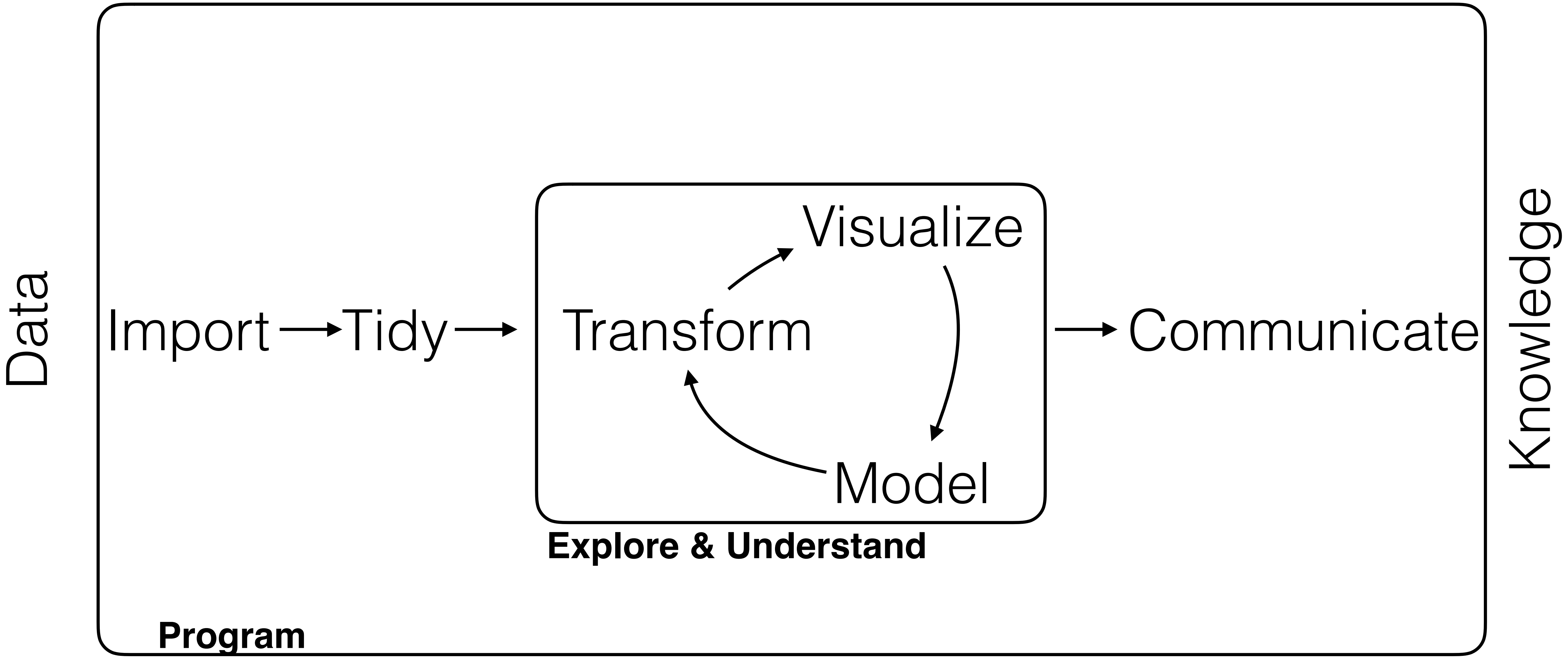
Transform Data

Strategic and Competitive Intelligence

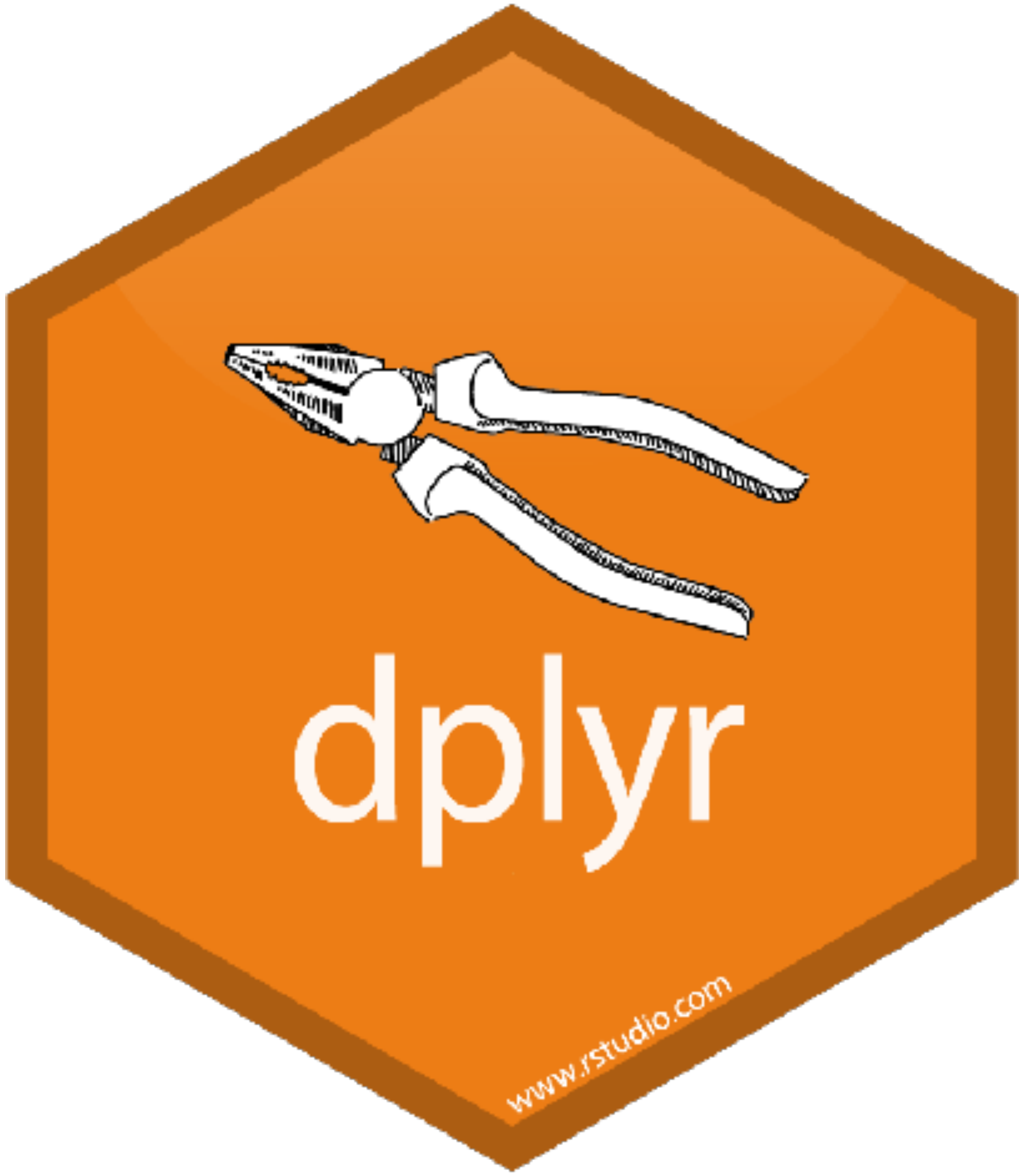
Filippo Chiarello

filippochiarello.90@gmail.com

What you will learn

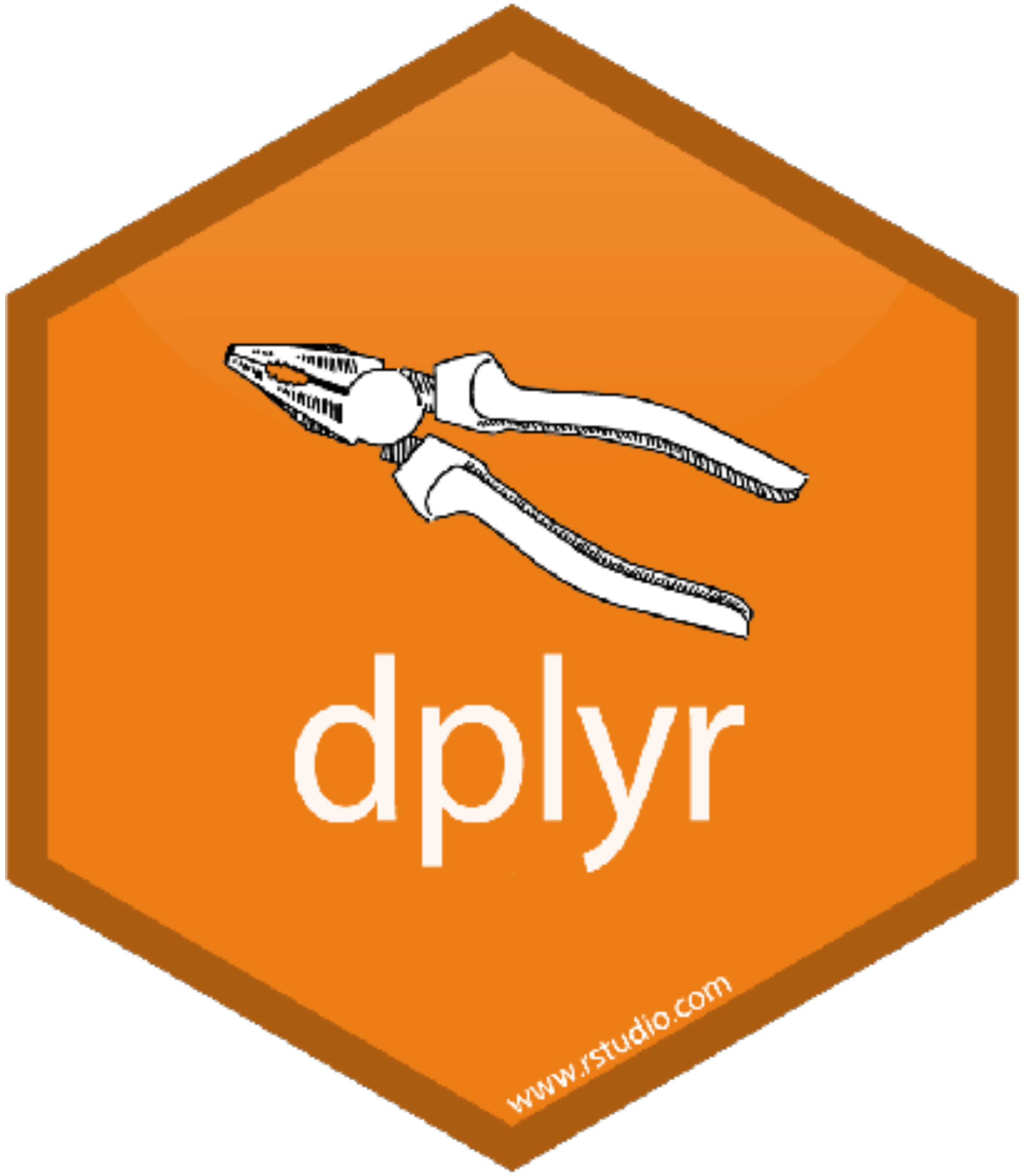


Adapt from: “R for Data Science”, H. Wickham, 2017



dplyr is a ***functional grammar of data manipulation***

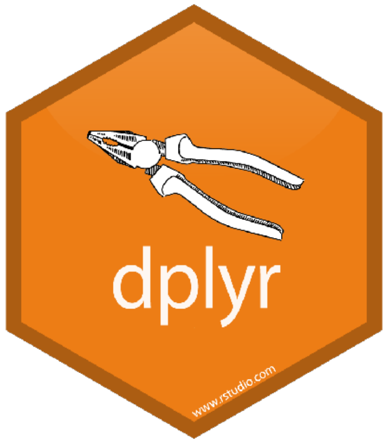
Which **actions** can be done on data?



dplyr

Makes data manipulation easy:

- By **constraining** your options, it helps you think about your data manipulation challenges.
- It provides **simple verbs**, functions that correspond to the most common data manipulation tasks
- Help you **translate** your thoughts into code



filter: Pick observations by their values

arrange: Reorder the rows

select: Pick variables by their names

mutate: Create new variables (functions of existing variables)

summarise: Collapse many values down to a single summary

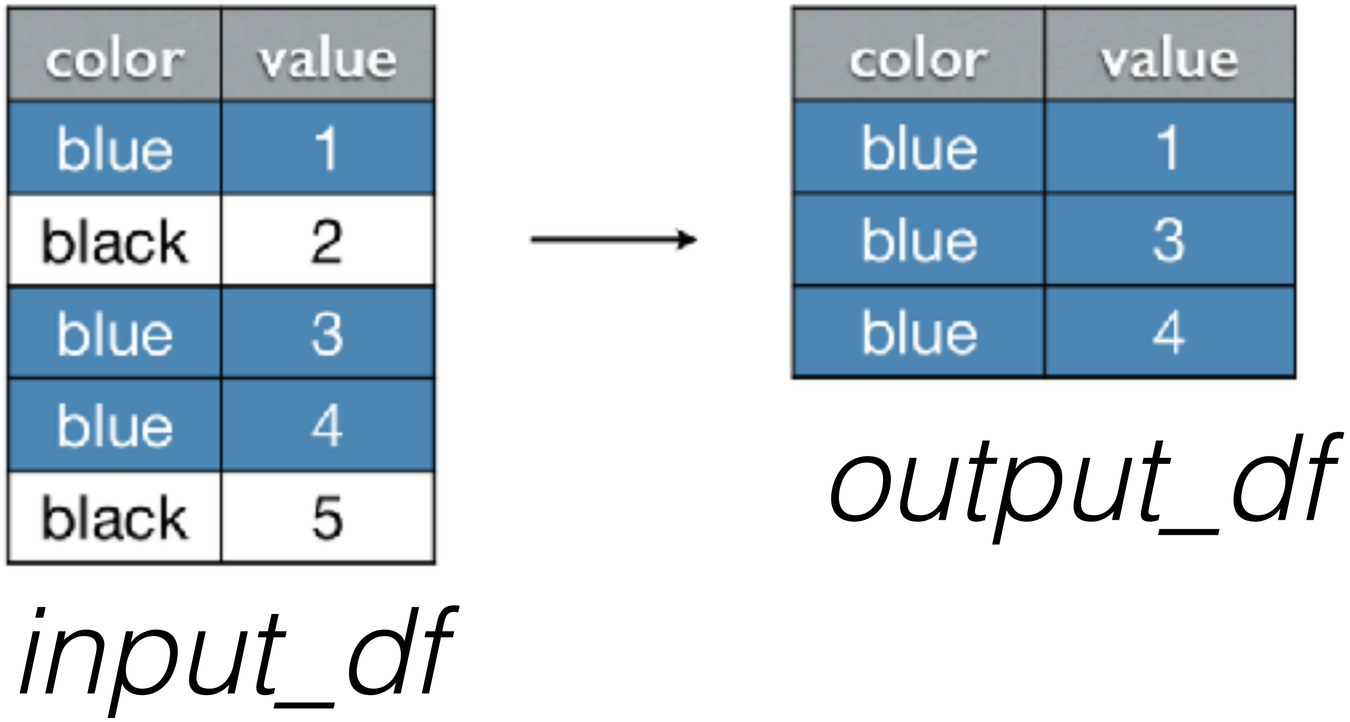
group by:changes the scope of each verb to operating on groups of observations

All verbs work similarly:

1 - The first argument is a data frame.
2 - The subsequent arguments gives specification about the verbs.
3 - The result is a new data frame.

$$\overset{3}{\mathit{output\_df}} <\!- \mathbf{verb}(\overset{1}{\mathit{input\_df}}, \overset{2}{\mathit{arguments}})$$

output_df <– **filter**(*input_df*, *color==“blue”*)



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

input_df

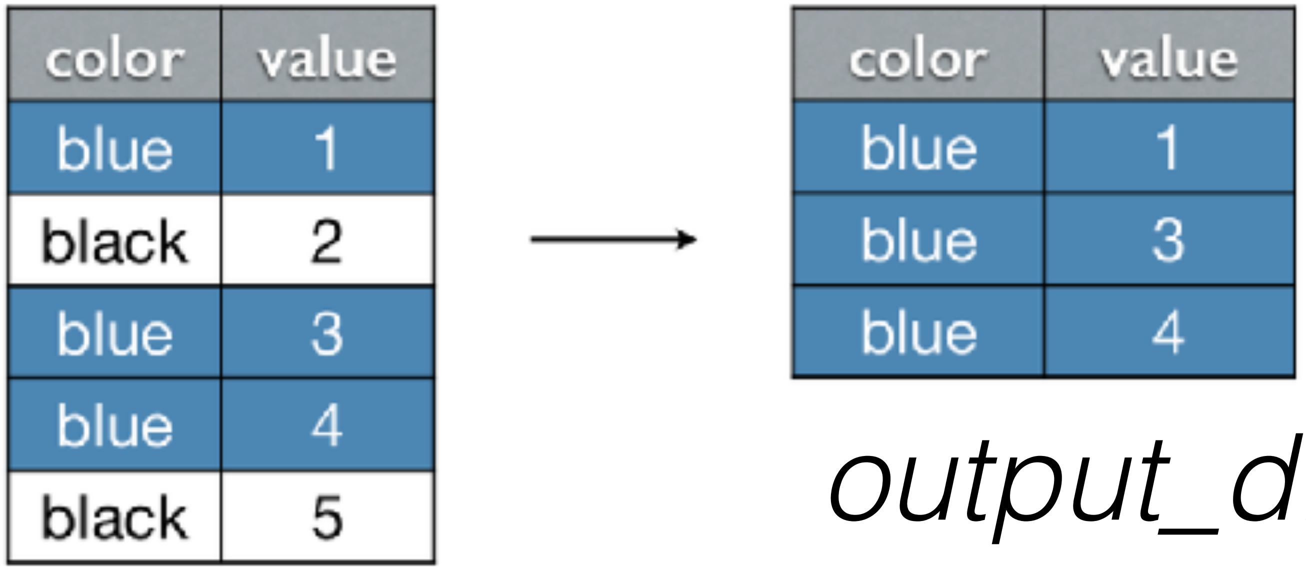
→

color	value
blue	1
blue	3
blue	4

output_df

Subset observations based on their values *output_df* <– **filter**(*input_df*, *color==“blue”*)

Any logical operation



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

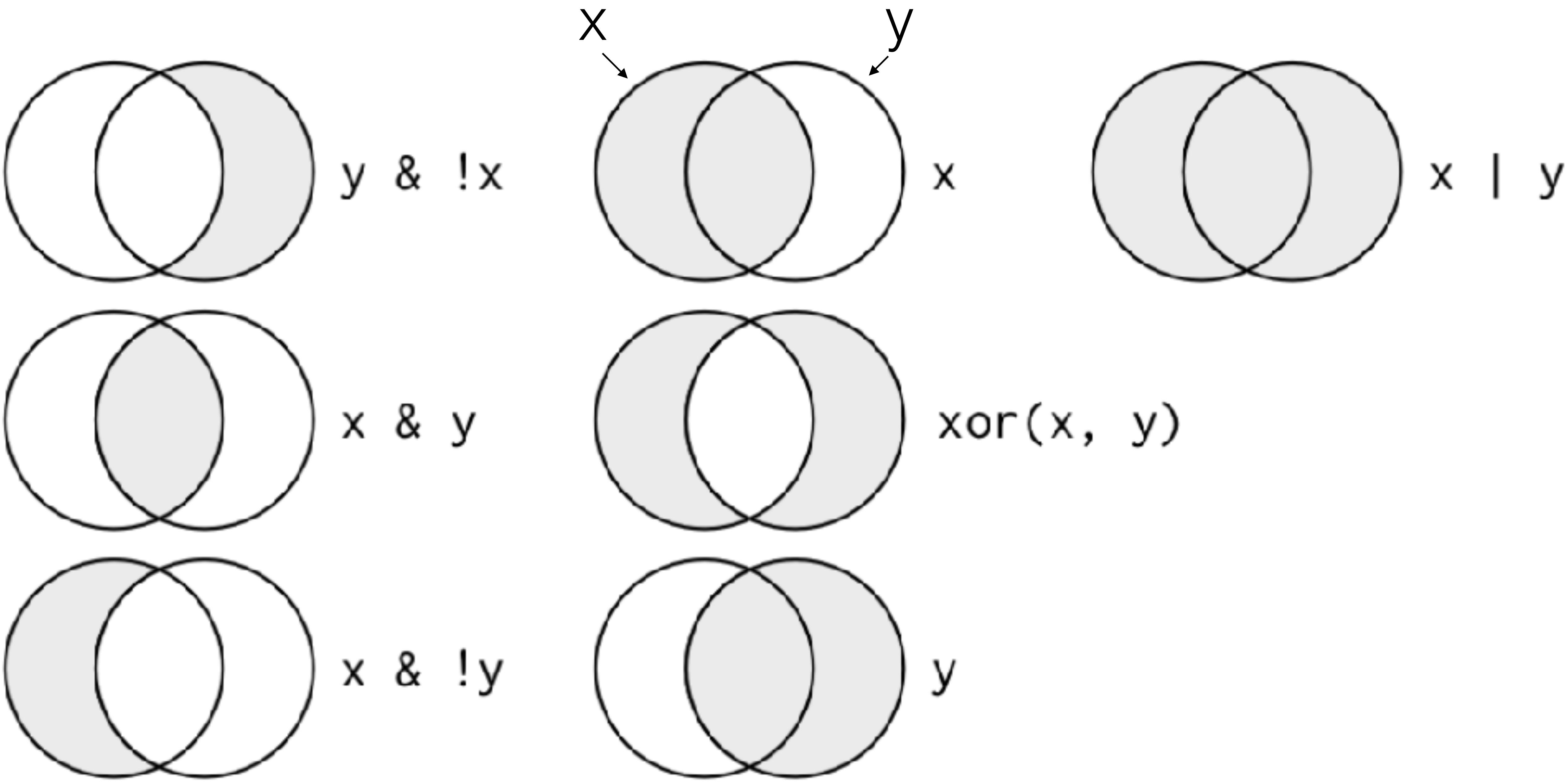
input_df

color	value
blue	1
blue	3
blue	4

output_df

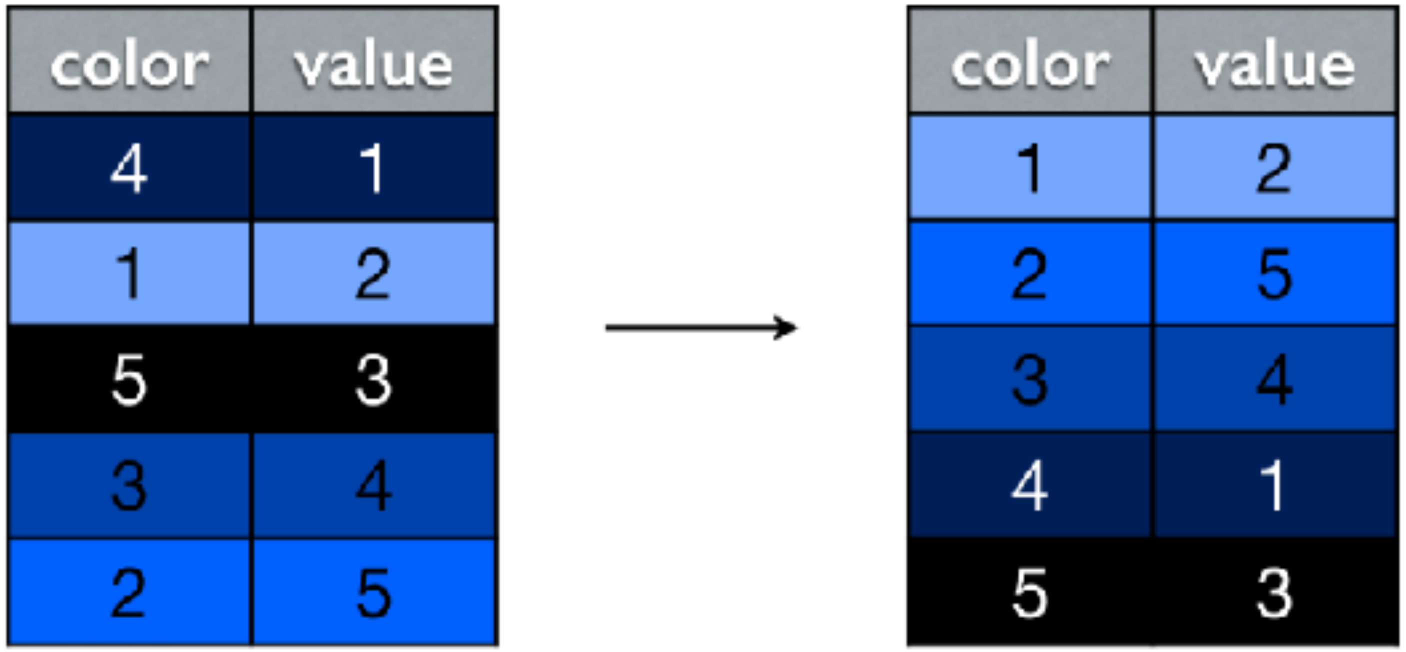
Subset observations based on their values

Logical operators



dplyr

output_df <– **arrange**(*input_df*, *color*)



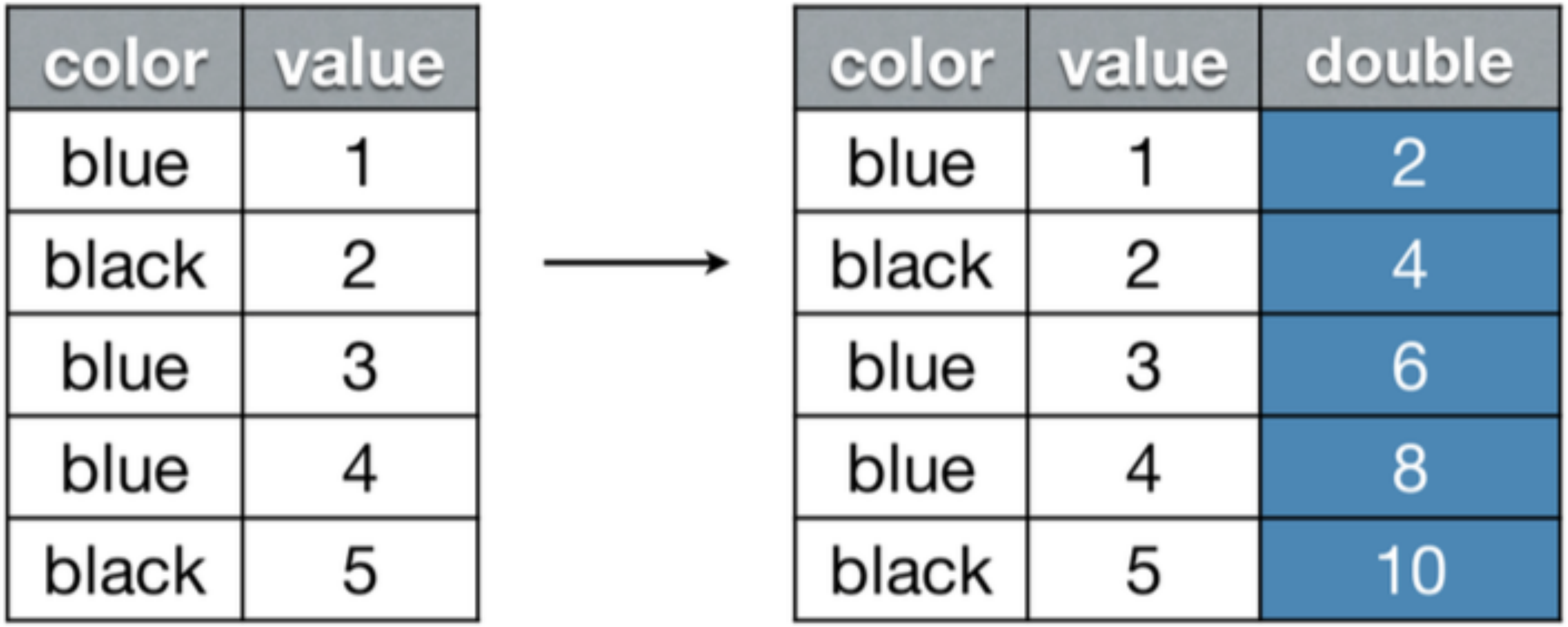
color	value
4	1
1	2
5	3
3	4
2	5

input_df

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

output_df

output_df <– **mutate**(*input_df, double = value*2*)



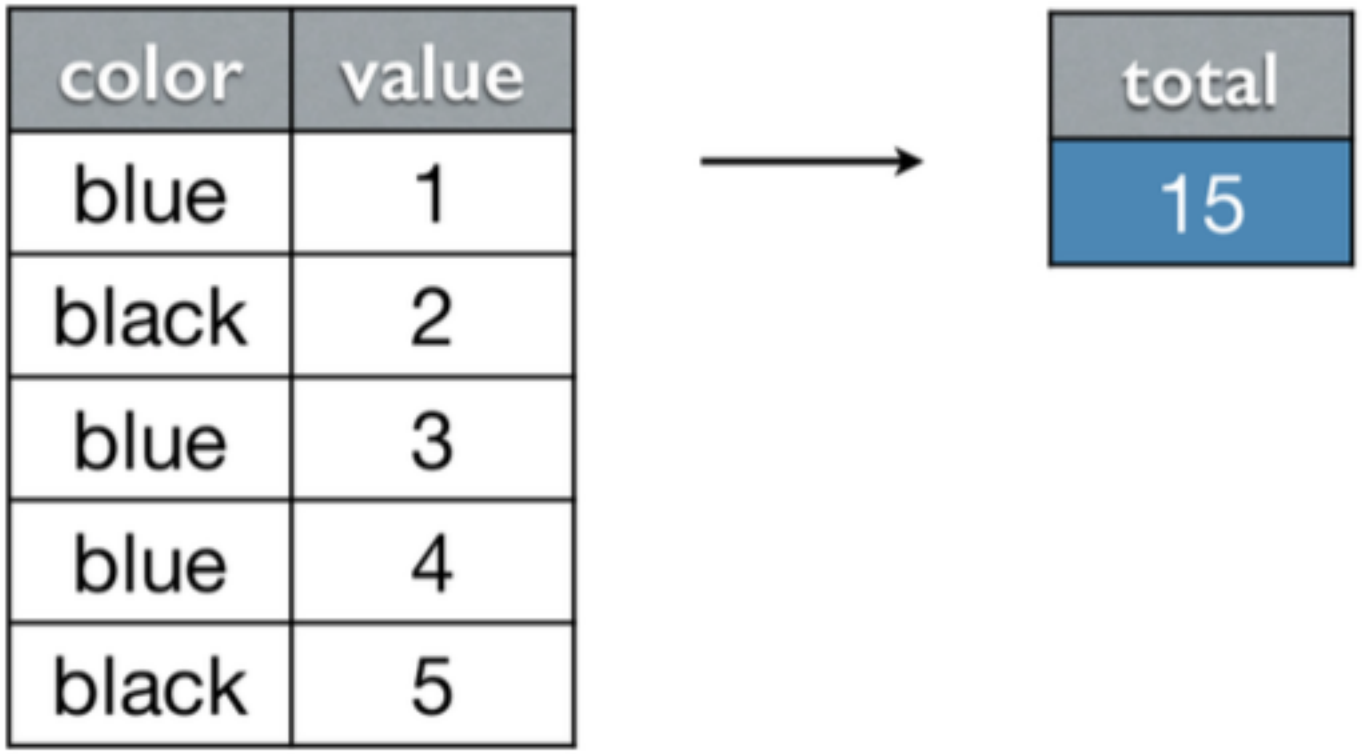
color	value
blue	1
black	2
blue	3
blue	4
black	5

input_df

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

output_df

```
output_df <– summarise(input_df, total = sum(value))
```

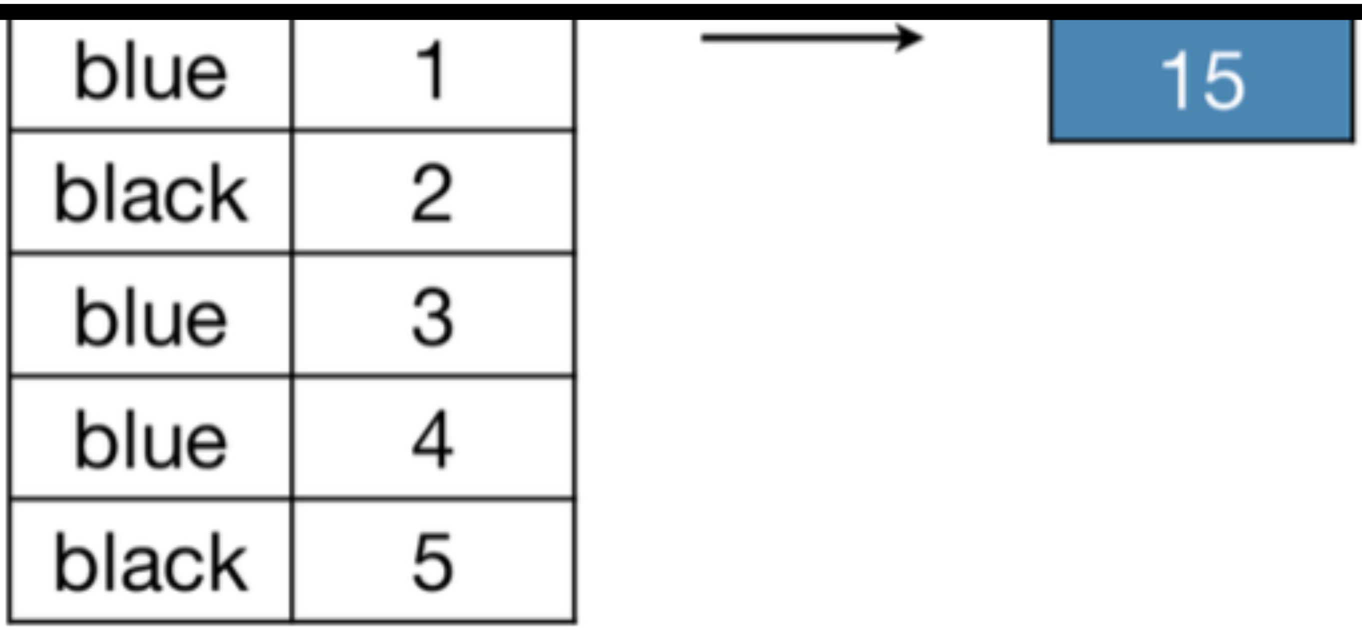


color	value
blue	1
black	2
blue	3
blue	4
black	5

total
15

output_df <– **summarise**(*input_df, total = sum(value*)

summarise() is not terribly useful unless we pair it with group_by()

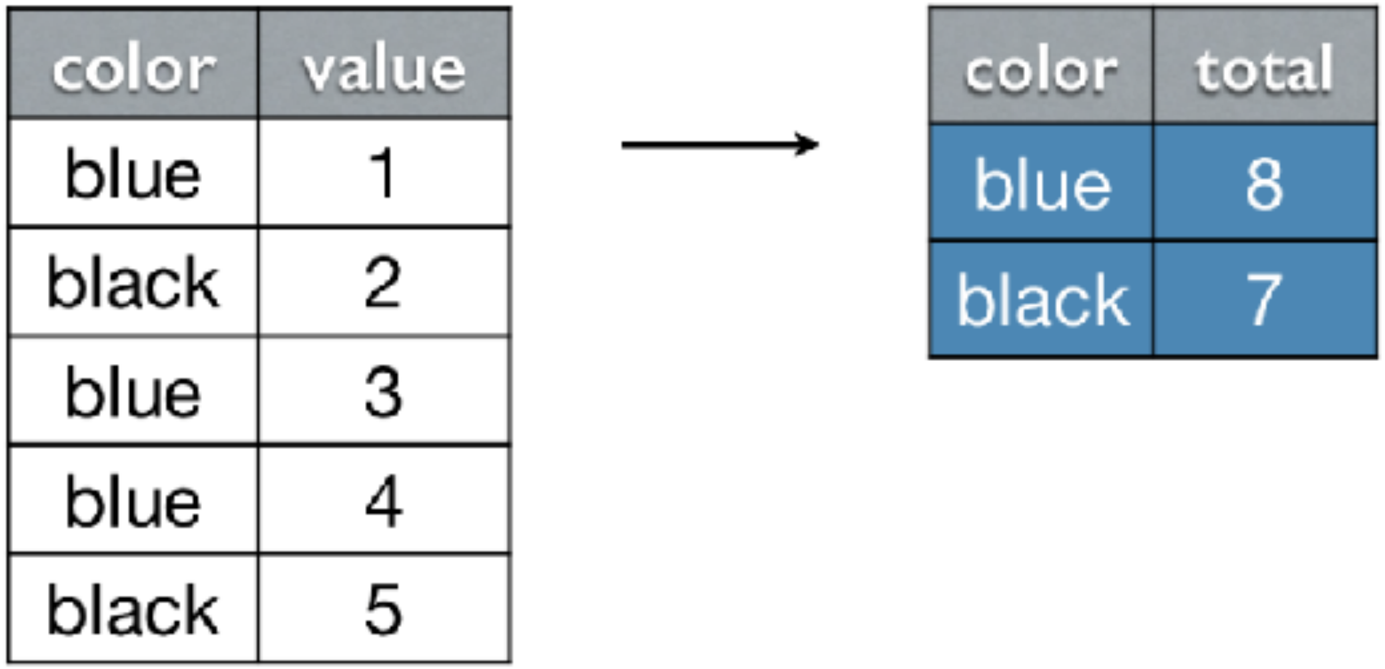


blue	1
black	2
blue	3
blue	4
black	5

group_by changes the unit of analysis from the complete dataset to individual groups.

```
> by_color <– group_by(input_df, color)

> summarise(by_color, total = sum(value))
```



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

color	total
blue	8
black	7

```
> input_df <- import_csv(“mydata.csv”)

> by_color <– group_by(input_df, color)

> output_df <– summarise(by_color, total = sum(value))
```

Combining multiple operations with the pipe

f(x, y)

↓

x %>% f(y)

Takes the member in the left and passes it as first argument of the function in the right

```
> input_df <- import_csv("mydata.csv")
> by_color <– group_by(input_df, color)
> output_df <– summarise(by_color, total = sum(value))
```

↓

```
output_df <- import_csv("mydata.csv") %>%
             group_by(color) %>%
             summarise(total = sum(value))
```

Exercise

TransformData.R

1: Find all flights that had an arrival delay of two or more hours

2: Find all flights that flew to Houston (IAH or HOU)

3.Find all flights that departed between midnight and 6am (inclusive)

4: Compare air_time with arr_time - dep_time. What do you expect to see?

5: Find the 10 most delayed flights

6: Look at the number of cancelled flights per day. Is there a pattern?

7: Is the proportion of cancelled flights related to the average delay?

8: What time of day should you fly if you want to avoid delays as much as possible?

9: Create your own questions [2]

10: Answer to a questions make by a colleague [2]