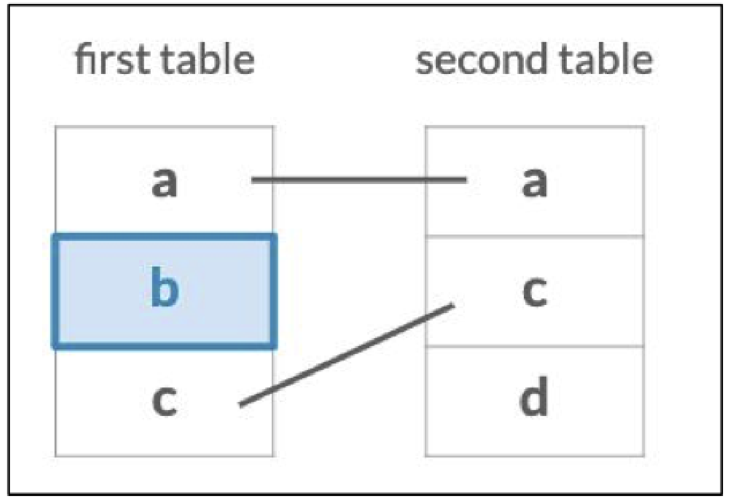
Full join



Semi join



Anti join



The bind_rows verb

JOINING DATA WITH DPLYR

Chris Cardillo
Data Scientist at DataCamp

Comparing tables

```
questions
```

```
# A tibble: 294,735 x 3
         id creation_date score
      <int> <date>        <int>
 1 22557677 2014-03-21        1
 2 22557707 2014-03-21        2
 3 22558084 2014-03-21        2
 4 22558395 2014-03-21        2
 5 22558613 2014-03-21        0
 6 22558677 2014-03-21        2
 7 22558887 2014-03-21        8
 8 22559180 2014-03-21        1
 9 22559312 2014-03-21        0
10 22559322 2014-03-21        2
# … with 294,725 more rows
```

```
answers
```

```
# A tibble: 380,635 x 4
         id creation_date question_id score
      <int> <date>              <int> <int>
 1 39143713 2016-08-25       39143518     3
 2 39143869 2016-08-25       39143518     1
 3 39143935 2016-08-25       39142481     0
 4 39144014 2016-08-25       39024390     0
 5 39144252 2016-08-25       39096741     6
 6 39144375 2016-08-25       39143885     5
 7 39144430 2016-08-25       39144077     0
 8 39144625 2016-08-25       39142728     1
 9 39144794 2016-08-25       39043648     0
10 39145033 2016-08-25       39133170     1
# … with 380,625 more rows
```

Binding rows

```
questions %>%
  bind_rows(answers)
```

```
# A tibble: 675,370 x 4
         id creation_date score question_id
      <int> <date>        <int>       <int>
 1 22557677 2014-03-21        1          NA
 2 22557707 2014-03-21        2          NA
 3 22558084 2014-03-21        2          NA
 4 22558395 2014-03-21        2          NA
 5 22558613 2014-03-21        0          NA
 6 22558677 2014-03-21        2          NA
 7 22558887 2014-03-21        8          NA
 8 22559180 2014-03-21        1          NA
 9 22559312 2014-03-21        0          NA
10 22559322 2014-03-21        2          NA
# … with 675,360 more rows
```

Using bind rows

```
questions_type <- questions %>%
  mutate(type = "question")
```

```
answers_type <- answers %>%
  mutate(type = "answer")

posts <- bind_rows(questions_type, answers_type)
posts
```

```
# A tibble: 675,370 x 5
         id creation_date score type     question_id
      <int> <date>        <int> <chr>          <int>
 1 22557677 2014-03-21        1 question          NA
 2 22557707 2014-03-21        2 question          NA
 3 22558084 2014-03-21        2 question          NA
 4 22558395 2014-03-21        2 question          NA
 5 22558613 2014-03-21        0 question          NA
 6 22558677 2014-03-21        2 question          NA
 7 22558887 2014-03-21        8 question          NA
 8 22559180 2014-03-21        1 question          NA
 9 22559312 2014-03-21        0 question          NA
10 22559322 2014-03-21        2 question          NA
# … with 675,360 more rows
```

Aggregating

```r
posts %>%
  group_by(type) %>%
  summarize(average_score = mean(score))
```

```
# A tibble: 2 x 2
  type     average_score
  <chr>            <dbl>
1 answer            2.88
2 question          1.90
```

Creating date variable

```r
library(lubridate)

posts %>%
  mutate(year = year(creation_date))
```

```
# A tibble: 675,370 x 6
         id creation_date score type     question_id  year
      <int> <date>        <int> <chr>          <int> <dbl>
 1 22557677 2014-03-21        1 question          NA  2014
 2 22557707 2014-03-21        2 question          NA  2014
 3 22558084 2014-03-21        2 question          NA  2014
 4 22558395 2014-03-21        2 question          NA  2014
 5 22558613 2014-03-21        0 question          NA  2014
 6 22558677 2014-03-21        2 question          NA  2014
 7 22558887 2014-03-21        8 question          NA  2014
 8 22559180 2014-03-21        1 question          NA  2014
 9 22559312 2014-03-21        0 question          NA  2014
10 22559322 2014-03-21        2 question          NA  2014
# … with 675,360 more rows
```

Counting date variable

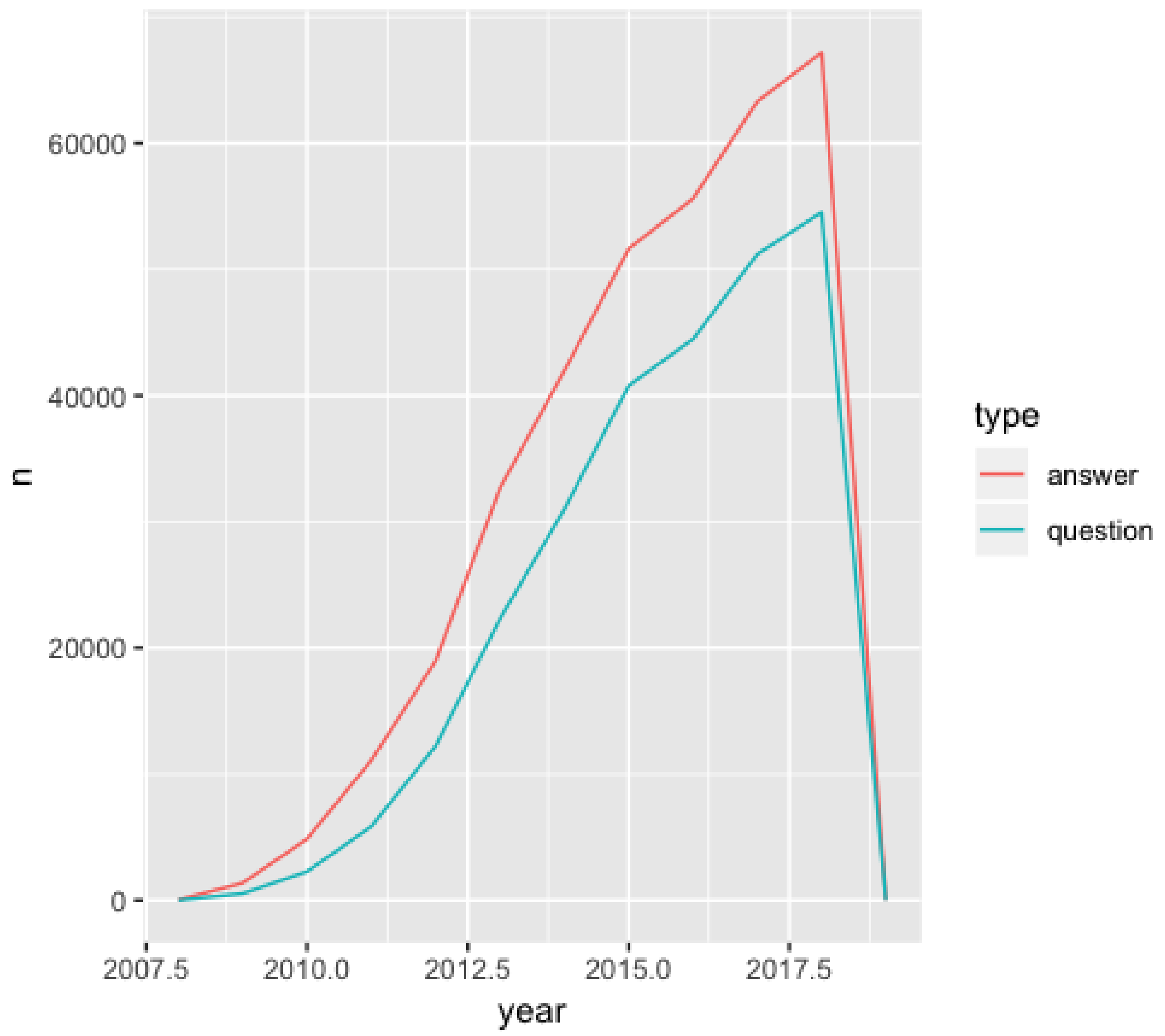
```
posts %>%
  mutate(year = year(creation_date)) %>%
  count(year, type)
```

```
# A tibble: 24 x 3
    year type         n
   <dbl> <chr>    <int>
 1  2008 answer      27
 2  2008 question     8
 3  2009 answer    1356
 4  2009 question   524
 5  2010 answer    4846
 6  2010 question  2264
 7  2011 answer   11077
 8  2011 question  5837
 9  2012 answer   18967
10  2012 question 12210
# … with 14 more rows
```

Plotting date variable

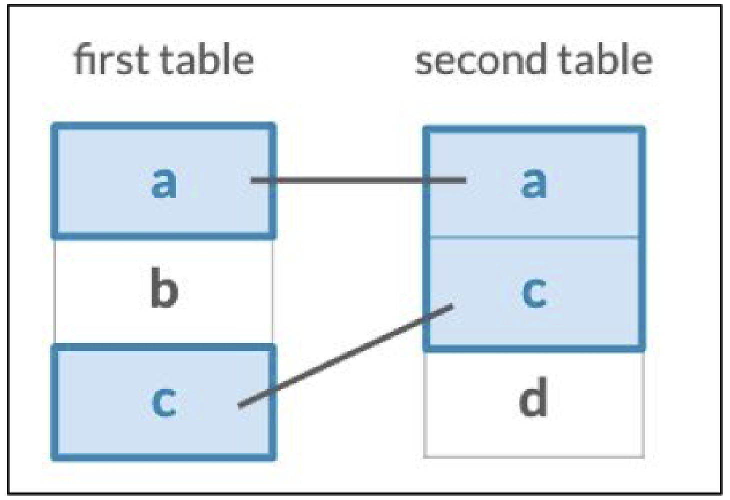
```
questions_answers_year <- posts %>%
  mutate(year = year(creation_date)) %>%
  count(year, type)

ggplot(questions_answers_year, aes(year, n, color = type)) +
  geom_line()
```

The posts plot

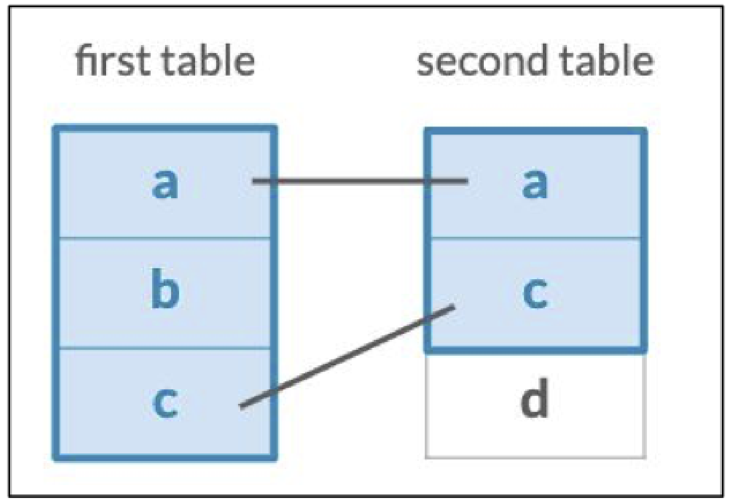


The joining verbs

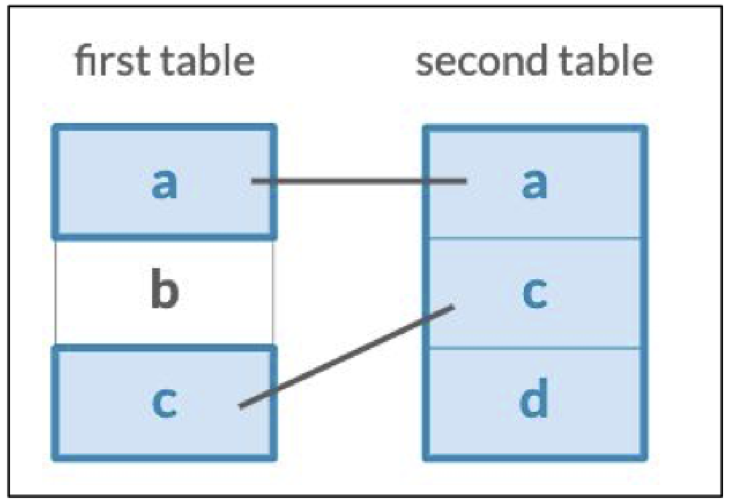
Inner join



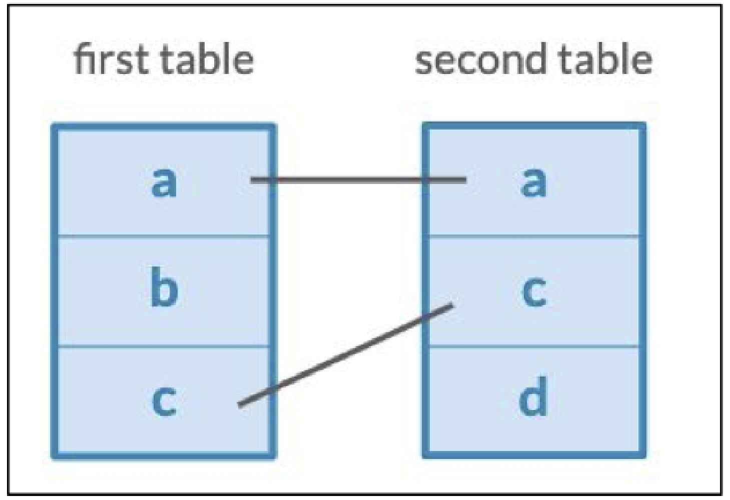
Left join



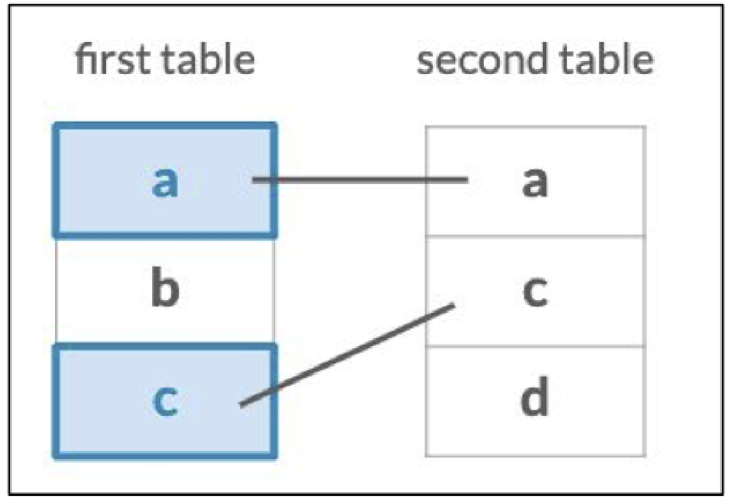
Right join



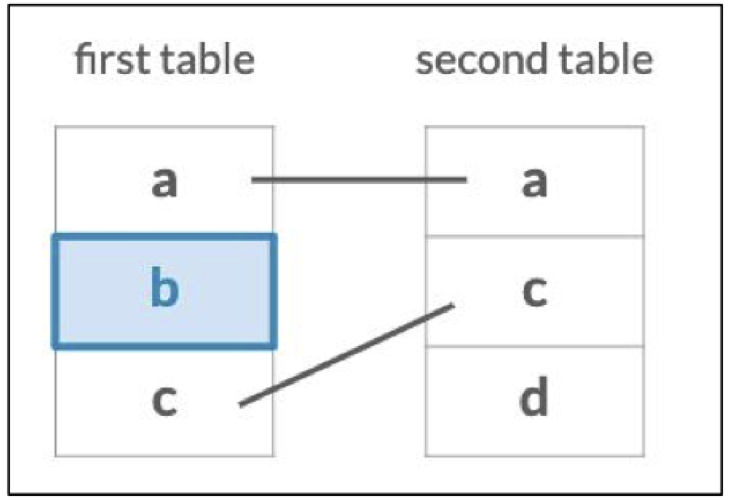
Full join



Semi join

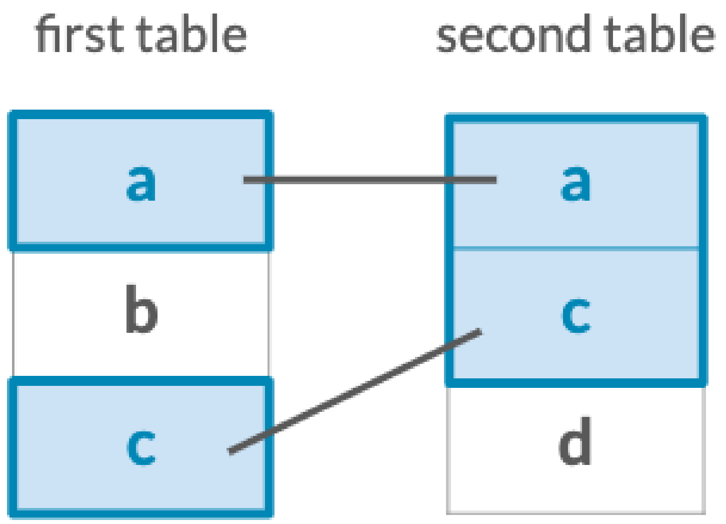


Anti join



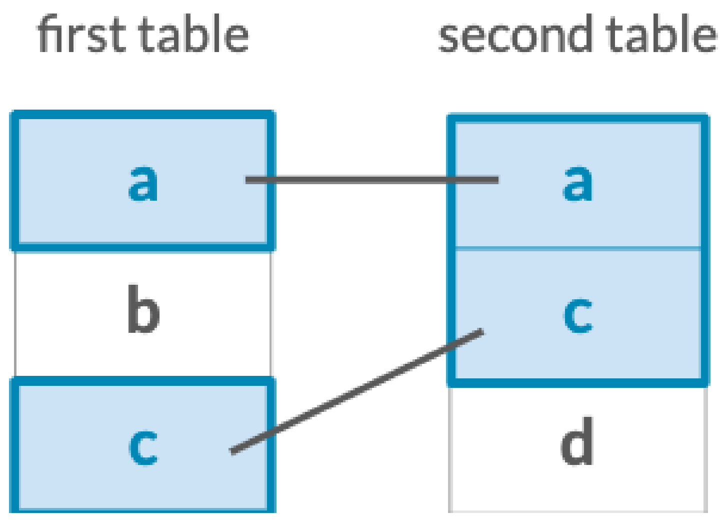
The mutating joins

Inner join

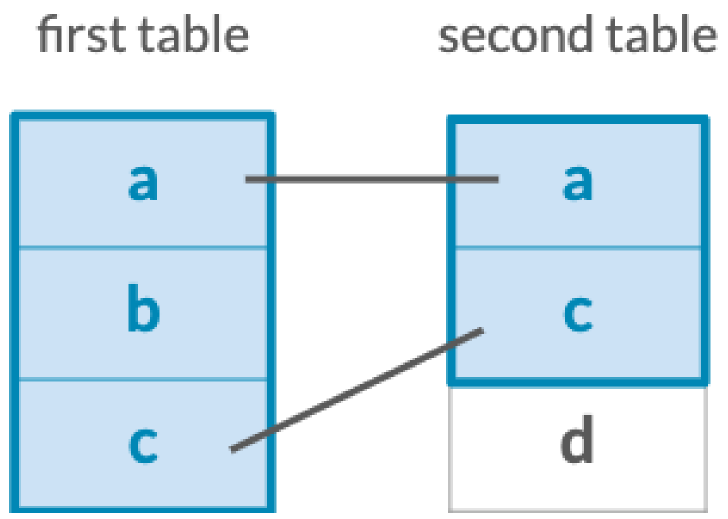


The mutating joins

Inner join

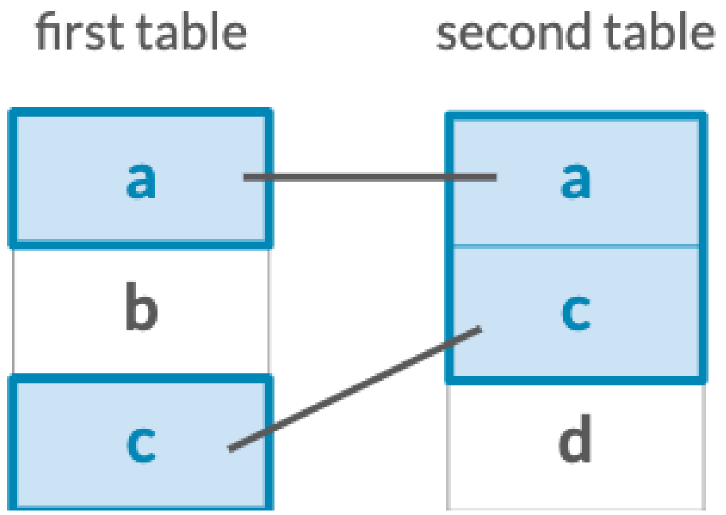


Left join

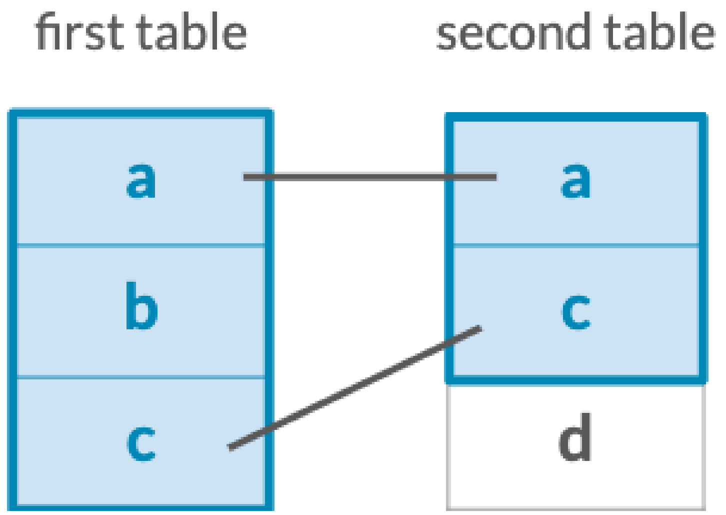


The mutating joins

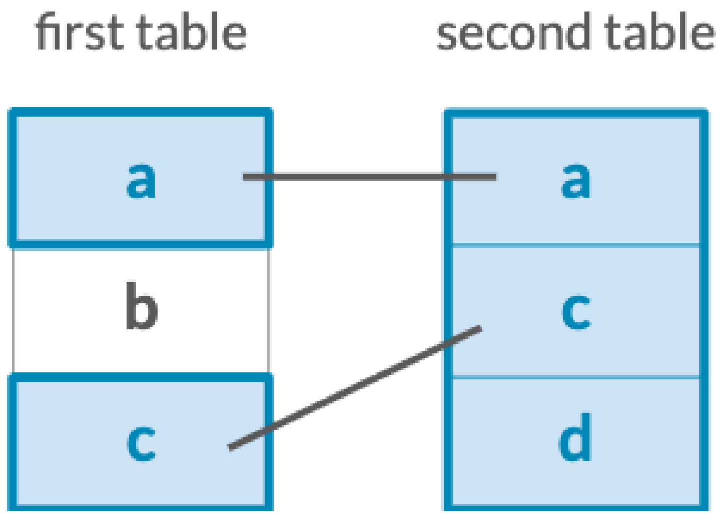
Inner join



Left join

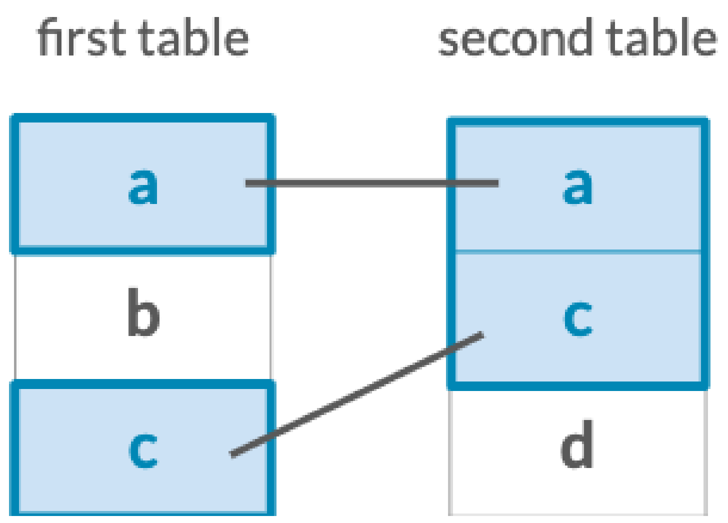


Right join

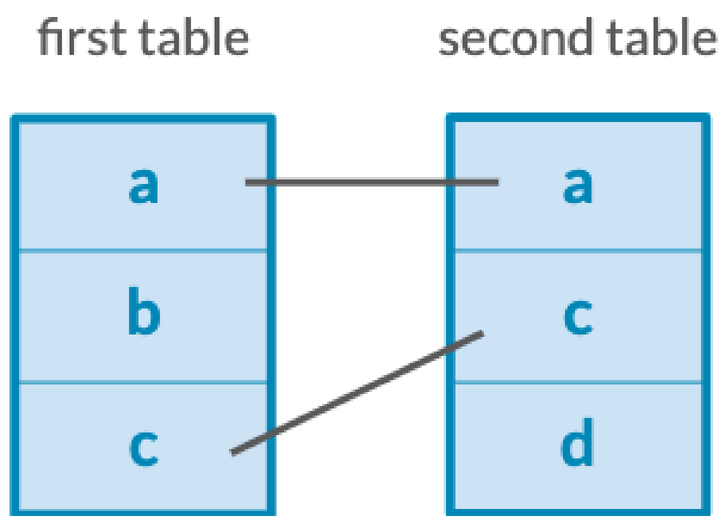


The mutating joins

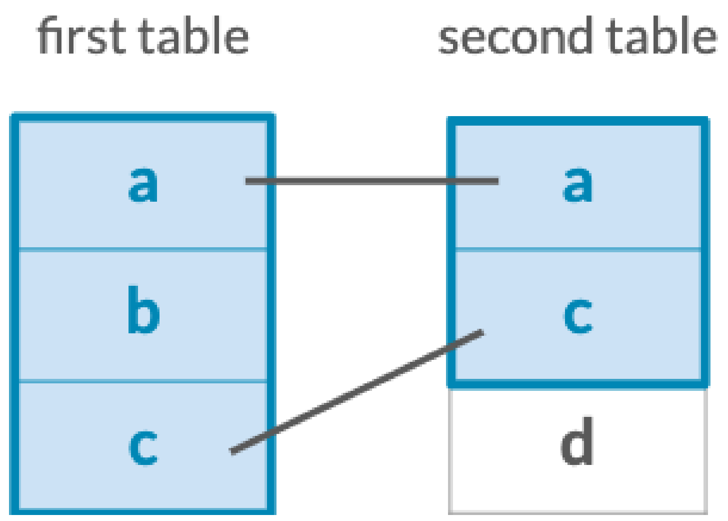
Inner join



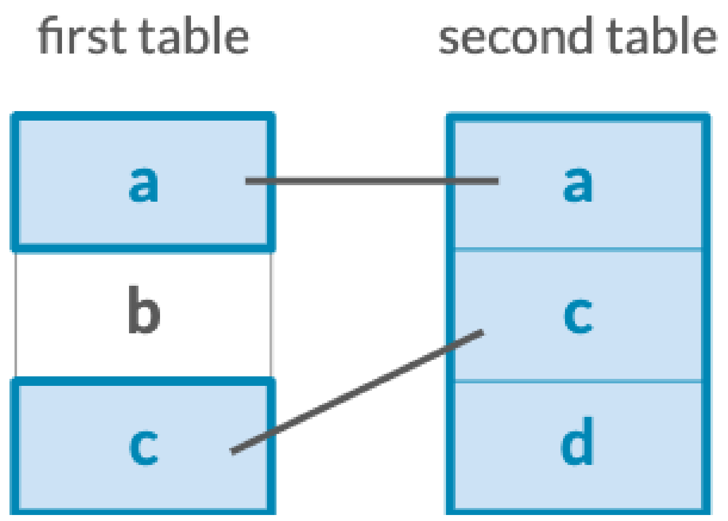
Full join



Left join

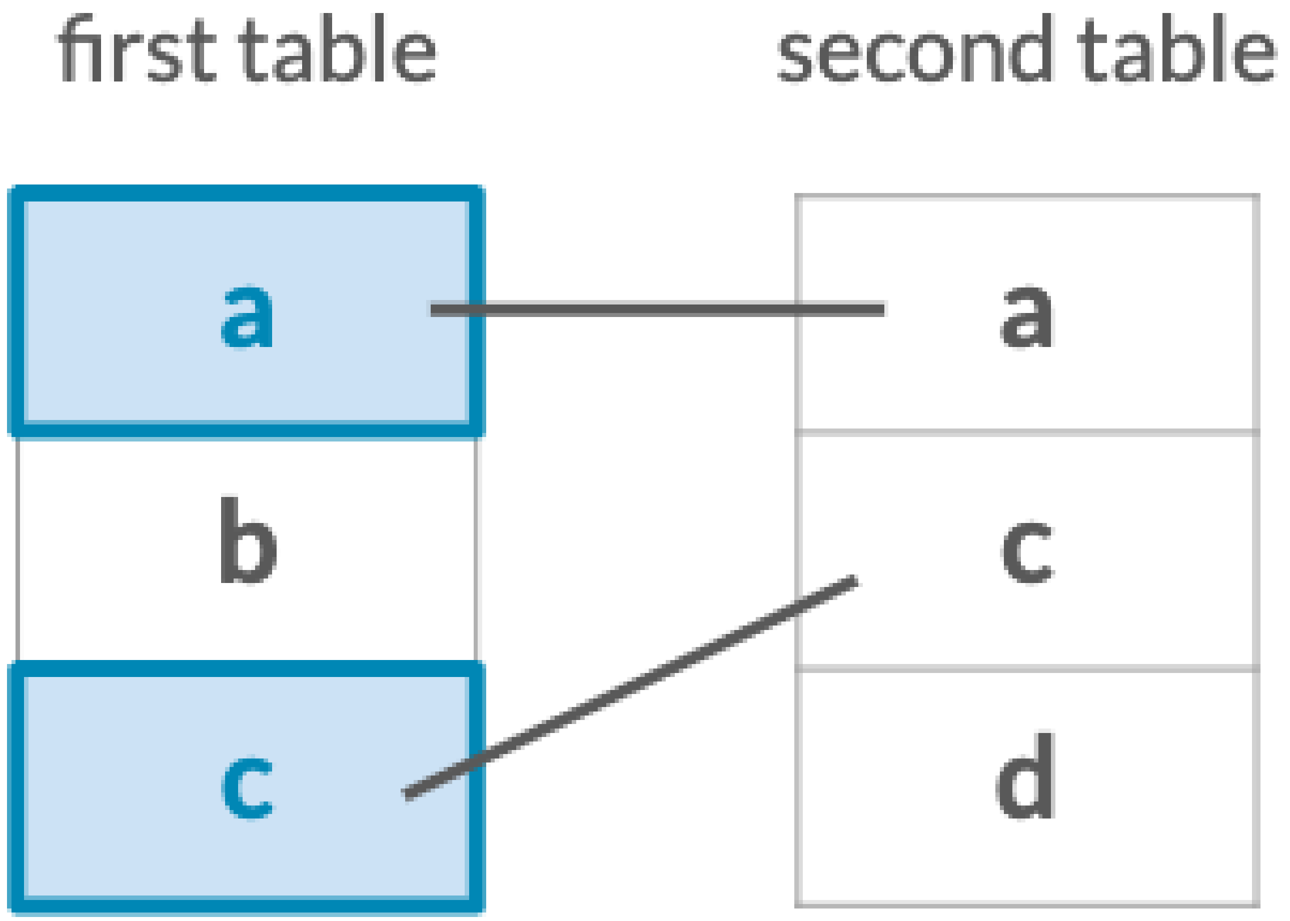


Right join



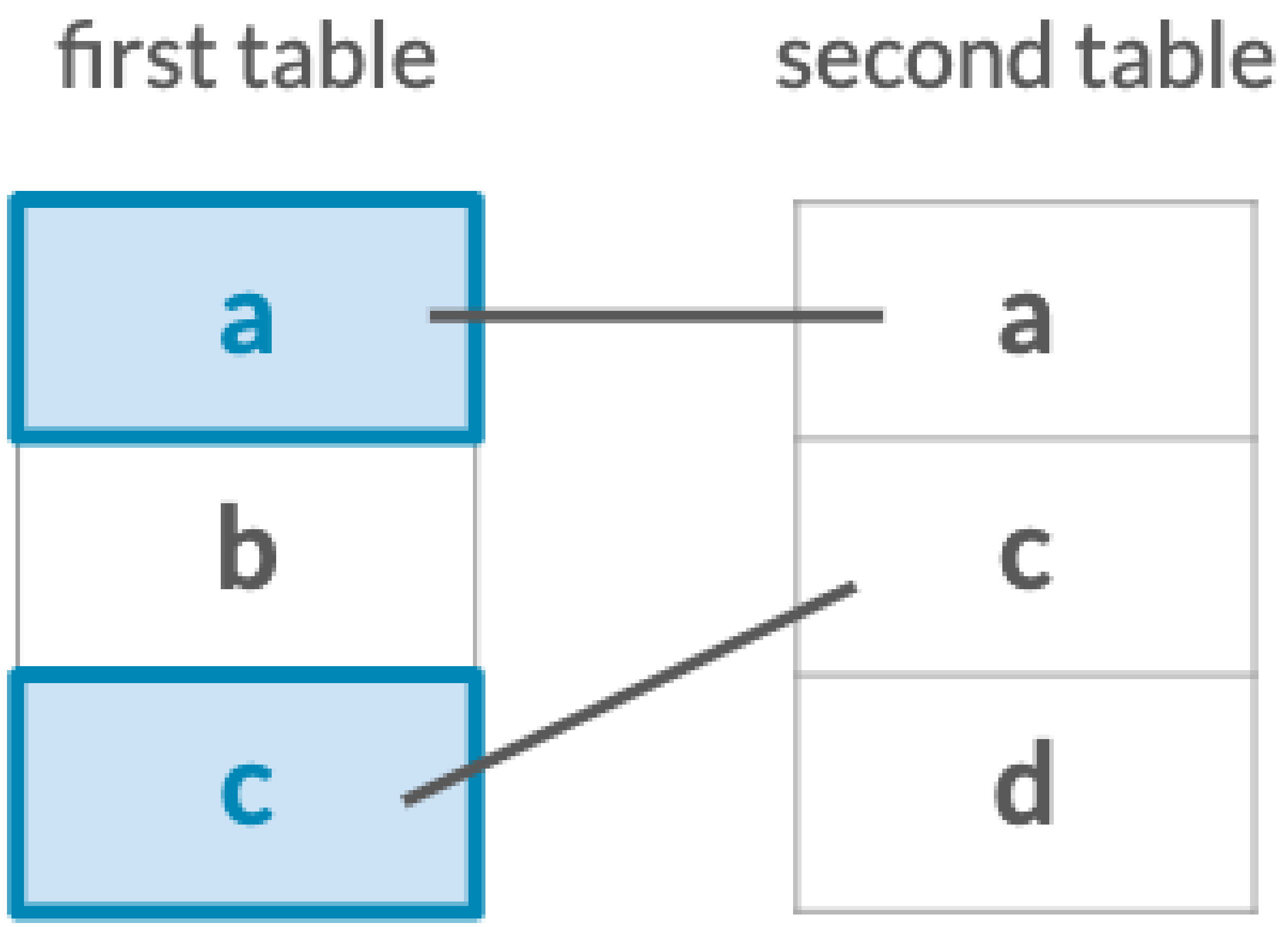
The filtering joins

Semi join

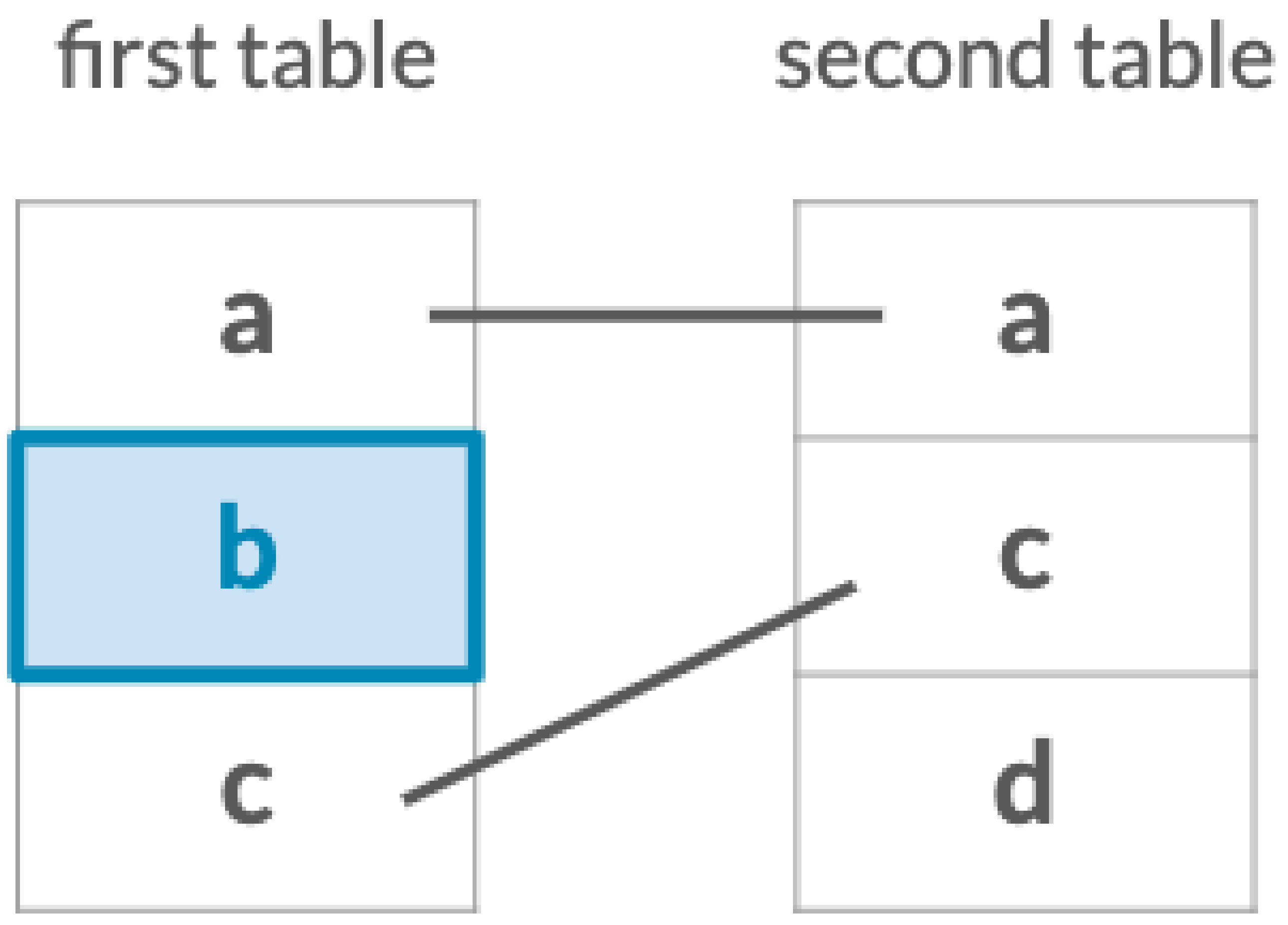


The filtering joins

Semi join



Anti join



Congratulations!