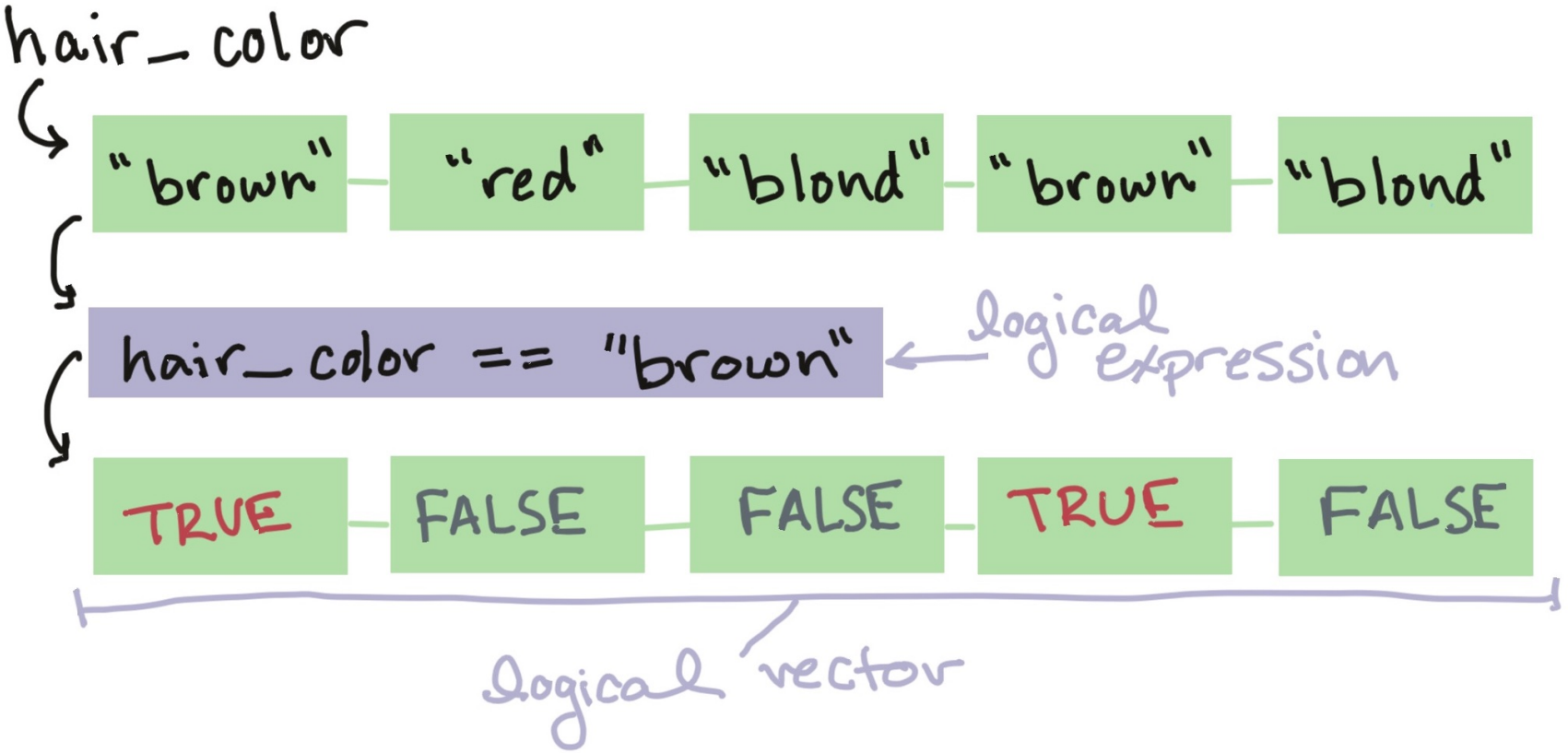
Exploring data #1

Logical operators, vectors, and expressions

Logical operators, vectors, and expressions

Logical operators are operators that conduct a logical test based on one or more vectors, while logical expressions are the full R expressions that use these operators to conduct the test. The output is a **logical vector**.



Logical expressions

Last week, you learned some about logical expressions and how to use them with the `filter` function.

You can use *logical vectors*, created with these expressions, for a lot data exploration tasks. We'll review them and add some more details this week.

Logical vectors

A logical expression outputs a *logical vector*. This logical vector will be the same length as the original vector tested by the logical statement:

```
length(beijing_pm$value)
```

```
## [1] 4344
```

```
length(beijing_pm$value > 500)
```

```
## [1] 4344
```

Logical vectors

Each element of the logical vector can only have one of three values (`TRUE`, `FALSE`, `NA`). The logical vector will have the value `TRUE` at any position where the original vector met the logical condition you tested, and `FALSE` anywhere else:

```
head(beijing_pm$value)
```

```
## [1] 505 485 466 435 405 402
```

```
head(beijing_pm$value > 500)
```

```
## [1]  TRUE FALSE FALSE FALSE FALSE FALSE
```

Logical vectors

Because the logical vector is the same length as the vector it's testing, you can add logical vectors to dataframes with `mutate`:

```
beijing_pm <- beijing_pm %>%
  mutate(beyond_index = value > 500)
```

Logical vectors

```
beijing_pm %>%
  select(sample_time, value, beyond_index)

## # A tibble: 4,344 x 3
##    sample_time          value beyond_index
##    <dttm>               <dbl> <lgl>
##  1 2017-01-01 00:00:00    505 TRUE
##  2 2017-01-01 01:00:00    485 FALSE
##  3 2017-01-01 02:00:00    466 FALSE
##  4 2017-01-01 03:00:00    435 FALSE
##  5 2017-01-01 04:00:00    405 FALSE
##  6 2017-01-01 05:00:00    402 FALSE
##  7 2017-01-01 06:00:00    407 FALSE
##  8 2017-01-01 07:00:00    435 FALSE
##  9 2017-01-01 08:00:00    472 FALSE
## 10 2017-01-01 09:00:00    465 FALSE
## # ... with 4,334 more rows
```

Logical vectors

As another example, you could add a column that is a logical vector of whether a day was in the "heating season", which usually ends on March 15 each year:

```
beijing_pm <- beijing_pm %>%
  mutate(heating = sample_time < ymd("2017-03-15"))
```

Common logical and relational operators in R

The **bang operator** (`!`) negates (flips) a logical expression:

```
c(1, 2, 3) == c(1, 2, 5)
```

```
## [1]  TRUE  TRUE FALSE
```

```
!(c(1, 2, 3) == c(1, 2, 5))
```

```
## [1] FALSE FALSE  TRUE
```

```
is.na(c(1, 2,  NA))
```

```
## [1] FALSE FALSE  TRUE
```

```
!is.na(c(1, 2, NA))
```

```
## [1]  TRUE  TRUE FALSE
```

Common logical and relational operators in R

The %in% operator will check each element of a vector to see if it's a value that is included in a second vector.

In this case, the two vectors don't have to have the same length:

```
c(1, 2, 3) %in% c(1, 5)
```

```
## [1]  TRUE FALSE FALSE
```

This logical expressions is asking *Is the first element of the first vector, 1, in the set given by the second vector, 1 and 5? Is the second element of the first vector, 2, in the set given by the second vector? Etc.*

Logical vectors

You can do a few cool things now with this vector. For example, you can use it with the `filter` function to pull out just the rows where `heating` is `TRUE`:

```
beijing_pm %>%
  filter(heating) %>%
  slice(1:3)
```

```
## # A tibble: 3 x 6
##   sample_time         value qc    aqi
##   <dttm>              <dbl> <chr> <fct>
## 1 2017-01-01 00:00:00   505 Valid Beyo~
## 2 2017-01-01 01:00:00   485 Valid Haza~
## 3 2017-01-01 02:00:00   466 Valid Haza~
## # ... with 2 more variables: beyond_index <lgl>,
## #   heating <lgl>
```

Logical vectors

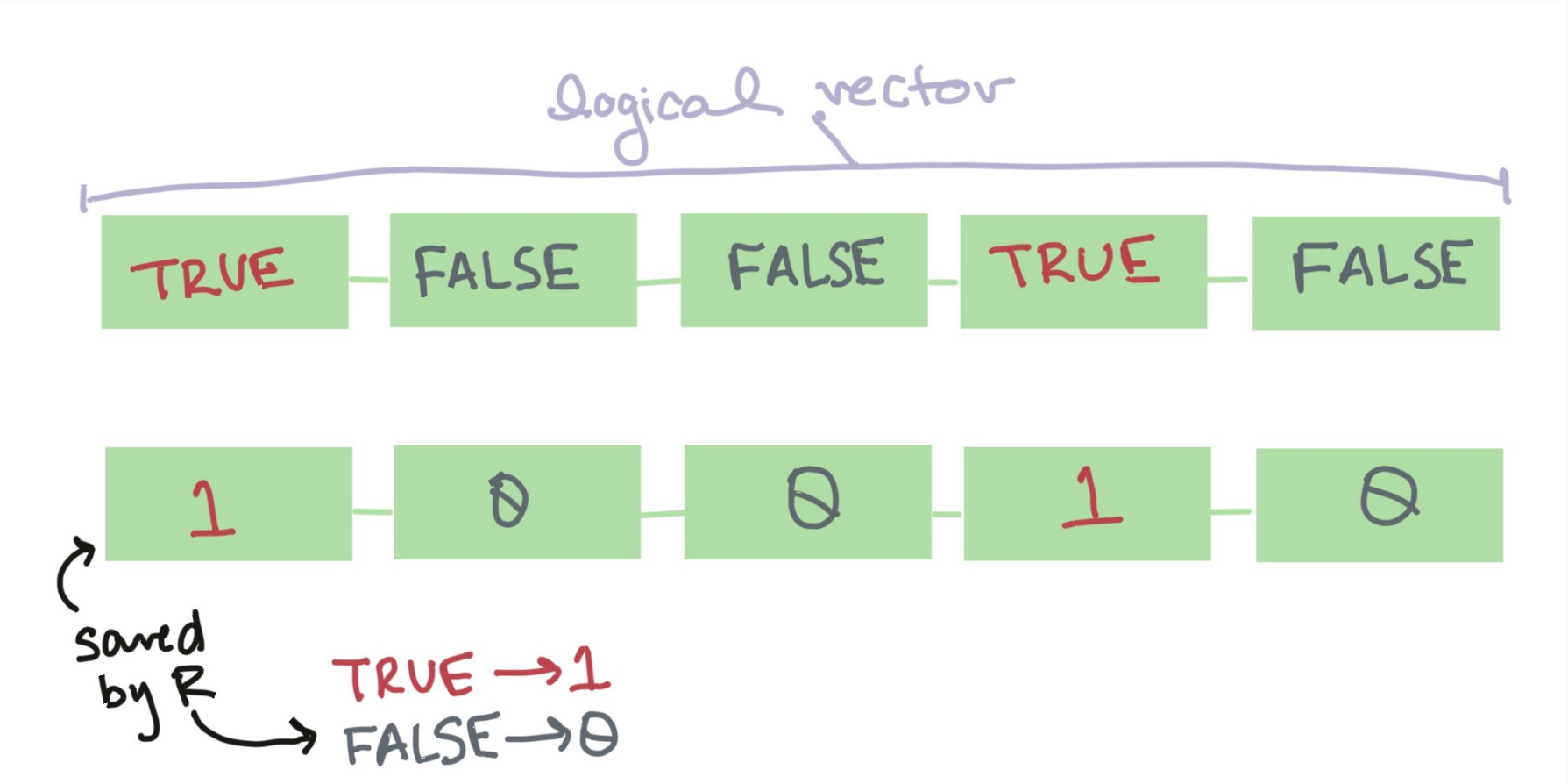
Or, with `!`, just the rows where `heating` is `FALSE`:

```
beijing_pm %>%
  filter(!heating) %>%
  slice(1:3)
```

```
## # A tibble: 3 x 6
##   sample_time         value qc    aqi
##   <dttm>              <dbl> <chr> <fct>
## 1 2017-03-15 00:00:00    54 Valid Mode~
## 2 2017-03-15 01:00:00    43 Valid Good
## 3 2017-03-15 02:00:00    39 Valid Good
## # ... with 2 more variables: beyond_index <lgl>,
## #   heating <lgl>
```

Logical vectors

All of the values in a logical vector are saved, at a deeper level, with a number. Values of `TRUE` are saved as 1 and values of `FALSE` are saved as 0.



Logical vectors

```
head(beijing_pm$beyond_index)
```

```
## [1]  TRUE FALSE FALSE FALSE FALSE FALSE
```

```
head(as.numeric(beijing_pm$beyond_index))
```

```
## [1] 1 0 0 0 0 0
```

Logical vectors

Therefore, you can use `sum()` to get the sum of all values in a vector. Because logical vector values are linked with numerical values of 0 or 1, you can use `sum()` to find out how many males and females are in the dataset:

```
sum(beijing_pm$beyond_index)
```

```
## [1] 27
```

```
sum(!beijing_pm$beyond_index)
```

```
## [1] 4317
```