Eddie Aguilar

Filter rows with filter()

✓ filter()

filter() lets you use a logical test to extract specific rows from a data frame. To use filter(), pass it the data frame followed by one or more logical tests.

So for example, we can use filter() to select every flight in flights that departed on January 1st. Click Run Code to give it a try:

R Code 1 fi 2 3	ØStar ilter(f)		, month == 1	, day == 1)					Run	Code
year <int></int>	mo <int></int>	d <int></int>	dep_time <int></int>	sched_dep_time <int></int>	dep_delay <dbl></dbl>	arr_time <int></int>	sched_arr_time <int></int>		carrier <chr></chr>)
2013	1	1	517	515	2	830	819	11	UA	
2013	1	1	533	529	4	850	830	20	UA	
2013	1	1	542	540	2	923	850	33	AA	
2013	1	1	544	545	-1	1004	1022	-18	В6	
2013	1	1	554	600	-6	812	837	-25	DL	
2013	1	1	554	558	-4	740	728	12	UA	
2013	1	1	555	600	-5	913	854	19	В6	
2013	1	1	557	600	-3	709	723	-14	EV	
2013	1	1	557	600	-3	838	846	-8	В6	
2013	1	1	558	600	-2	753	745	8	AA	
I-10 of	f 842 rov	/s 1-1	0 of 19 column	S			Previous 1 2 3	4 5 6	85	Ne

✓ output

Like all dplyr functions, [filter()] returns a new data frame for you to save or use. It doesn't overwrite the old data frame.

If you want to save the output of <code>filter()</code> , you'll need to use the assignment operator, <-

Rerun the command in the code chunk below, but first arrange to save the output to an object named [jan1].

R Code	₽ Star	t Over	♀ Solution						► Run Co	ode
1 ()j 2 3	na1 <-	filte	r(flights, n	month == 1, day == 1	1)[)]					
<int></int>	mo <int></int>		dep_time <int></int>	sched_dep_time <int></int>	dep_delay <dbl></dbl>	arr_time <int></int>	sched_arr_time <int></int>	arr_delay <dbl></dbl>	carrier <chr></chr>	٠
2013	1	1	517	515	2	830	819	11	UA	
2013	1	1	533	529	4	850	830	20	UA	
2013	1	1	542	540	2	923	850	33	AA	
2013	1	1	544	545	-1	1004	1022	-18	B6	
2013	1	1	554	600	-6	812	837	-25	DL	
2013	1	1	554	558	-4	740	728	12	UA	
2012	4	- 4	555	enn	E	042	0 E A	40	DC	

Logical Comparisons

Comparison operators

R provides a suite of comparison operators that you can use to compare values: >, >=, <, <=, != (not equal), and == (equal). Each creates a logical test. For example, is pi greater than three?

```
pi > 3

## [1] TRUE
```

When you place a logical test inside of filter(), filter applies the test to each row in the data frame and then returns the rows that pass, as a new data frame

Our code above returned every row whose month value was equal to one and whose day value was equal to one.

Watch out!

When you start out with R, the easiest mistake to make is to test for equality with = instead of == . When this happens you'll get an informative error:

```
filter(flights, month = 1)

## Error in `filter()`:

## ! We detected a named input.

## i This usually means that you've used `=` instead of `==`.

## i Did you mean `month == 1`?
```

Multiple tests

If you give <code>filter()</code> more than one logical test, <code>filter()</code> will combine the tests with an implied "and." In other words, <code>filter()</code> will return only the rows that return <code>TRUE</code> for every test. You can combine tests in other ways with Boolean operators...

Previous Topic

Next Topic

√ Test Your Knowledge

What will the following code return? <code>filter(flights, month == 11 | month == 12)</code>

Vevery flight that departed in November or December

Xevery flight that departed in November and December

Xevery flight except for those that departed in November or December

Xevery flight except for those that departed in November or December

Xevery flight except for those that departed in November or December

Xevery flight except for those that departed in November or December

Xevery flight except for those that departed in November or December

Xevery flight except for those that departed in November or December

Xevery flight except for those that departed in November or December

Xevery flight except for those that departed in November or December

Xevery flight except for those that departed in November or December

Xevery flight except for those that departed in November or December

Xevery flight except for those that departed in November or December

Xevery flight except for those that departed in November or December

Xevery flight except for those that departed in November or December

Xevery flight except for those that departed in November or December

Xevery flight except for those that departed in November or December

Common mistakes

In R, the order of operations doesn't work like English. You can't write | filter(flights, month == 11 | 12), even though you might say "finds all flights that departed in November or December". Be sure to write out a complete test on each side of a boolean operator.

Here are four more tips to help you use logical tests and Boolean operators in R:

1. A useful short-hand for this problem is 😨 * *in* *y . This will select every row where 😨 is one of the values in 💡 . We could use it to rewrite the code in the question above:

```
nov_dec <- filter(flights, month %in% c(11, 12))
```

2. Sometimes you can simplify complicated subsetting by remembering De Morgan's law: $\lfloor (\mathbf{x} \ \boldsymbol{\epsilon} \ \boldsymbol{y}) \rfloor$ is the same as $\lfloor (\mathbf{x} \ \boldsymbol{\epsilon} \ \boldsymbol{y}) \rfloor$ is the same as $\lfloor (\mathbf{x} \ \boldsymbol{\epsilon} \ \boldsymbol{y}) \rfloor$. For example, if you wanted to find flights that weren't delayed (on arrival or departure) by more than two hours, you could use either of the following two filters:

```
filter(flights, !(arr_delay > 120 | dep_delay > 120))
filter(flights, arr_delay <= 120, dep_delay <= 120)</pre>
```

- 3. As well as [s] and [], R also has ss and []. Don't use them with filter()! You'll learn when you should use them later.
- 4. Whenever you start using complicated, multipart expressions in [filter()], consider making them explicit variables instead. That makes it much easier to check your work. You'll learn how to create new variables shortly.

Previous Topic

Next Topic

Missing values

NA

Missing values can make comparisons tricky in R. R uses Na to represent missing or unknown values. Na s are "contagious" because almost any operation involving an unknown value (Na) will also be unknown (Na). For example, can you determine what value these expressions that use missing values should evaluate to? Make a prediction and then click "Submit Answer".

is.na()

The most confusing result above is this one:

```
NA == NA
## [1] NA
```

It's easiest to understand why this is true with a bit more context:

```
# Let x be Mary's age. We don't know how old she is.
x <- NA
# Let y be John's age. We don't know how old he is.
y <- NA
# Are John and Mary the same age?
x == y

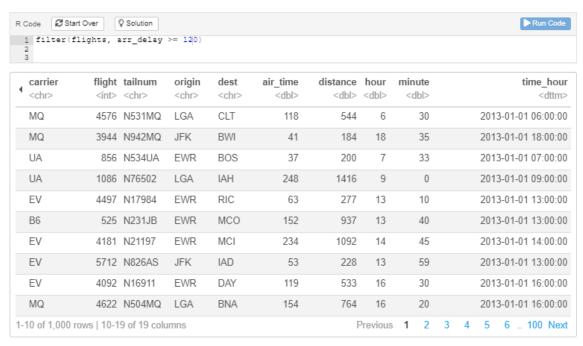
## [1] NA
# We don't know!</pre>
```

filter() and NAs

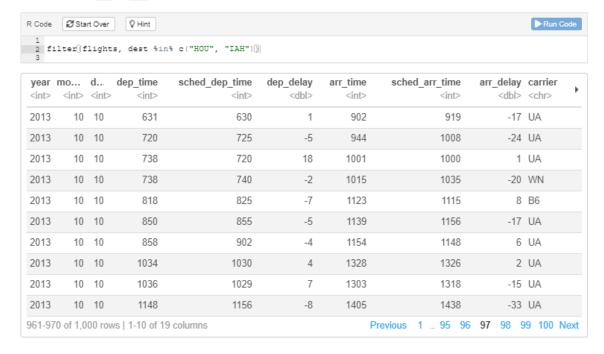
filter() only includes rows where the condition is TRUE; it excludes both FALSE and NA values. If you want to preserve missing values, ask for them explicitly:

Use the code chunks below to find all flights that

1. Had an arrival delay of two or more hours



2. Flew to Houston (IAH or HOU)

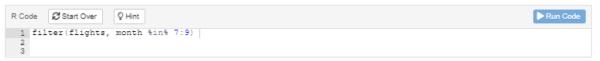


3. Were operated by United (UA), American (AA), or Delta (DL)



-	mo <int></int>		dep_time <int></int>	sched_dep_time <int></int>	dep_delay <dbl></dbl>	arr_time <int></int>	sched_arr_time <int></int>	arr_delay <dbl></dbl>	carrier <chr></chr>
2013	1	1	517	515	2	830	819	11	UA
2013	1	1	533	529	4	850	830	20	UA
2013	1	1	542	540	2	923	850	33	AA
2013	1	1	554	600	-6	812	837	-25	DL
2013	1	1	554	558	-4	740	728	12	UA
2013	1	1	558	600	-2	753	745	8	AA
2013	1	1	558	600	-2	924	917	7	UA
2013	1	1	558	600	-2	923	937	-14	UA
2013	1	1	559	600	-1	941	910	31	AA
2013	1	1	559	600	-1	854	902	-8	UA
1-10 of	f 1,000 r	ows	1-10 of 19 col	lumns		Pi	revious 1 2 3	4 5 6	100 Next

4. Departed in summer (July, August, and September)



year <int></int>	mo <int></int>		dep_time <int></int>	sched_dep_time <int></int>	dep_delay <dbl></dbl>	arr_time <int></int>	sched_arr_time <int></int>		carrier <chr></chr>
2013	7	1	1	2029	212	236	2359	157	B6
2013	7	1	2	2359	3	344	344	0	B6
2013	7	1	29	2245	104	151	1	110	B6
2013	7	1	43	2130	193	322	14	188	B6
2013	7	1	44	2150	174	300	100	120	AA
2013	7	1	46	2051	235	304	2358	186	B6
2013	7	1	48	2001	287	308	2305	243	VX
2013	7	1	58	2155	183	335	43	172	B6
2013	7	1	100	2146	194	327	30	177	B6
2013	7	1	100	2245	135	337	135	122	B6
1-10 of	f 1,000 r	ows	1-10 of 19 colu	ımns		Р	revious 1 2 3	4 5 6	100 Next

5. Arrived more than two hours late, but didn't leave late



year <int></int>		d <int></int>	dep_time <int></int>	sched_dep_time <int></int>	dep_delay <dbl></dbl>	arr_time <int></int>	sched_arr_time <int></int>	arr_delay <dbl></dbl>		•
2013	10	7	1350	1350	0	1736	1526	130	EV	
2013	5	23	1810	1810	0	2208	2000	128	MQ	
2013	7	1	905	905	0	1443	1223	140	DL	
	i 1-10 c			303	0	1443	1223	140	DL	

6. Were delayed more than an hour, but made up more than 30 minutes in flight



year <int></int>	month <int></int>	day <int></int>	dep_time <int></int>	sched_dep_time <int></int>	dep_delay <dbl> ▶</dbl>
2013	1	1	2205	1720	285
2013	1	1	2326	2130	116
2013	1	3	1503	1221	162
2013	1	3	1839	1700	99
2013	1	3	1850	1745	65
2013	1	3	1941	1759	102
2013	1	3	1950	1845	65
2013	1	3	2257	2000	177
2013	1	4	1917	1700	137
2013	1	4	2010	1745	145
1-10 of 1,000 rov	ws 1-6 of 1	9 columns		Previous 1 2 3 4	5 6 100 Next

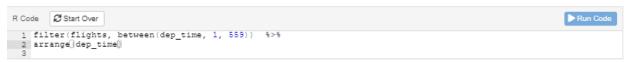
7. Departed between midnight and 6am (inclusive)



year <int></int>	month <int></int>	day <int></int>	dep_time <int></int>	sched_dep_time <int></int>	dep_delay <dbl></dbl>
2013	4	19	121	1940	341
2013	4	22	121	2245	156
2013	6	28	121	1659	502
2013	7	20	121	2359	82
2013	2	23	122	2359	83
2013	2	23	122	2059	263
2013	2	27	122	2250	152
2013	2	27	122	2358	84
2013	3	18	122	2245	157
2013	1	25	123	2000	323
991-1000 of 1,0	00 rows 1-6	of 19 colur	mns	Previous 1 95 96 97	98 99 100 Next

√ Exercise 2

Another useful dplyr filtering helper is between(). What does it do? Can you use between() to simplify the code needed to answer the previous challenges?



year <int></int>	month <int></int>	day <int></int>	dep_time <int></int>	sched_dep_time <int></int>	dep_delay <dbl></dbl>	arr_time <int></int>
2013	4	19	121	1940	341	314
2013	4	22	121	2245	156	230
2013	6	28	121	1659	502	329
2013	7	20	121	2359	82	506
2013	2	23	122	2359	83	554
2013	2	23	122	2059	263	345
2013	2	27	122	2250	152	227
2013	2	27	122	2358	84	600
2013	3	18	122	2245	157	226

√ Exercise 3

How many flights have a missing dep time? What other variables are missing? What might these rows represent?

√ Exercise 4

Why is NA $^{\circ}$ 0 not missing? Why is NA | TRUE not missing? Why is FALSE & NA not missing? Can you figure out the general rule? (NA $^{\circ}$ 0 is a tricky counterexample!)

```
R Code Start Over Hints

1 # Si NA es finito, la multiplicación será 0, pero si es finito, la multiplicación
2 # será NaN, al no saber que es Na, no se puede saber si la multiplicación
3 # será NaN o 0, por lo tanto el resultado también es NA.

[1] NA
```

Previous Topic