Manipulate!

Hadley Wickham
@hadleywickham
Chief Scientist, RStudio

July 2015

Outline

- Warmups
- Flights data
- One table verbs
- Data pipelines

Warmups

Warmups

What are the seven most common types of variable found in a data frame?

Your turn

What are the most useful summary functions in R? (i.e. functions that take a vector of n values and return a single value).

Summary functions

- `mean(x), median(x)`
- `n(), n_distinct(x)`
- `sd(x), IQR(x), mad(x)`
- `min(x), max(x), quantile(x, p)`
- `first(x), last(x), nth(x, i)`

Warmups

What does the sum of a logical vector tell you? What about the mean?

```r
x <- c(TRUE, FALSE, FALSE, TRUE)
as.numeric(x)

sum(x)
mean(x) # = sum(x) / length(x)
```

Warmups

What do the following expressions return?

```
NA + 5
```

```
10 < NA
```

```
10 == NA
```

```
NA == NA
```

Why?

```
# NAs are tricky!
NA + 5
10 * NA

10 < NA
10 == NA
NA == NA
is.na(NA)
```

Name	Age	Sex
John	35	M
Mary	NA	F
Sam	NA	NA

Is Mary the same age as Jaime?
Are Sam's age and sex the same?

Flights data

```
library(dplyr)
library(nycflights13)

# Every flight departing New York City in 2013
flights
str(flights)
View(flights)
```

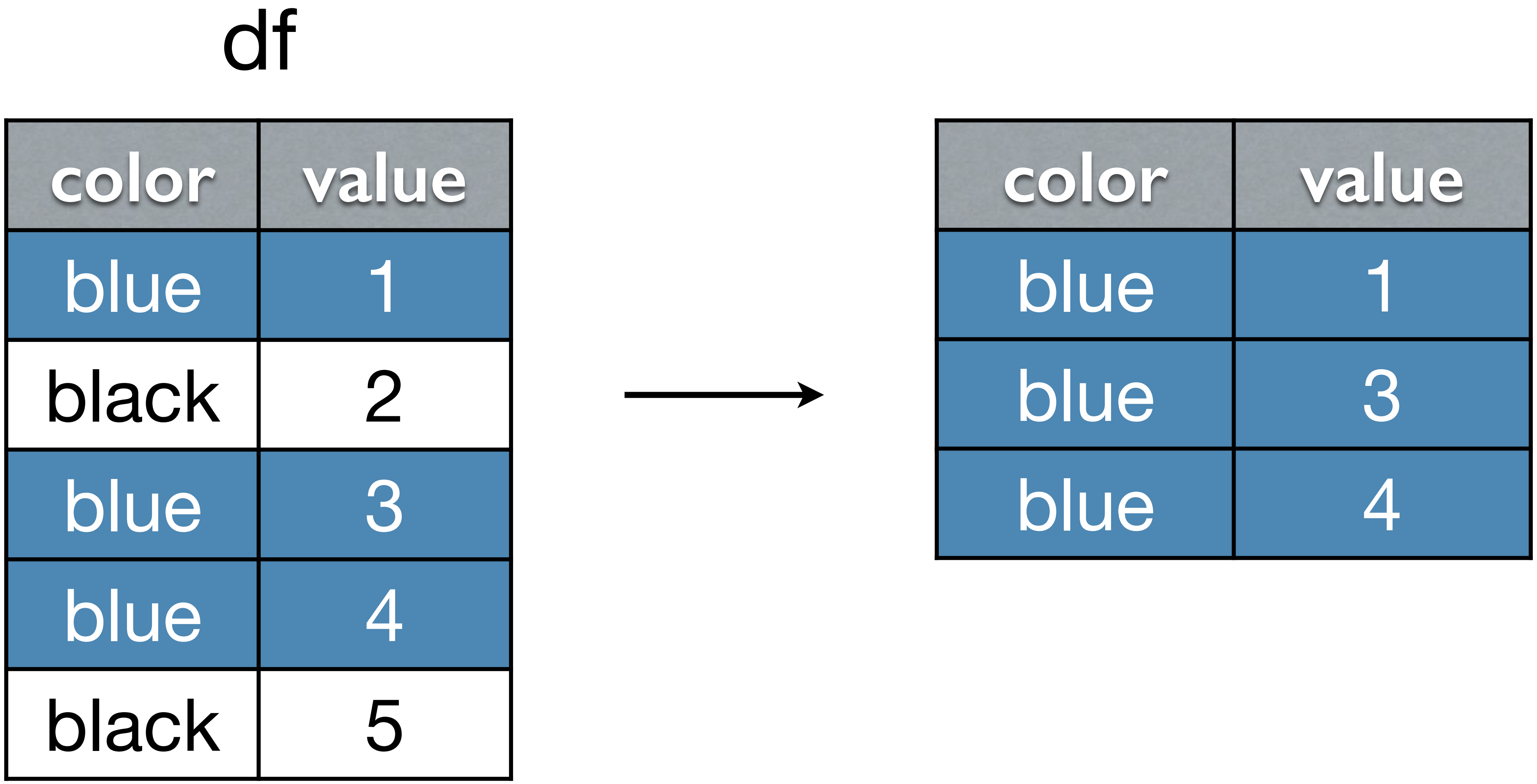
One table verbs

- **filter**: keep observations matching criteria
- **select**: pick variables by name
- **arrange**: reorder observations
- **mutate**: add new variables
- **summarise**: reduce many values to 1

Structure

- First argument is a data frame
- Subsequent arguments say what to do with data frame
- Always return a data frame
- Never modify in place!

```r
df <- data.frame(
  color = c("blue", "black", "blue", "blue", "black"),
  value = 1:5
)
```

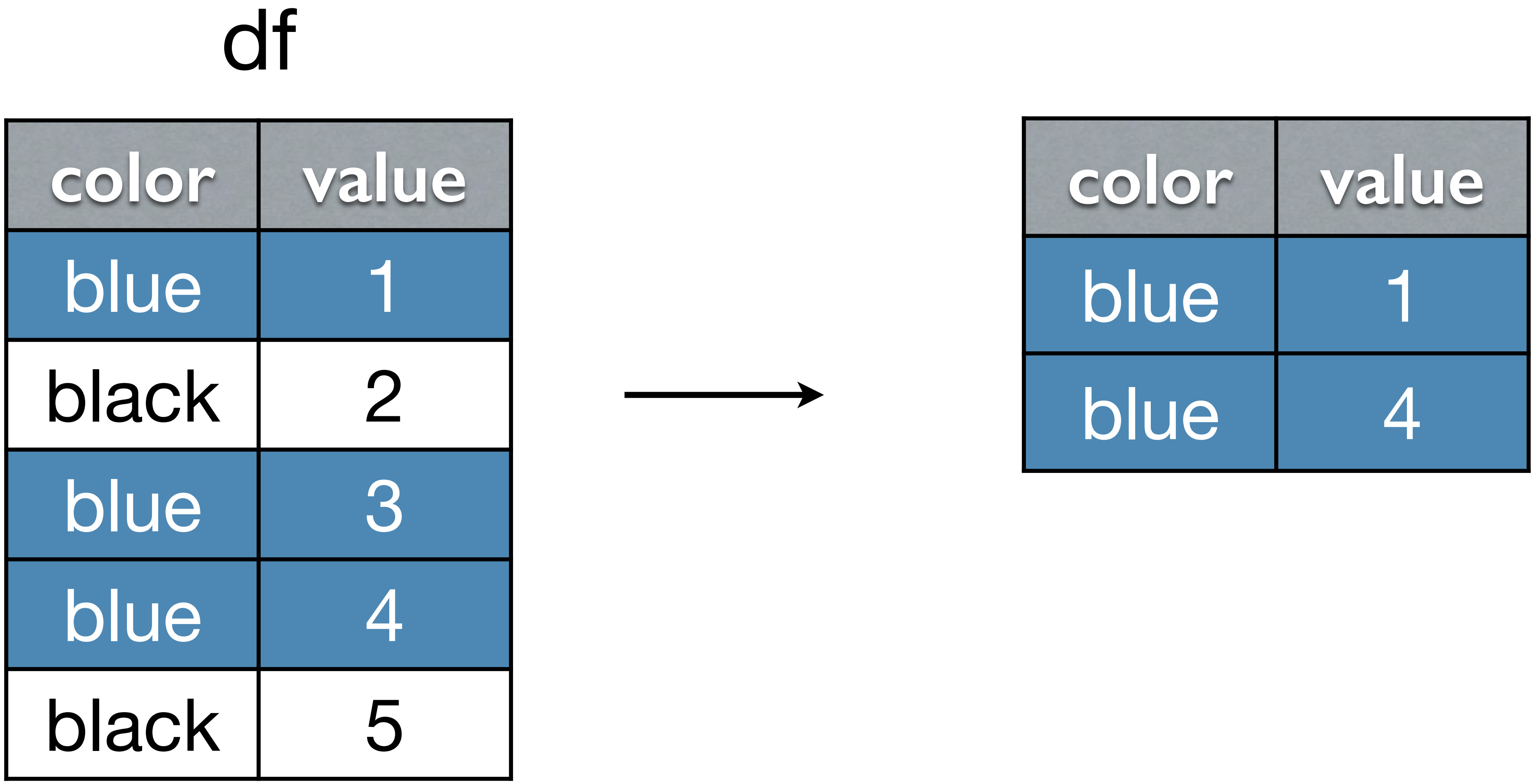


df

color	value
blue	1
black	2
blue	3
blue	4
black	5

color	value
blue	1
blue	3
blue	4

```
filter(df, color == ”blue”)
```



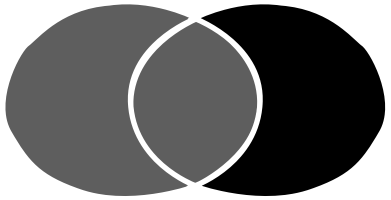
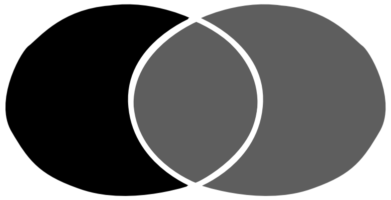
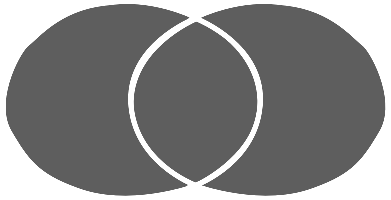
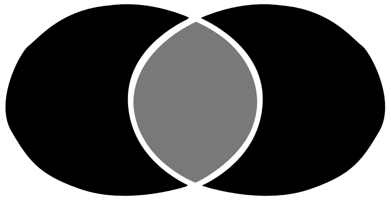
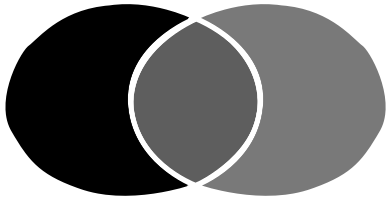
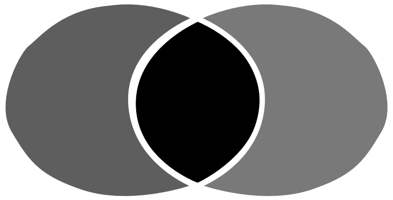
df

color	value
blue	1
black	2
blue	3
blue	4
black	5

→

color	value
blue	1
blue	4

```
filter(df, value %in% c(1, 4))
```

	a
	b
	a | b
	a & b
	a & !b
	xor(a, b)

```
x > 1
x >= 1
x < 1
x <= 1
x != 1
x == 1
x %in% ("a", "b")
```

```
# Just prints out results
filter(flights, dest %in% c(”IAH”, ”HOU”))
# The original is unchanged:
flights

# To create a new variable use <-
houston <- filter(flights, dest %in% c(”IAH”, ”HOU”))
houston

# BE CAREFUL!
flights <- filter(flights, dest %in% c(”IAH”, ”HOU”))
```

Find all flights:

To SFO or OAK

In January

Delayed by more than an hour

That departed between midnight and five am.

That departed before 5am or after 10pm?

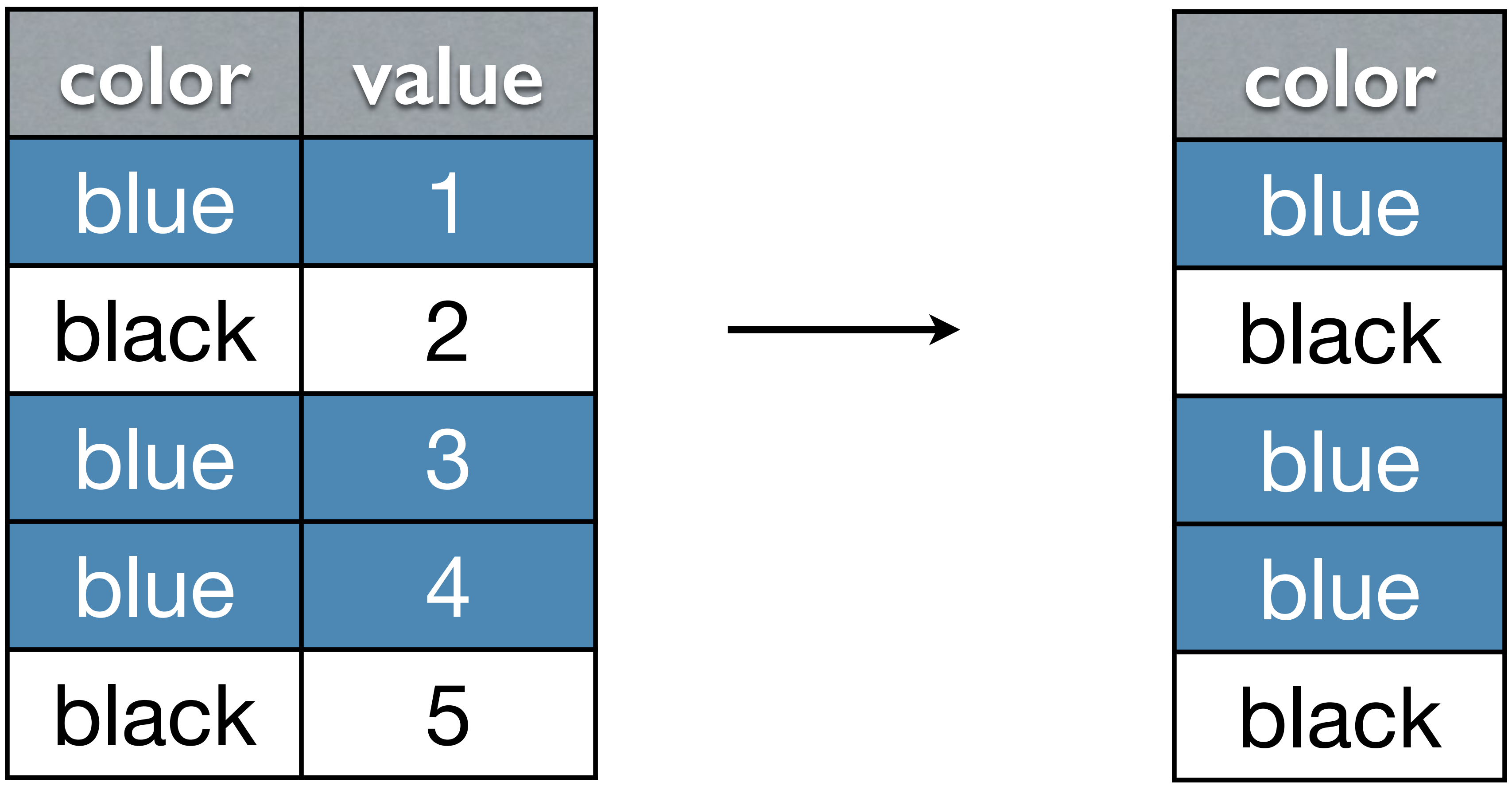
Where the arrival delay was more than twice the departure delay

```
filter(flights, dest %in% c(”SFO”, ”OAK”))
filter(flights, dest == ”SFO” | dest == ”OAK”)
# Not this!
filter(flights, dest == ”SFO” | ”OAK”)

filter(flights, month == 1)

filter(flights, hour >= 0 & hour <= 5)
filter(flights, hour >= 0, hour <= 5)
filter(flights, hour <= 5 | hour >= 22)

filter(flights, arr_delay > 2 * dep_delay)
```



df

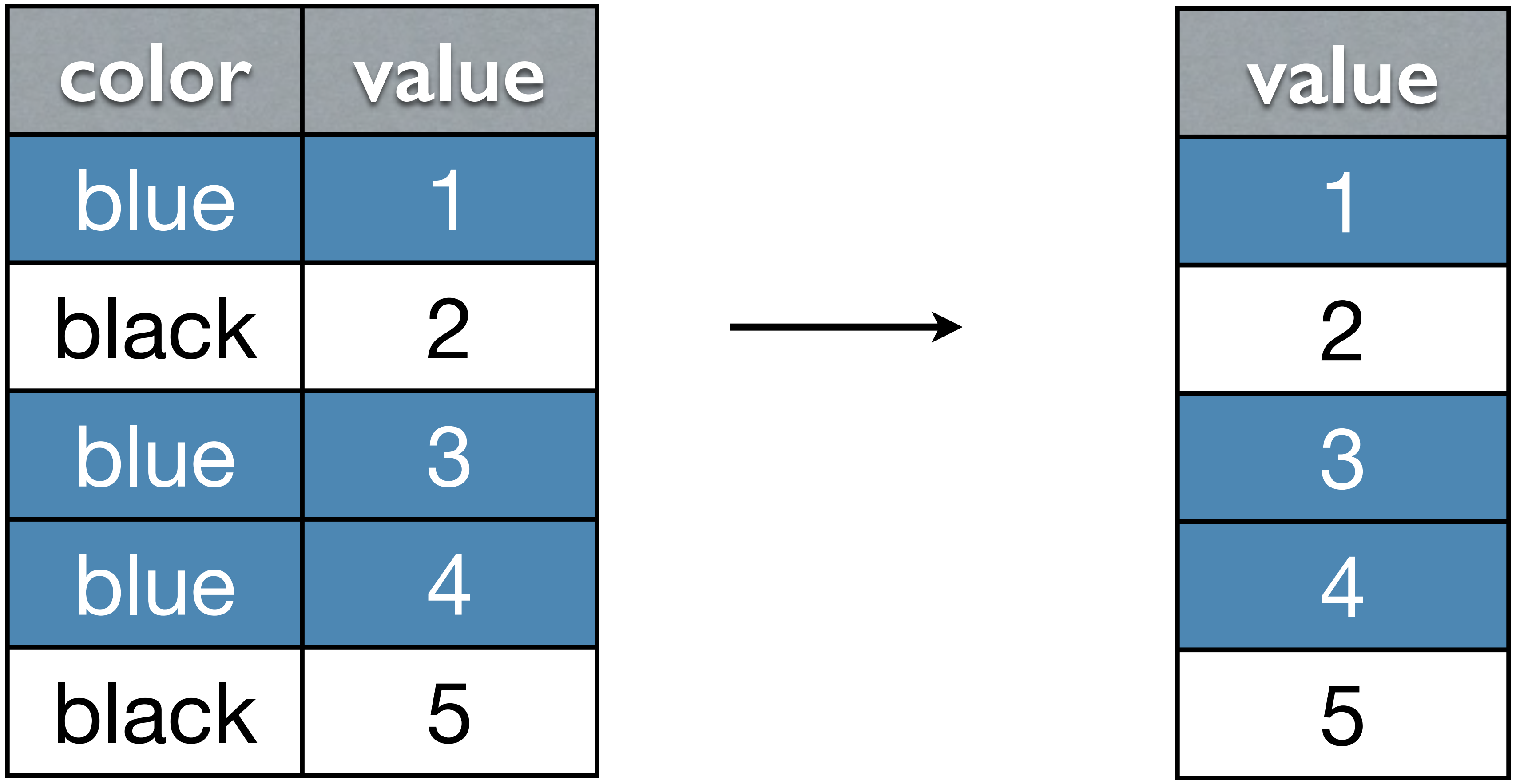
color	value
blue	1
black	2
blue	3
blue	4
black	5

→

color
blue
black
blue
blue
black

```
select(df, color)
```

df



color	value
blue	1
black	2
blue	3
blue	4
black	5

value
1
2
3
4
5

```
select(df, -color)
```

Your turn

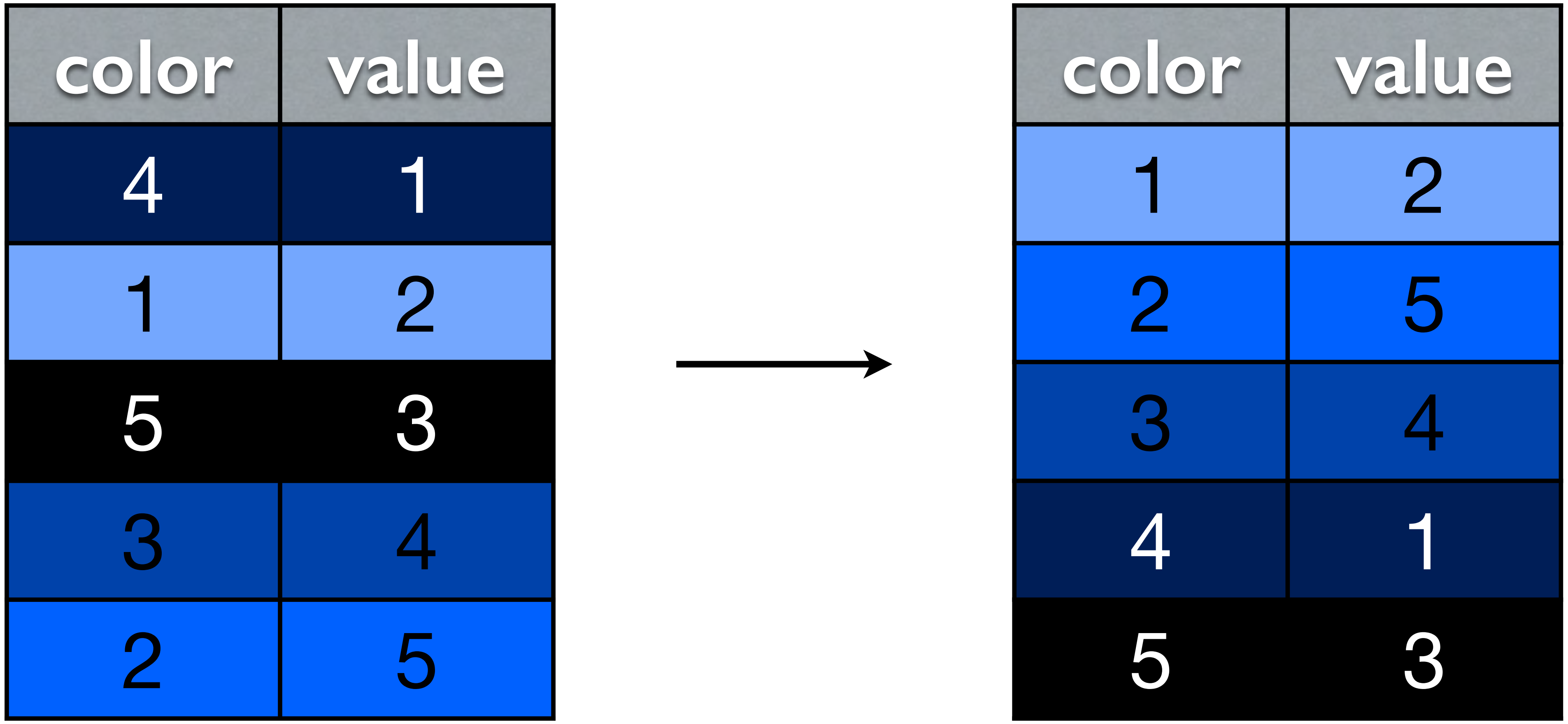
Read the help for `select()`. What other ways can you select variables?

Write down three ways to select the two delay variables.

```
select(flights, arr_delay, dep_delay)
select(flights, c(arr_delay, dep_delay))
select(flights, dep_delay, dep_delay + 2)
select(flights, ends_with(”delay”))
select(flights, contains(”delay”))

x <- c(”arr_delay”, ”dep_delay”)
select(flights, one_of(x))
```

df

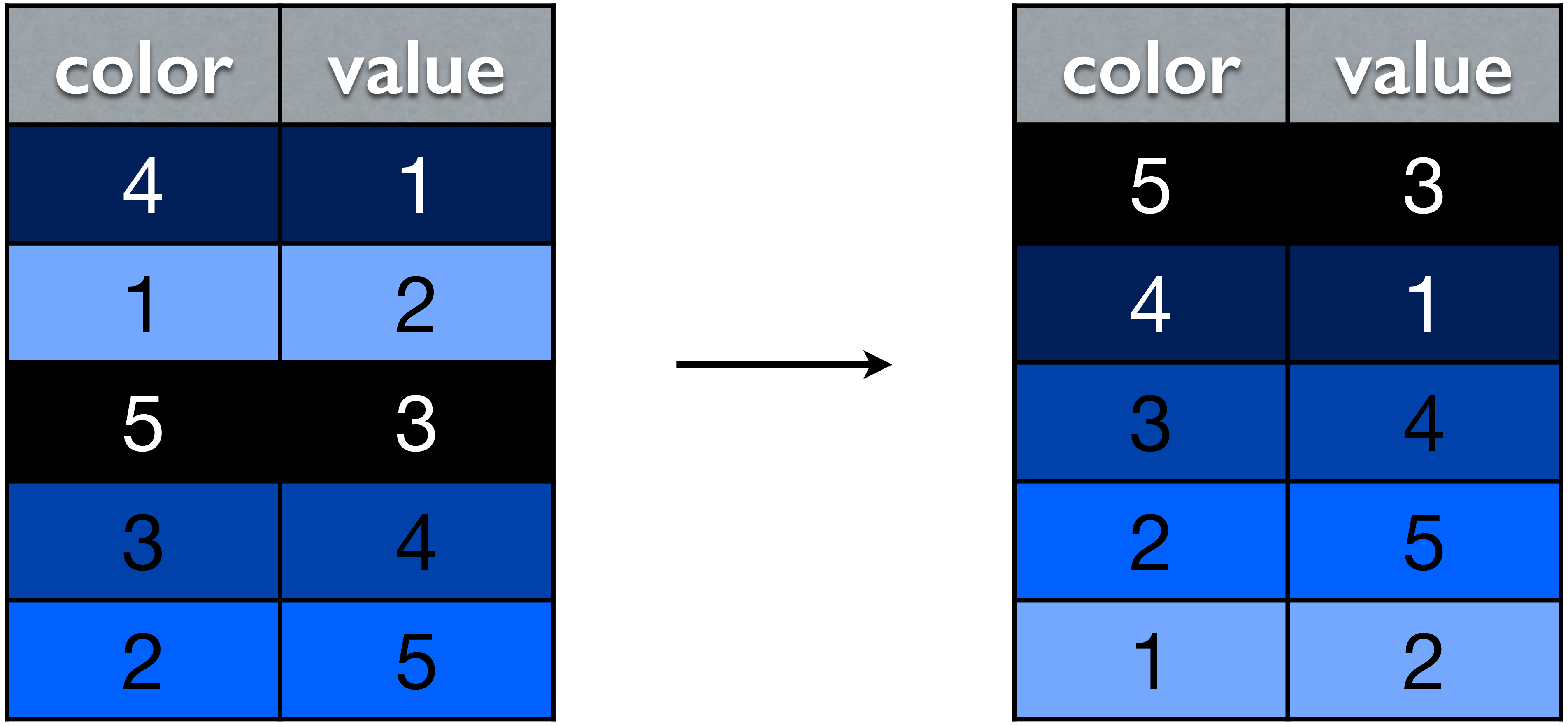


color	value
4	1
1	2
5	3
3	4
2	5

color	value
1	2
2	5
3	4
4	1
5	3

```
arrange(df, color)
```

df



color	value
4	1
1	2
5	3
3	4
2	5

→

color	value
5	3
4	1
3	4
2	5
1	2

```
arrange(df, desc(color))
```

Your turn

Order the flights by departure date and time.

Which flights were most delayed?

Which flights caught up the most time during the flight?

```
arrange(flights, month, day, hour, minute)

arrange(flights, desc(dep_delay))
arrange(flights, desc(arr_delay))

arrange(flights, desc(dep_delay - arr_delay))
```

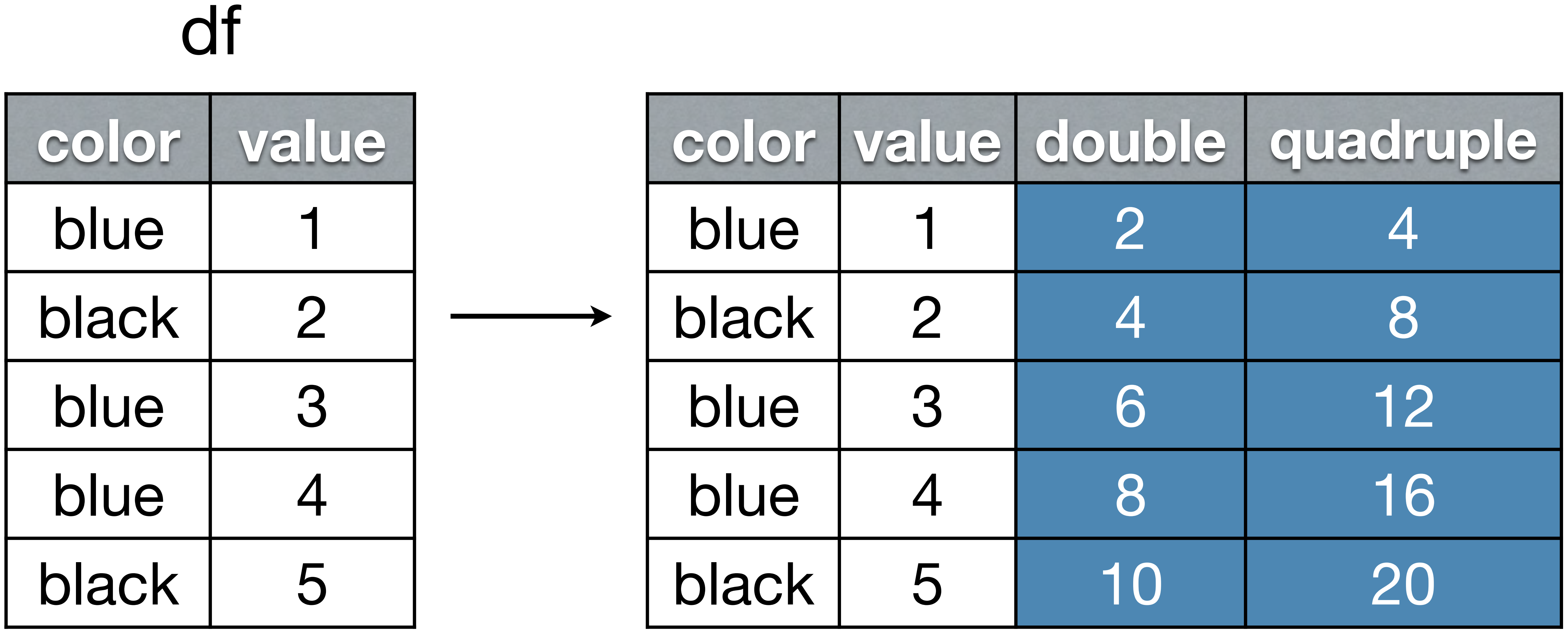
df

color	value
blue	1
black	2
blue	3
blue	4
black	5

→

color	value	double
blue	1	2
black	2	4
blue	3	6
blue	4	8
black	5	10

```
mutate(df, double = 2 * value)
```



df

color	value
blue	1
black	2
blue	3
blue	4
black	5

color	value	double	quadruple
blue	1	2	4
black	2	4	8
blue	3	6	12
blue	4	8	16
black	5	10	20

```
mutate(df, double = 2 * value,
   quadruple = 2 * double)
```

Your turn

Compute speed in mph from `air_time` (in minutes) and `distance` (in miles). Which flight flew the fastest?

Add a new variable that shows how much time was made up or lost in flight.

How did I compute `hour` and `minute` from `dep_time`?

(Hint: you might need to use `View()` to see the new variables)

```
flights <- mutate(flights,
  speed = dist / (time / 60))
arrange(flights, desc(speed))

mutate(flights, delta = dep_delay - arr_delay)

mutate(flights,
  hour = dep_time %/% 100,
  minute = dep_time %% 100)
```

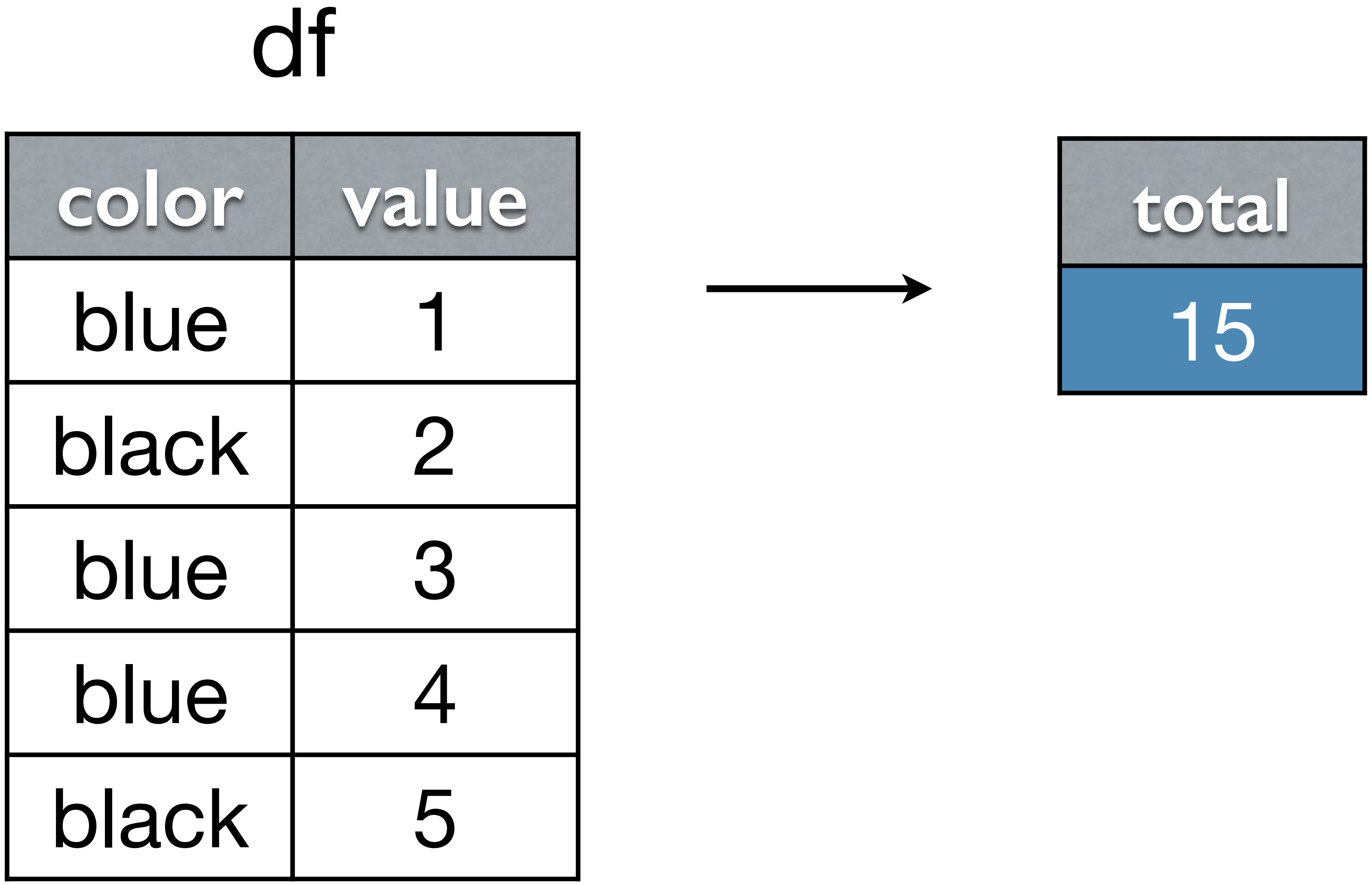
Your turn

What do `rename()` and `transmute()` do? How are they different to `select()` and `mutate()`?

Use the help (`?rename`, `?transmute`) to find out.

	All variables	Only mentioned
Select	`rename()`	`select()`
Modify	`mutate()`	`transmute()`

Grouped summarise



df

color	value
blue	1
black	2
blue	3
blue	4
black	5

total
15

```
summarise(df, total = sum(value))
```

df

color	value
blue	1
black	2
blue	3
blue	4
black	5

→

color	total
blue	8
black	7

```
by_color <- group_by(df, color)
summarise(by_color, total = sum(value))
```

```
by_date <- group_by(flights, month, day)
by_hour <- group_by(flights, month, day, hour)
by_plane <- group_by(flights, tailnum)
by_dest <- group_by(flights, dest)
```

Summary functions

- `min(x), median(x), max(x), quantile(x, p)`
- `n(), n_distinct(x), sum(x), mean(x)`
- `(x > 10)`
- `sd(x), var(x), IQR(x), mad(x)`

Always include when summarising!

```
by_date <- group_by(flights, month, day)
delays <- summarise(by_date,
  mean = mean(dep_delay),
  median = median(dep_delay),
  q75 = quantile(dep_delay, 0.75),
  over_15 = mean(dep_delay > 15),
  over_30 = mean(dep_delay > 30),
  over_60 = mean(dep_delay > 60)
 )
```

```
by_date <- group_by(flights, date)
delays <- summarise(by_date,
  mean = mean(dep_delay, na.rm = TRUE),
  median = median(dep_delay, na.rm = TRUE),
  q75 = quantile(dep_delay, 0.75, na.rm = TRUE),
  over_15 = mean(dep_delay > 15, na.rm = TRUE),
  over_30 = mean(dep_delay > 30, na.rm = TRUE),
  over_60 = mean(dep_delay > 60, na.rm = TRUE)
 )
```

```
# OR

by_date <- group_by(flights, date)
no_missing <- filter(flights, !is.na(dep_delay))
delays <- summarise(no_missing,
 mean = mean(dep_delay),
  median = median(dep_delay),
  q75 = quantile(dep_delay, 0.75),
  over_15 = mean(dep_delay > 15),
  over_30 = mean(dep_delay > 30),
  over_60 = mean(dep_delay > 60)
 )
```

Your turn

Which hour has the highest average delay?

How many flights does each plane make? (tail number)

Compute the average (?) distance to each destination.

Data pipelines

```
# Downside of functional interface is that it's
# hard to read multiple operations:
hourly_delay <- filter(
  summarise(
    group_by(
      filter(
        flights,
        !is.na(dep_delay)
      ),
      date, hour
    ),
    delay = mean(dep_delay),
    n = n()
  ),
  n > 10
)
```

```
# Solution: the pipe operator from magrittr
# x %>% f(y) -> f(x, y)

x %>% f(y)
# f(x, y)

x %>% f(z, .)
# f(z, x)

x %>% f(y) %>% g(z)
# g(f(x, y), z)

# Turns function composition (hard to read)
# into sequence (easy to read)
```

```r
foo_foo <- little_bunny()

foo_foo %>%
  hop_through(forest) %>%
  scoop_up(field_mouse) %>%
  bop_on(head)

# vs
bop_on(
  scoop_up(
    hop_through(foo_foo, forest),
    field_mouse
  ),
  head
)
```

```r
hourly_delay <- flights %>%
  filter(!is.na(dep_delay)) %>%
  group_by(date, hour) %>%
  summarise(delay = mean(dep_delay), n = n()) %>%
  filter(n > 10)

# Hint: pronounce %>% as then
```

Your turn

Create data pipelines to answer the following questions:

Which destinations have the highest average delays?

On average, how do delays (of non-cancelled flights) vary over the course of a day? (Hint: `time = hour + minute / 60`). Use a plot!

```
flights %>%
  group_by(dest) %>%
  summarise(
    arr_delay = mean(arr_delay, na.rm = TRUE),
    n = n()) %>%
  arrange(desc(arr_delay))

# It would be nice to plot these on a map...
```

```
per_hour <- flights %>%
 mutate(time = hour + minute / 60) %>%
  group_by(time) %>%
  summarise(
    arr_delay = mean(arr_delay, na.rm = TRUE),
    n = n()
  )

qplot(time, arr_delay, data = per_hour)
qplot(time, arr_delay, data = per_hour, size = n) +
  scale_size_area()
qplot(time, arr_delay, data = filter(per_hour, n > 30),
  size = n) + scale_size_area()

ggplot(filter(per_hour, n > 30), aes(time, arr_delay)) +
  geom_vline(xintercept = 5:24, colour = ”white”, size = 2) +
  geom_point()
```

Where next

```
browseVignettes(package = ”dplyr”)
# 1. two-table verbs
# 2. databases
# 3. do()

# RStudio cheatsheets:
# http://rstudio.com/cheatsheets

# Common questions & answers
http://stackoverflow.com/questions/tagged/dplyr?
sort=frequent
```