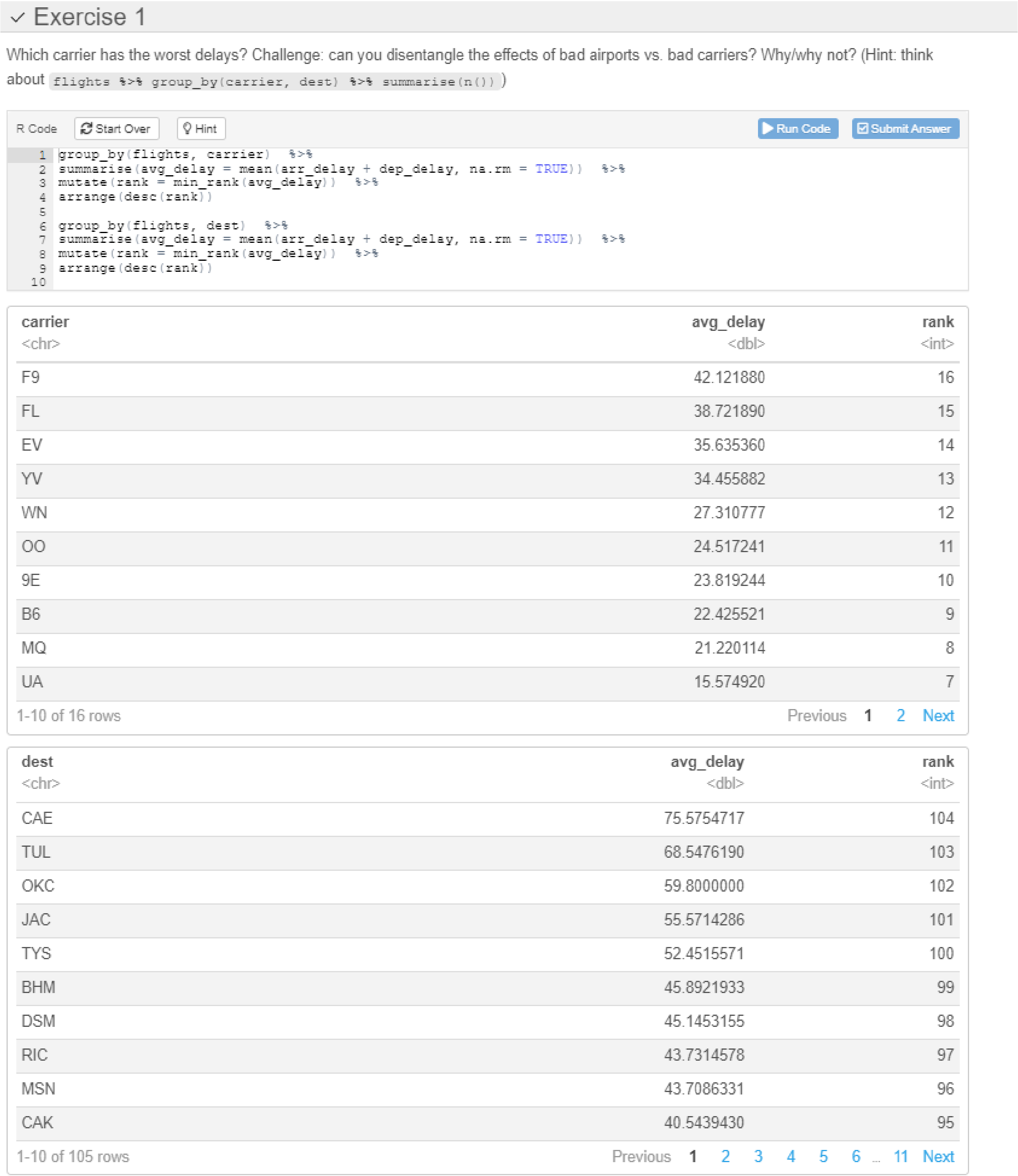
Eddie Aguilar



✓ Exercise 1

Which carrier has the worst delays? Challenge: can you disentangle the effects of bad airports vs. bad carriers? Why/why not? (Hint: think about `flights %>% group_by(carrier, dest) %>% summarise(n())`)

```r
group_by(flights, carrier)  %>%
summarise(avg_delay = mean(arr_delay + dep_delay, na.rm = TRUE))  %>%
mutate(rank = min_rank(avg_delay))  %>%
arrange(desc(rank))

group_by(flights, dest)  %>%
summarise(avg_delay = mean(arr_delay + dep_delay, na.rm = TRUE))  %>%
mutate(rank = min_rank(avg_delay))  %>%
arrange(desc(rank))
```

carrier <chr>	avg_delay <dbl>	rank <int>
F9	42.121880	16
FL	38.721890	15
EV	35.635360	14
YV	34.455882	13
WN	27.310777	12
OO	24.517241	11
9E	23.819244	10
B6	22.425521	9
MQ	21.220114	8
UA	15.574920	7

1-10 of 16 rows

dest <chr>	avg_delay <dbl>	rank <int>
CAE	75.5754717	104
TUL	68.5476190	103
OKC	59.8000000	102
JAC	55.5714286	101
TYS	52.4515571	100
BHM	45.8921933	99
DSM	45.1453155	98
RIC	43.7314578	97
MSN	43.7086331	96
CAK	40.5439430	95

1-10 of 105 rows

✓ Exercise 2

For each plane, count the number of flights before the first delay of greater than 1 hour.

R Code Start Over Hint ▶ Run Code ☑ Submit Answer

```r
group_by(flights, tailnum)  %>%
mutate(hour_delay = dep_delay > 60)  %>%
select(tailnum, everything())  %>%
arrange(tailnum, month, day)  %>%
filter(!is.na(dep_delay))  %>%
summarise(Flights_before_first_delay = sum(cumall(!hour_delay)))

```

tailnum <chr>	Flights_before_first_delay <int>
D942DN	0
N0EGMQ	53
N10156	9
N102UW	25
N103US	46
N104UW	3
N10575	0
N105UW	22
N107US	20
N108UW	36

1-10 of 1,000 rows Previous **1** 2 3 4 5 6 ... 100 Next

✓ Grouping by multiple variables

When you group by multiple variables, each summary peels off one level of the grouping. That makes it easy to progressively roll up a dataset. Run the code below and inspect each result to see how its grouping criteria has changed (the grouping criteria is displayed at the top of the tibble).

R Code Start Over ▶ Run Code

```r
daily <- group_by(flights, year, month, day)
(per_day <- summarise(daily, total = sum(dep_delay, na.rm = TRUE)))
(per_month <- summarise(per_day, total = sum(total, na.rm = TRUE)))
(per_year  <- summarise(per_month, total = sum(total, na.rm = TRUE)))
```

```
`summarise()` has grouped output by 'year', 'month'. You can override using the
`.groups` argument.
```

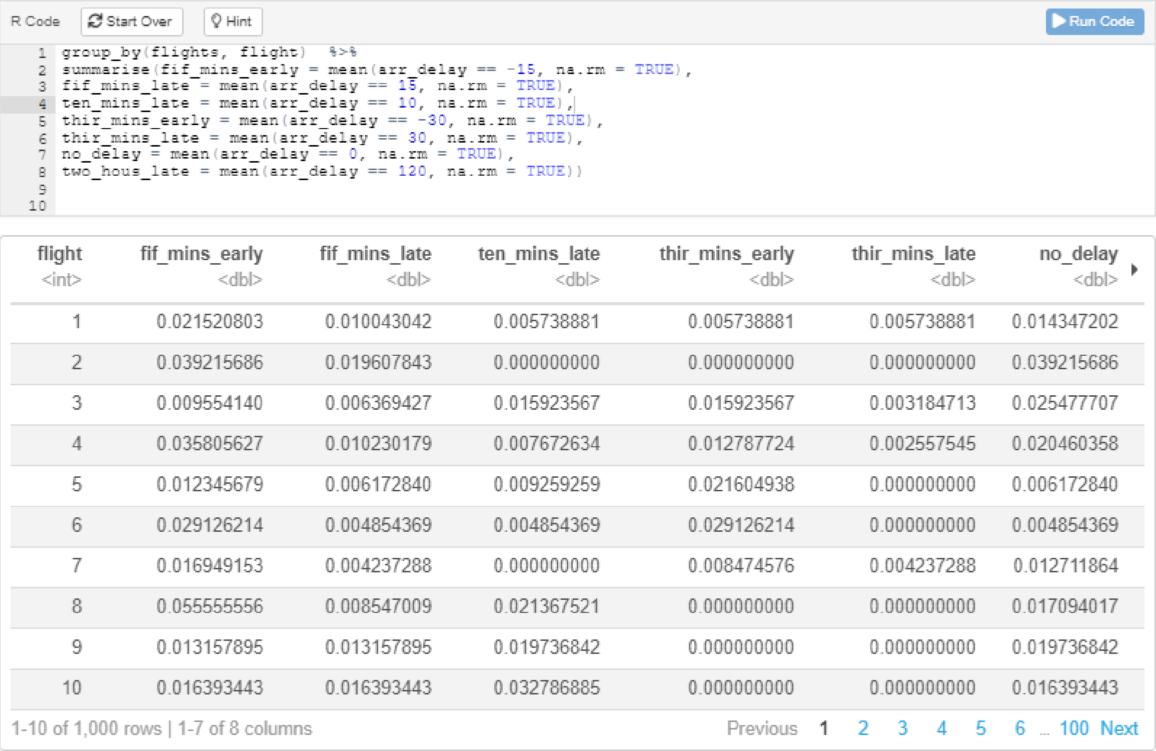
year <int>	month <int>	day <int>	total <dbl>
2013	1	1	9678
2013	1	2	12958
2013	1	3	9933
2013	1	4	8137

✓ Exercise 3

Brainstorm at least 5 different ways to assess the typical delay characteristics of a group of flights. Consider the following scenarios:

- A flight is 15 minutes early 50% of the time, and 15 minutes late 50% of the time.
- A flight is always 10 minutes late.
- A flight is 30 minutes early 50% of the time, and 30 minutes late 50% of the time.
- 99% of the time a flight is on time. 1% of the time it’s 2 hours late.

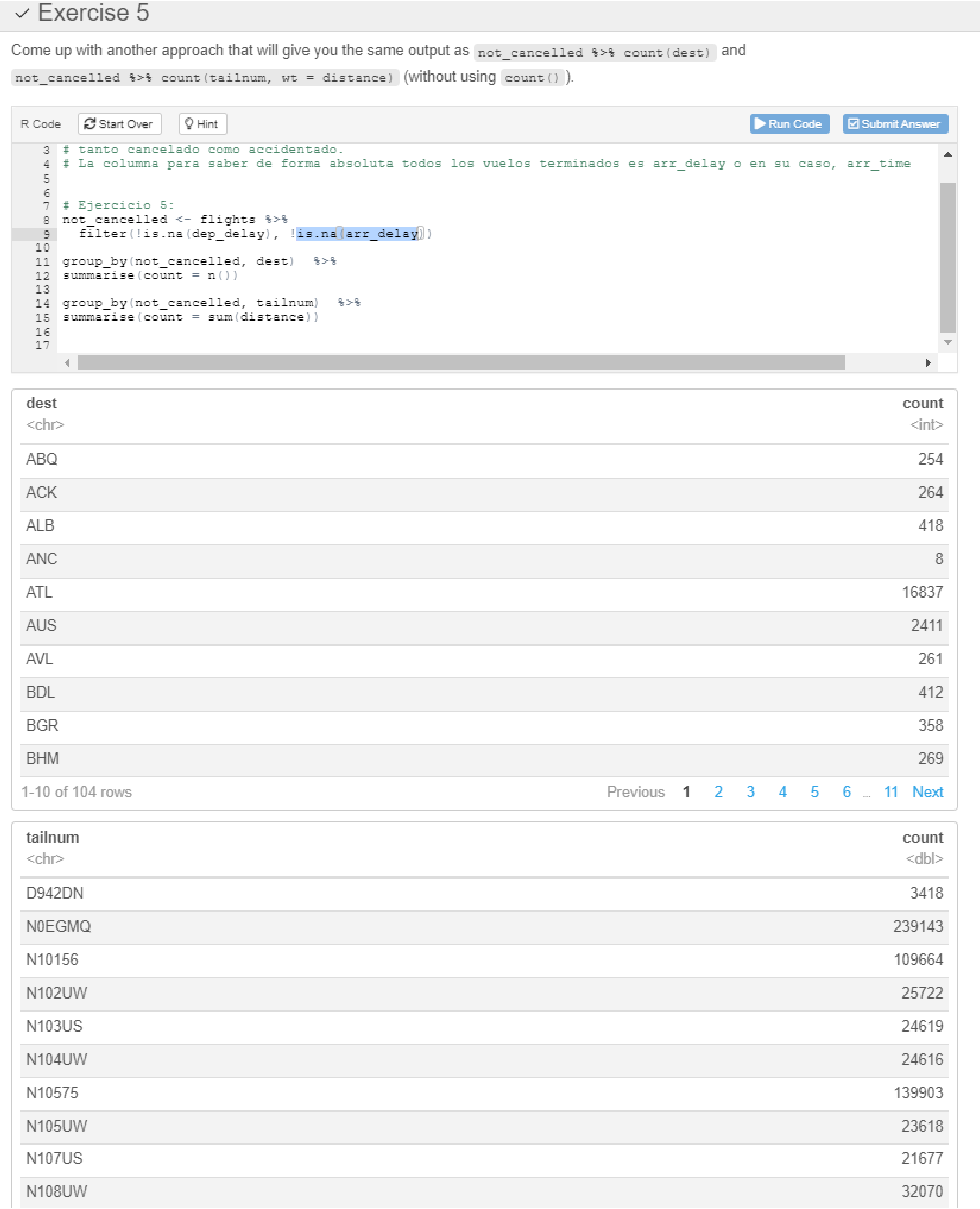
Which is more important: arrival delay or departure delay?



```r
group_by(flights, flight)  %>%
summarise(fif_mins_early = mean(arr_delay == -15, na.rm = TRUE),
fif_mins_late = mean(arr_delay == 15, na.rm = TRUE),
ten_mins_late = mean(arr_delay == 10, na.rm = TRUE),
thir_mins_early = mean(arr_delay == -30, na.rm = TRUE),
thir_mins_late = mean(arr_delay == 30, na.rm = TRUE),
no_delay = mean(arr_delay == 0, na.rm = TRUE),
two_hous_late = mean(arr_delay == 120, na.rm = TRUE))


```

flight	fif_mins_early	fif_mins_late	ten_mins_late	thir_mins_early	thir_mins_late	no_delay
<int>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	0.021520803	0.010043042	0.005738881	0.005738881	0.005738881	0.014347202
2	0.039215686	0.019607843	0.000000000	0.000000000	0.000000000	0.039215686
3	0.009554140	0.006369427	0.015923567	0.015923567	0.003184713	0.025477707
4	0.035805627	0.010230179	0.007672634	0.012787724	0.002557545	0.020460358
5	0.012345679	0.006172840	0.009259259	0.021604938	0.000000000	0.006172840
6	0.029126214	0.004854369	0.004854369	0.029126214	0.000000000	0.004854369
7	0.016949153	0.004237288	0.000000000	0.008474576	0.004237288	0.012711864
8	0.055555556	0.008547009	0.021367521	0.000000000	0.000000000	0.017094017
9	0.013157895	0.013157895	0.019736842	0.000000000	0.000000000	0.019736842
10	0.016393443	0.016393443	0.032786885	0.000000000	0.000000000	0.016393443

1-10 of 1,000 rows | 1-7 of 8 columns



✓ Exercise 5

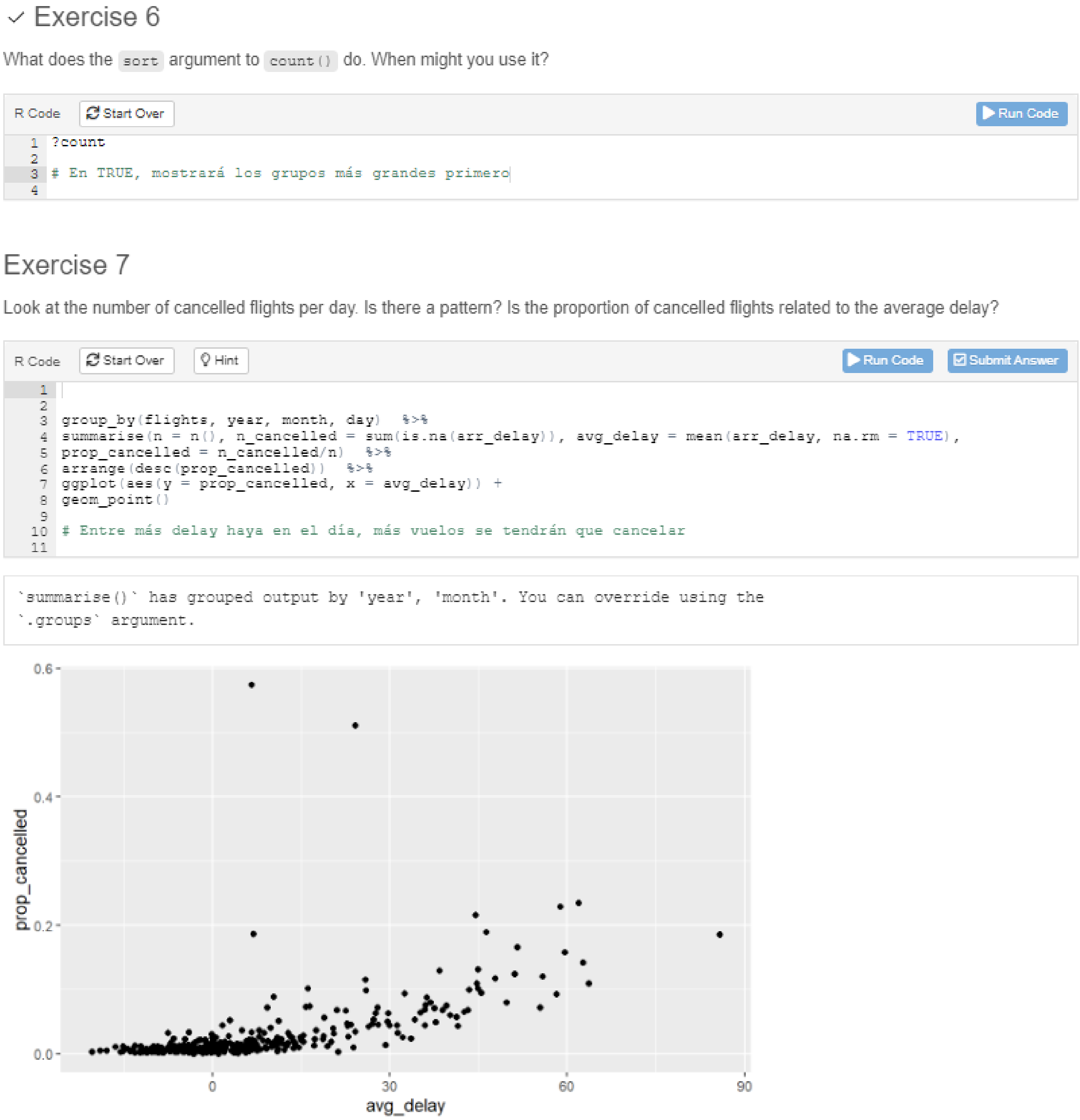
Come up with another approach that will give you the same output as `not_cancelled %>% count(dest)` and `not_cancelled %>% count(tailnum, wt = distance)` (without using `count()`).

```r
# tanto cancelado como accidentado.
# La columna para saber de forma absoluta todos los vuelos terminados es arr_delay o en su caso, arr_time


# Ejercicio 5:
not_cancelled <- flights %>%
  filter(!is.na(dep_delay), !is.na(arr_delay))

group_by(not_cancelled, dest)  %>%
summarise(count = n())

group_by(not_cancelled, tailnum)  %>%
summarise(count = sum(distance))


```

dest <chr>	count <int>
ABQ	254
ACK	264
ALB	418
ANC	8
ATL	16837
AUS	2411
AVL	261
BDL	412
BGR	358
BHM	269

1-10 of 104 rows Previous 1 2 3 4 5 6 ... 11 Next

tailnum <chr>	count <dbl>
D942DN	3418
N0EGMQ	239143
N10156	109664
N102UW	25722
N103US	24619
N104UW	24616
N10575	139903
N105UW	23618
N107US	21677
N108UW	32070



✓ Exercise 6

What does the `sort` argument to `count()` do. When might you use it?

```
?count

# En TRUE, mostrará los grupos más grandes primero

```

Exercise 7

Look at the number of cancelled flights per day. Is there a pattern? Is the proportion of cancelled flights related to the average delay?

```


group_by(flights, year, month, day)  %>%
summarise(n = n(), n_cancelled = sum(is.na(arr_delay)), avg_delay = mean(arr_delay, na.rm = TRUE),
prop_cancelled = n_cancelled/n)  %>%
arrange(desc(prop_cancelled))  %>%
ggplot(aes(y = prop_cancelled, x = avg_delay)) +
geom_point()

# Entre más delay haya en el día, más vuelos se tendrán que cancelar

```

```
`summarise()` has grouped output by 'year', 'month'. You can override using the
`.groups` argument.
```