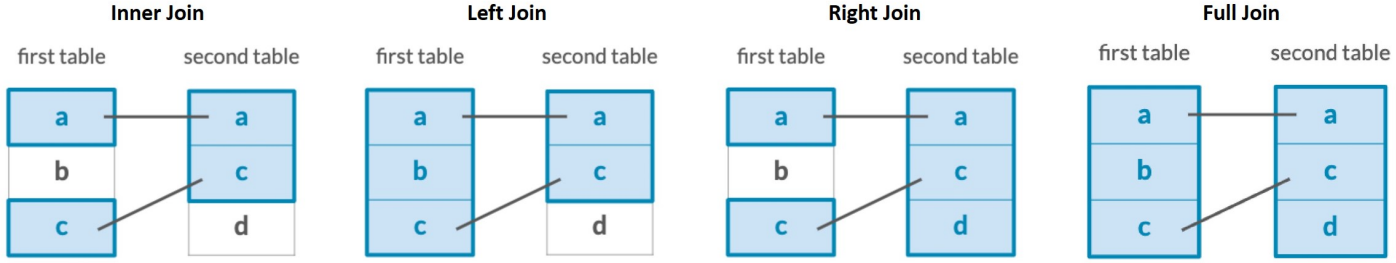
Joining Data with dplyr in R

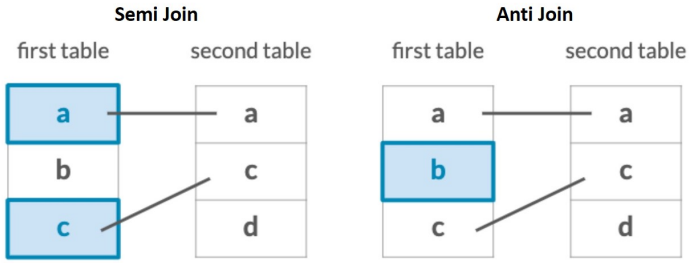
Bourbon0212

2019/9/24

- Overview
- Mutating Joins
  - `inner_join()`
  - `left_join()` & `right_join()`
  - `full_join()`
- Filtering Joins
  - `semi_join()` & `anti_join()`
  - Visualization
- Case Study: Joins on Stack Overflow Data

Overview

- **Mutating Joins**
  - `inner_join()` : Keep only observations that match perfectly between tables.
  - `left_join()` : Keep all observations in the first (left) table, including matching observations in the second one.
  - `right_join()` : Keep all observations in the second (right) table, including matching observations in the first one.
  - `full_join()` : Keep all observations from both tables.
  - `left_join()` & `right_join()` mirrored eached other.
- 

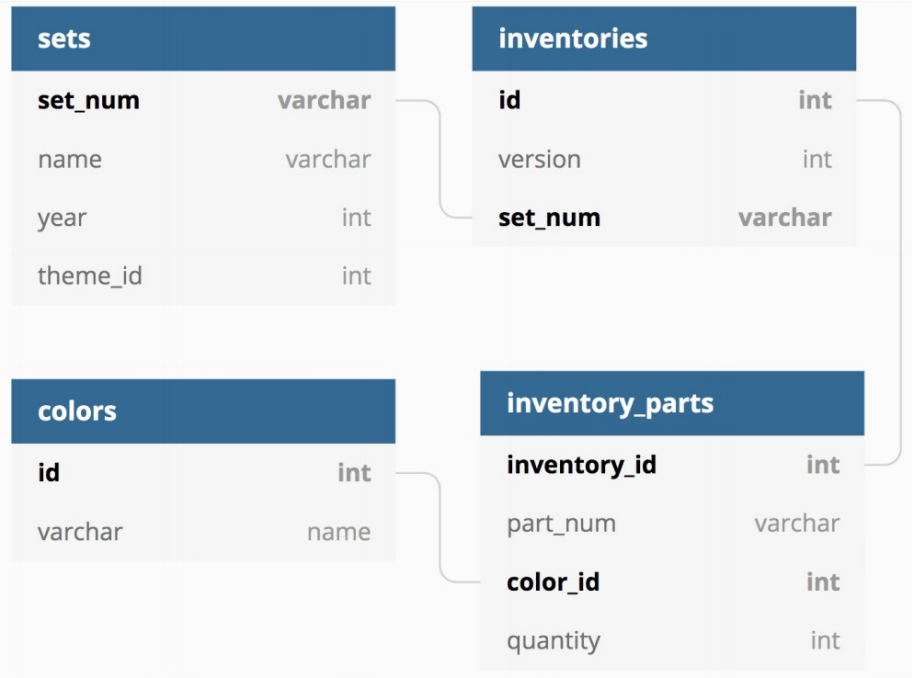
- **Filtering Joins**
  - `semi_join()` : Filter the first table for observations that match the second.
  - `anti_join()` : Filter the first table for observations that don't match the second.
- 

- Arguments
  - `by` : the columns on which they should be joined.
    - if the col_names are the same, specifing that column works.
  - `suffix` : the suffix of the same col_names after joining.
- Always keeps mind on the **SIZE** of the result after joining tables.
- `bind_rows()` : Stack the two tables.

Mutating Joins

`inner_join()`

> What's the most common color of a LEGO piece?




```
library(tidyverse)

chunk1 <- sets %>%
    inner_join(inventories, by = "set_num") %>%
    inner_join(inventory_parts, by = c("id" = "inventory_id")) %>%
    inner_join(colors, by = c("color_id" = "id"), suffix = c("_set", "_color"))

head(chunk1)
```

```
## # A tibble: 6 x 11
##   set_num name_set  year theme_id    id version part_num color_id quantity
##   <chr>   <chr>    <dbl>    <dbl> <dbl>   <dbl> <chr>       <dbl>    <dbl>
## 1 700.3-1 Medium ~  1949      365 24197       1 bdoor01         2        2
## 2 700.3-1 Medium ~  1949      365 24197       1 bdoor01        15        1
## 3 700.3-1 Medium ~  1949      365 24197       1 bdoor01         4        1
## 4 700.3-1 Medium ~  1949      365 24197       1 bslot02        15        6
```

```
## 5 700.3-1 Medium ~  1949      365 24197       1 bslot02         2        6
## 6 700.3-1 Medium ~  1949      365 24197       1 bslot02         4        6
## # ... with 2 more variables: name_color <chr>, rgb <chr>
```

```
chunk1 %>%
  count(name_color) %>%
  arrange(desc(n))
```

```
## # A tibble: 134 x 2
##    name_color            n
##    <chr>             <int>
##  1 Black             48068
##  2 White             30105
##  3 Light Bluish Gray 26024
##  4 Red               21602
##  5 Dark Bluish Gray  19948
##  6 Yellow            17088
##  7 Blue              12980
##  8 Light Gray         8632
##  9 Reddish Brown      6960
## 10 Tan                6664
## # ... with 124 more rows
```

Notice that Black and White are the two most prominent colors.

`left_join()` & `right_join()`

> Compare Millennium Falcon and Star Destroyer sets of different parts and colors.

```
# Data Preparation
inventory_parts_joined <- inventories %>%
```

```
  inner_join(inventory_parts, by = c("id" = "inventory_id")) %>%
  select(-id, -version) %>%
  arrange(desc(quantity))

millennium_falcon <- inventory_parts_joined %>%
  filter(set_num == "7965-1")

star_destroyer <- inventory_parts_joined %>%
  filter(set_num == "75190-1")

# Combine the star_destroyer and millennium_falcon tables
millennium_falcon %>%
  left_join(star_destroyer, by = c("part_num", "color_id"), suffix = c("_falcon", "_star_destroyer")) %>%
  select(-set_num_falcon, -set_num_star_destroyer)
```

```
## # A tibble: 263 x 4
##    part_num color_id quantity_falcon quantity_star_destroyer
##    <chr>       <dbl>           <dbl>                   <dbl>
##  1 63868          71              62                      NA
##  2 3023            0              60                      NA
##  3 3021           72              46                       6
##  4 2780            0              37                      36
##  5 60478          72              36                      NA
##  6 6636           71              34                       2
##  7 3009           71              28                       2
##  8 3665           71              22                      NA
##  9 2412b          72              20                      11
## 10 3010           71              19                      NA
## # ... with 253 more rows
```

> Compare Millennium Falcon and Star Destroyer sets of different colors.

```
# Aggregate Millennium Falcon for the total quantity in each part
millennium_falcon_colors <- millennium_falcon %>%
```

```
  group_by(color_id) %>%
  summarize(total_quantity = sum(quantity))

# Aggregate Star Destroyer for the total quantity in each part
star_destroyer_colors <- star_destroyer %>%
  group_by(color_id) %>%
  summarize(total_quantity = sum(quantity))

# Left join the Millennium Falcon colors to the Star Destroyer colors
millennium_falcon_colors %>%
  left_join(star_destroyer_colors, by = "color_id", suffix = c("_falcon", "_star_destroyer"))
```

```
## # A tibble: 21 x 3
##    color_id total_quantity_falcon total_quantity_star_destroyer
##       <dbl>                 <dbl>                         <dbl>
##  1        0                   201                           336
##  2        1                    15                            23
##  3        4                    17                            53
##  4       14                     3                             4
##  5       15                    15                            17
##  6       19                    95                            12
##  7       28                     3                            16
##  8       33                     5                            NA
##  9       36                     1                            14
## 10       41                     6                            15
## # ... with 11 more rows
```

> LEGO parts counts and its name.

```
parts %>%
    count(part_cat_id) %>%
    right_join(part_categories, by = c("part_cat_id" = "id")) %>%
    # Use replace_na to replace missing values in the n column
    replace_na(list(n = 0))
```

```
## # A tibble: 64 x 3
##    part_cat_id     n name
##          <dbl> <dbl> <chr>
##  1           1   135 Baseplates
##  2           3   303 Bricks Sloped
##  3           4  1900 Duplo, Quatro and Primo
##  4           5   107 Bricks Special
##  5           6   128 Bricks Wedged
##  6           7    97 Containers
##  7           8    24 Technic Bricks
##  8           9   167 Plates Special
##  9          11   490 Bricks
## 10          12    85 Technic Connectors
## # ... with 54 more rows
```

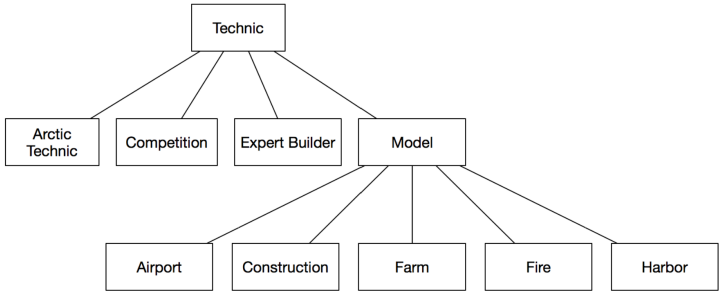
> Joining tables to themselves to find the relationships between themes.

```
# Join themes to itself again to find the grandchild relationships
themes %>%
  inner_join(themes, by = c("id" = "parent_id"), suffix = c("_parent", "_child")) %>%
  inner_join(themes, by = c("id_child" = "parent_id"), suffix = c("_parent", "_grandchild"))
```

```
## # A tibble: 158 x 7
##    id_parent name_parent parent_id id_child name_child id_grandchild name
##        <dbl> <chr>           <dbl>    <dbl> <chr>              <dbl> <chr>
##  1         1 Technic            NA        5 Model                  6 Airp~
##  2         1 Technic            NA        5 Model                  7 Cons~
##  3         1 Technic            NA        5 Model                  8 Farm
##  4         1 Technic            NA        5 Model                  9 Fire
##  5         1 Technic            NA        5 Model                 10 Harb~
##  6         1 Technic            NA        5 Model                 11 Off-~
##  7         1 Technic            NA        5 Model                 12 Race
##  8         1 Technic            NA        5 Model                 13 Ridi~
##  9         1 Technic            NA        5 Model                 14 Robot
```

```
## 10         1 Technic            NA        5 Model                15 Traf~
## # ... with 148 more rows
```

From the result, we can find the relationship between themes, like the following figure shows.



full_join()

> Comparing Batman and Star Wars LEGO parts.

```
inventory_sets_themes <- inventory_parts_joined %>%
  inner_join(sets, by = "set_num") %>%
  inner_join(themes, by = c("theme_id" = "id"), suffix = c("_set", "_theme"))

batman <- inventory_sets_themes %>%
  filter(name_theme == "Batman")

star_wars <- inventory_sets_themes %>%
  filter(name_theme == "Star Wars")

# Count the part number and color id, weight by quantity
batman_parts <- batman %>%
  count(part_num, color_id, wt = quantity)
```

```
star_wars_parts <- star_wars %>%
  count(part_num, color_id, wt = quantity)

# Full join Batman and Star Wars LEGO parts
parts_joined <- batman_parts %>%
  # Combine the star_wars_parts table
  full_join(star_wars_parts, by = c("part_num", "color_id"), suffix = c("_batman", "_star_wars")) %>%
  # Replace NAs with 0s in the n_batman and n_star_wars columns
  replace_na(list(n_batman = 0, n_star_wars = 0))
parts_joined
```

```
## # A tibble: 3,628 x 4
##    part_num color_id n_batman n_star_wars
##    <chr>       <dbl>    <dbl>       <dbl>
##  1 10113           0       11           0
##  2 10113         272        1           0
##  3 10113         320        1           0
##  4 10183          57        1           0
##  5 10190           0        2           0
##  6 10201           0        1          21
##  7 10201           4        3           0
##  8 10201          14        1           0
##  9 10201          15        6           0
## 10 10201          71        4           5
## # ... with 3,618 more rows
```

```
parts_joined %>%
  # Sort the number of star wars pieces in descending order
  arrange(desc(n_star_wars)) %>%
  # Join the colors table to the parts_joined table
  inner_join(colors, by = c("color_id" = "id")) %>%
  # Join the parts table to the previous join
  inner_join(parts, by = "part_num", suffix = c("_color", "_part"))
```

```
## # A tibble: 3,628 x 8
##    part_num color_id n_batman n_star_wars name_color  rgb   name_part
##    <chr>       <dbl>    <dbl>       <dbl> <chr>       <chr> <chr>
##  1 2780            0      104         392 Black       #051~ Technic Pin w~
##  2 32062           0        1         141 Black       #051~ Technic Axle ~
##  3 4274            1       56         118 Blue        #005~ Technic Pin 1~
##  4 6141           36       11         117 Trans-Red   #C91~ Plate Round 1~
##  5 3023           71       10         106 Light Blui~ #A0A~ Plate 1 x 2
##  6 6558            1       30         106 Blue        #005~ Technic Pin L~
##  7 43093           1       44          99 Blue        #005~ Technic Axle ~
##  8 3022           72       14          95 Dark Bluis~ #6C6~ Plate 2 x 2
##  9 2357           19        0          84 Tan         #E4C~ Brick 2 x 2 C~
## 10 6141          179       90          81 Flat Silver #898~ Plate Round 1~
## # ... with 3,618 more rows, and 1 more variable: part_cat_id <dbl>
```

```
Since the pieces are sorted by number of Star Wars pieces in descending order, you can see that the most common S
tar Wars piece is Black and has the part number 2780. While there are 392 pieces of this part in the Star Wars th
eme, you can also see from the table that there are 104 pieces of the same part in the Batman theme.
```

Filtering Joins

`semi_join()` & `anti_join()`

> Something within one set but not another.

```
batmobile <- inventory_parts_joined %>%
  filter(set_num == "7784-1") %>%
  select(-set_num)

batwing <- inventory_parts_joined %>%
  filter(set_num == "70916-1") %>%
  select(-set_num)
```

```
# Filter the batwing set for parts that are also in the batmobile set
batwing %>%
  semi_join(batmobile, by = c("part_num"))
```

```
## # A tibble: 126 x 3
##    part_num color_id quantity
##    <chr>       <dbl>    <dbl>
##  1 3023            0       22
##  2 3024            0       22
##  3 3623            0       20
##  4 2780            0       17
##  5 3666            0       16
##  6 3710            0       14
##  7 6141            4       12
##  8 2412b          71       10
##  9 6141           72       10
## 10 6558            1        9
## # ... with 116 more rows
```

```
# Filter the batwing set for parts that aren't in the batmobile set
batwing %>%
  anti_join(batmobile, by = c("part_num"))
```

```
## # A tibble: 183 x 3
##    part_num color_id quantity
##    <chr>       <dbl>    <dbl>
##  1 11477           0       18
##  2 99207          71       18
##  3 22385           0       14
##  4 99563           0       13
##  5 10247          72       12
##  6 2877           72       12
##  7 61409          72       12
##  8 11153           0       10
```

```
##  9 98138          46       10
## 10 2419           72        9
## # ... with 173 more rows
```

```
 Based on these joins, we now know that there are 126 parts in the batwing set that are also in the batmobile se
t, and 183 parts that are in the batwing set that aren't in the batmobile set.
```

Visualization

```
inventory_parts_themes <- inventories %>%
  inner_join(inventory_parts, by = c("id" = "inventory_id")) %>%
  arrange(desc(quantity)) %>%
  select(-id, -version) %>%
  inner_join(sets, by = "set_num") %>%
  inner_join(themes, by = c("theme_id" = "id"), suffix = c("_set", "_theme"))

# Aggregating sets to look at their differences
batman_colors <- inventory_parts_themes %>%
  # Filter the inventory_parts_themes table for the Batman theme
  filter(name_theme == "Batman") %>%
  group_by(color_id) %>%
  summarize(total = sum(quantity)) %>%
  # Add a percent column of the total divided by the sum of the total
  mutate(percent = total / sum(total))

star_wars_colors <- inventory_parts_themes %>%
  filter(name_theme == "Star Wars") %>%
  group_by(color_id) %>%
  summarize(total = sum(quantity)) %>%
  mutate(percent = total / sum(total))

# Combining sets
colors_joined <- batman_colors %>%
  full_join(star_wars_colors, by = "color_id", suffix = c("_batman", "_star_wars")) %>%
```

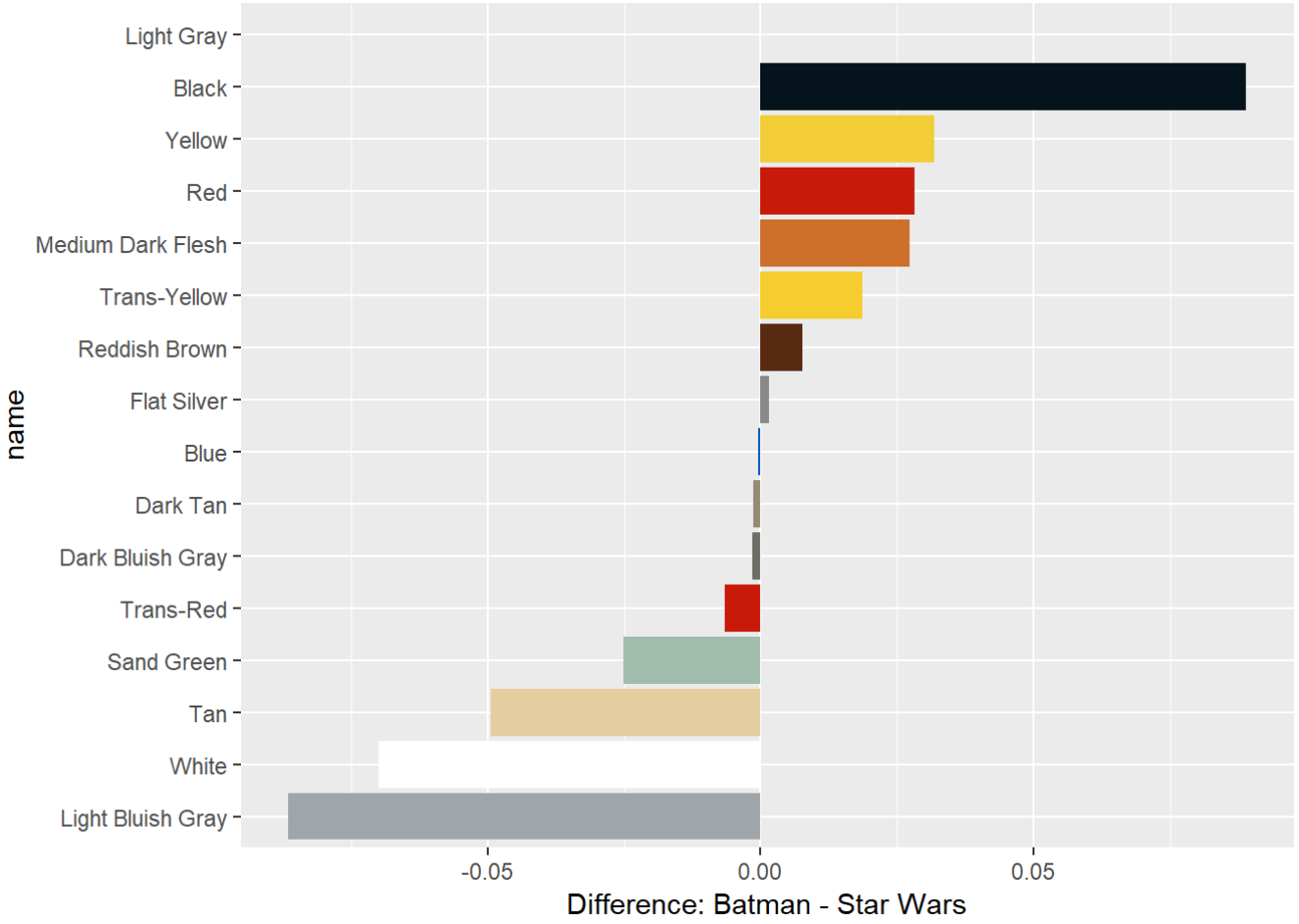
```
  replace_na(list(total_batman = 0, total_star_wars = 0)) %>%
  inner_join(colors, by = c("color_id" = "id")) %>%
  mutate(difference = percent_batman - percent_star_wars,
         total = total_batman + total_star_wars) %>%
  filter(total >= 200) %>%
  mutate(name = fct_reorder(name, difference)) # Reorder according to difference
colors_joined
```

```
## # A tibble: 16 x 9
##    color_id total_batman percent_batman total_star_wars percent_star_wars
##       <dbl>        <dbl>          <dbl>           <dbl>             <dbl>
##  1        0         2807        0.296              3258           0.207
##  2        1          243        0.0256              410           0.0261
##  3        4          529        0.0558              434           0.0276
##  4       14          426        0.0449              207           0.0132
##  5       15          404        0.0426             1771           0.113
##  6       19          142        0.0150             1012           0.0644
##  7       28           98        0.0103              183           0.0116
##  8       36           86        0.00907             246           0.0156
##  9       46          200        0.0211               39           0.00248
## 10       70          297        0.0313              373           0.0237
## 11       71         1148        0.121              3264           0.208
## 12       72         1453        0.153              2433           0.155
## 13       84          278        0.0293               31           0.00197
## 14      179          154        0.0162              232           0.0148
## 15      378           22        0.00232             430           0.0273
## 16        7            0       NA                   209           0.0133
## # ... with 4 more variables: name <fct>, rgb <chr>, difference <dbl>,
## #   total <dbl>
```

```
# Visualize with ggplot2
color_palette <- setNames(colors_joined$rgb, colors_joined$name)

ggplot(colors_joined, aes(x = name, difference, fill = name)) +
  geom_col() +
  coord_flip() +
```

```
  scale_fill_manual(values = color_palette, guide = FALSE) +
  labs(y = "Difference: Batman - Star Wars")
```




```
 As you can see from the plot, the Batman set has more black, yellow, and red, while the Star Wars set has more l
ight bluish gray, white, and tan.
```

Case Study: Joins on Stack Overflow Data

Three of the Stack Overflow survey datasets are `questions`, `question_tags`, and `tags`:

- questions: an ID and the score, or how many times the question has been upvoted; the data only includes R-based questions.
- question_tags: a tag ID for each question and the question's id.
- tags: a tag id and the tag's name, which can be used to identify the subject of each question, such as ggplot2 or dplyr.

- `bind_rows()`: Stack the two tables.

```
answer_counts <- answers %>%
    count(question_id, sort = TRUE)

question_answer_counts <- questions %>%
    left_join(answer_counts, by = c("id" = "question_id")) %>%
    replace_na(list(n = 0))

tagged_answers <- question_answer_counts %>%
    inner_join(question_tags, by = c("id" = "question_id")) %>%
    inner_join(tags, by = c("tag_id" = "id"))

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

answers_with_tags <- answers %>%
  inner_join(question_tags, by = "question_id") %>%
  inner_join(tags, by = c("tag_id" = "id"))

# Combine the two tables into posts_with_tags
posts_with_tags <- bind_rows(questions_with_tags %>% mutate(type = "question"), answers_with_tags %>% mutate(type
= "answer"))

# Add a year column, then aggregate by type, year, and tag_name
library(lubridate)

by_type_year_tag <- posts_with_tags %>%
  mutate(year = year(creation_date)) %>%
```

```
  count(type, year, tag_name)

# Filter for the dplyr and ggplot2 tag names
by_type_year_tag_filtered <- by_type_year_tag %>%
  filter(tag_name %in% c("dplyr", "ggplot2"))

# Create a line plot faceted by the tag name
ggplot(by_type_year_tag_filtered, aes(year, n, color = type)) +
  geom_line() +
  facet_wrap(~ tag_name)
```


```
Notice answers on dplyr questions are growing faster than dplyr questions themselves; meaning the average dplyr q
uestion has more answers than the average ggplot2 question.
```