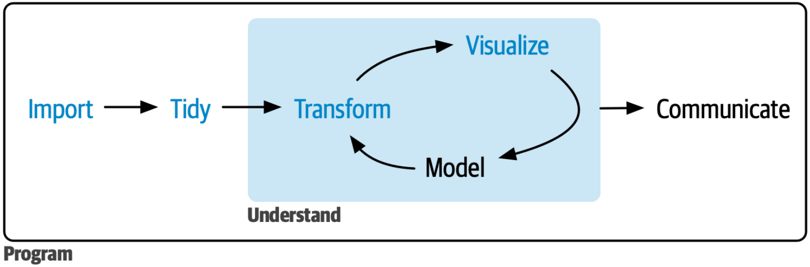
https://r4ds.hadley.nz/ Chapter 3-9

CSCI 297b, Spring 2023

May 4, 2023

The Big Picture



The dplyr package and the nycflights13 dataset

```
library(nycflights13)
library(tidyverse)
```

the nycflights13 dataset

```
> glimpse(flights)
Rows: 336,776
Columns: 19
$ year           <int> 2013, 2013, 2013, 2013,...
$ month          <int> 1, 1, 1, 1, 1, 1, 1, 1,...
$ day            <int> 1, 1, 1, 1, 1, 1, 1, 1,...
$ dep_time       <int> 517, 533, 542, 544, 554...
$ sched_dep_time <int> 515, 529, 540, 545, 600...
$ dep_delay      <dbl> 2, 4, 2, -1, -6, -4, -5...
$ arr_time       <int> 830, 850, 923, 1004, 81...
$ sched_arr_time <int> 819, 830, 850, 1022, 83...
$ arr_delay      <dbl> 11, 20, 33, -18, -25, 1...
$ carrier        <chr> "UA", "UA", "AA", "B6",...
$ flight         <int> 1545, 1714, 1141, 725, ...
$ tailnum        <chr> "N14228", "N24211", "N6...
$ origin         <chr> "EWR", "LGA", "JFK", "J...
$ dest           <chr> "IAH", "IAH", "MIA", "B...
$ air_time       <dbl> 227, 227, 160, 183, 116...
$ distance       <dbl> 1400, 1416, 1089, 1576,...
$ hour           <dbl> 5, 5, 5, 5, 6, 5, 6, 6,...
$ minute         <dbl> 15, 29, 40, 45, 0, 58, ...
$ time_hour      <dttm> 2013-01-01 05:00:00, 2...
```

The `dplyr` package

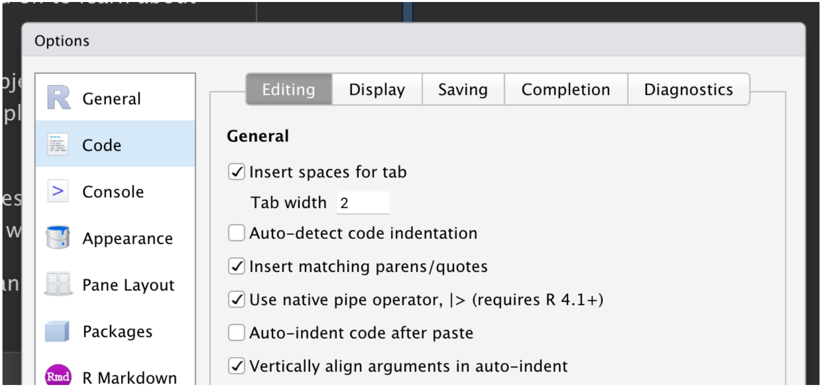
- The first argument is always a data frame.
- The subsequent arguments typically describe which columns to operate on, using the variable names (without quotes).
- The output is always a new data frame.
- Each verb operates on either
  - rows,
  - columns,
  - groups, or
  - tables

The pipe

```
      x |> f(y)  ⇔  f(x, y)
x |> f(y) |> g(z)  ⇔  g(f(x, y), z)
```

```
flights |>
  filter(dest == "IAH") |>
  group_by(year, month, day) |>
  summarize(
    arr_delay = mean(arr_delay, na.rm = TRUE)
  )
```

Global options



General

- [✓] Insert spaces for tab
- Tab width 2
- [ ] Auto-detect code indentation
- [✓] Insert matching parens/quotes
- [✓] Use native pipe operator, |> (requires R 4.1+)
- [ ] Auto-indent code after paste
- [✓] Vertically align arguments in auto-indent

- Enable "Use native pipe operator."
- This will enable you to produce the pipe with `ctrl-shift-M`

filter

```
flights |>
  filter(dep_delay > 120)
#> # A tibble: 9,723 × 19
#>    year month   day dep_time sched_dep_time dep_delay arr
#>   <int> <int> <int>    <int>          <int>     <dbl>
#> 1  2013     1     1      848           1835       853
#> 2  2013     1     1      957            733       144
#> 3  2013     1     1     1114            900       134
#> 4  2013     1     1     1540           1338       122
#> 5  2013     1     1     1815           1325       290
#> 6  2013     1     1     1842           1422       260
#> # i 9,717 more rows
#> # i 11 more variables: arr_delay <dbl>, carrier <chr>, fl
```

arrange

```
flights |>
  arrange(year, month, day, dep_time)
#> # A tibble: 336,776 × 19
#>    year month   day dep_time sched_dep_time dep_delay arr
#>   <int> <int> <int>    <int>          <int>     <dbl>
#> 1  2013     1     1      517            515         2
#> 2  2013     1     1      533            529         4
#> 3  2013     1     1      542            540         2
#> 4  2013     1     1      544            545        -1
#> 5  2013     1     1      554            600        -6
#> 6  2013     1     1      554            558        -4
#> # i 9,717 more rows
#> # i 11 more variables: arr_delay <dbl>, carrier <chr>, fl
```

distinct

```
# Find all unique origin and destination pairs
flights |>
  distinct(origin, dest)
#> # A tibble: 224 × 2
#>   origin dest
#>   <chr>  <chr>
#> 1 EWR    IAH
#> 2 LGA    IAH
#> 3 JFK    MIA
#> 4 JFK    BQN
#> 5 LGA    ATL
#> 6 EWR    ORD
#> # i 218 more rows
```

count

```
flights |>
  count(origin, dest, sort = TRUE)
#> # A tibble: 224 × 3
#>   origin dest      n
#>   <chr>  <chr> <int>
#> 1 JFK    LAX   11262
#> 2 LGA    ATL   10263
#> 3 LGA    ORD    8857
#> 4 JFK    SFO    8204
#> 5 LGA    CLT    6168
#> 6 EWR    ORD    6100
#> # i 218 more rows
```

Do exercise 6

mutate

```
flights |>
  mutate(
    gain = dep_delay - arr_delay,
    speed = distance / air_time * 60,
    .before = 1
  )
#> # A tibble: 336,776 × 21
#>    gain speed  year month   day dep_time sched_dep_time
#>   <dbl> <dbl> <int> <int> <int>    <int>          <int>
#> 1    -9  370.  2013     1     1      517            515
#> 2   -16  374.  2013     1     1      533            529
#> 3   -31  408.  2013     1     1      542            540
#> 4    17  517.  2013     1     1      544            545
#> 5    19  394.  2013     1     1      554            600
#> 6   -16  288.  2013     1     1      554            558
#> # i 336,770 more rows
#> # i 12 more variables: sched_arr_time <int>, arr_delay <
```

select

```
flights |>
  select(year, month, day)

flights |>
  select(year:day)

flights |>
  select(!year:day)

flights |>
  select(where(is.character))
```

- `starts_with("abc")`: matches names that begin with “abc”.
- `ends_with("xyz")`: matches names that end with “xyz”.
- `contains("ijk")`: matches names that contain “ijk”.
- `num_range("x", 1:3)`: matches x1, x2 and x3.

rename

```
flights |>
  rename(tail_num = tailnum)
#> # A tibble: 336,776 × 19
#>    year month   day dep_time sched_dep_time dep_delay arr
#>   <int> <int> <int>    <int>          <int>     <dbl>
#> 1  2013     1     1      517            515         2
#> 2  2013     1     1      533            529         4
#> 3  2013     1     1      542            540         2
#> 4  2013     1     1      544            545        -1
#> 5  2013     1     1      554            600        -6
#> 6  2013     1     1      554            558        -4
```

relocate

```
flights |>
  relocate(time_hour, air_time)
#> # A tibble: 336,776 × 19
#>   time_hour           air_time  year month   day dep_time
#>   <dttm>                 <dbl> <int> <int> <int>    <int
#> 1 2013-01-01 05:00:00      227  2013     1     1      517
#> 2 2013-01-01 05:00:00      227  2013     1     1      533
#> 3 2013-01-01 05:00:00      160  2013     1     1      542
#> 4 2013-01-01 05:00:00      183  2013     1     1      544
#> 5 2013-01-01 06:00:00      116  2013     1     1      554
#> 6 2013-01-01 05:00:00      150  2013     1     1      554
```

Do exercise 7

The Pipe

```
flights |>
  filter(dest == "IAH") |>
  mutate(speed = distance / air_time * 60) |>
  select(year:day, dep_time, carrier, flight, speed) |>
  arrange(desc(speed))
#> # A tibble: 7,198 × 7
#>    year month   day dep_time carrier flight speed
#>   <int> <int> <int>    <int> <chr>    <int> <dbl>
#> 1  2013     7     9      707 UA         226  522.
#> 2  2013     8    27     1850 UA        1128  521.
#> 3  2013     8    28      902 UA        1711  519.
#> 4  2013     8    28     2122 UA        1022  519.
#> 5  2013     6    11     1628 UA        1178  515.
#> 6  2013     8    27     1017 UA         333  515.
```

The Pipe vs. function nesting

```
arrange(
  select(
    mutate(
      filter(
        flights,
        dest == "IAH"
      ),
      speed = distance / air_time * 60
    ),
    year:day, dep_time, carrier, flight, speed
  ),
  desc(speed)
)
```

The Pipe vs. assignment to temporaries

```
flights1 <- filter(flights, dest == "IAH")
flights2 <- mutate(flights1, speed = distance / air_time * 60)
flights3 <- select(flights2, year:day, dep_time, carrier, flight, speed)
arrange(flights3, desc(speed))
```

group_by

```
flights |>
  group_by(month)
#> # A tibble: 336,776 × 19
#> # Groups:   month [12]
#>    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_tim
#>   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int
#> 1  2013     1     1      517            515         2      830            81
#> 2  2013     1     1      533            529         4      850            83
#> 3  2013     1     1      542            540         2      923            85
#> 4  2013     1     1      544            545        -1     1004           102
#> 5  2013     1     1      554            600        -6      812            83
#> 6  2013     1     1      554            558        -4      740            72
```

- All subsequent operations will now work “by month”

summarize

```
flights |>
  group_by(month) |>
  summarize(
    avg_delay = mean(dep_delay)
  )
#> # A tibble: 12 × 2
#>   month avg_delay
#>   <int>     <dbl>
#> 1     1        NA
#> 2     2        NA
#> 3     3        NA
#> 4     4        NA
#> 5     5        NA
#> 6     6        NA
```

- We forgot `na.rm`

summarize

```
flights |>
  group_by(month) |>
  summarize(
    delay = mean(dep_delay, na.rm = TRUE)
  )
#> # A tibble: 12 × 2
#>   month delay
#>   <int> <dbl>
#> 1     1  10.0
#> 2     2  10.8
#> 3     3  13.2
#> 4     4  13.9
#> 5     5  13.0
#> 6     6  20.8
```

summarize with n

```
flights |>
  group_by(month) |>
  summarize(
    delay = mean(dep_delay, na.rm = TRUE),
    n = n()
  )
#> # A tibble: 12 × 3
#>   month delay     n
#>   <int> <dbl> <int>
#> 1     1  10.0 27004
#> 2     2  10.8 24951
#> 3     3  13.2 28834
#> 4     4  13.9 28330
#> 5     5  13.0 28796
#> 6     6  20.8 28243
```

`slice`

- `df |> slice_head(n = 1)`
  takes the first row from each group.
- `df |> slice_tail(n = 1)`
  takes the last row in each group.
- `df |> slice_min(x, n = 1)`
  takes the row with the smallest value of column x.
- `df |> slice_max(x, n = 1)`
  takes the row with the largest value of column x.
- `df |> slice_sample(n = 1)`
  takes one random row.

slice

```
flights |>
  group_by(dest) |>
  slice_max(arr_delay, n = 1) |>
  relocate(dest)
#> # A tibble: 108 × 19
#> # Groups:   dest [105]
#>   dest   year month   day dep_time sched_dep_time dep_delay arr_time
#>   <chr> <int> <int> <int>    <int>          <int>     <dbl>    <int>
#> 1 ABQ    2013     7    22     2145           2007        98      132
#> 2 ACK    2013     7    23     1139            800       219     1250
#> 3 ALB    2013     1    25      123           2000       323      229
#> 4 ANC    2013     8    17     1740           1625        75     2042
#> 5 ATL    2013     7    22     2257            759       898      121
#> 6 AUS    2013     7    10     2056           1505       351     2347
```

Groiuping by more than one variable

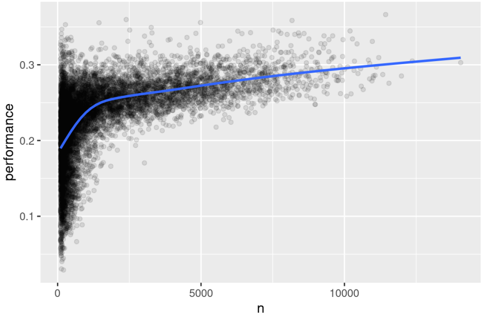
```
daily <- flights |>
  group_by(year, month, day)
daily
#> # A tibble: 336,776 × 19
#> # Groups:   year, month, day [365]
#>    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_tim
#>   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int
#> 1  2013     1     1      517            515         2      830            81
#> 2  2013     1     1      533            529         4      850            83
#> 3  2013     1     1      542            540         2      923            85
#> 4  2013     1     1      544            545        -1     1004           102
#> 5  2013     1     1      554            600        -6      812            83
#> 6  2013     1     1      554            558        -4      740            72
```

Case study

```
batters <- Lahman::Batting |>
  group_by(playerID) |>
  summarize(
    performance = sum(H, na.rm = TRUE) / sum(AB, na.rm = TRUE),
    n = sum(AB, na.rm = TRUE)
  )
batters
#> # A tibble: 20,166 × 3
#>   playerID  performance     n
#>   <chr>           <dbl> <int>
#> 1 aardsda01      0          4
#> 2 aaronha01      0.305  12364
#> 3 aaronto01      0.229    944
#> 4 aasedo01       0          5
#> 5 abadan01       0.0952    21
#> 6 abadfe01       0.111      9
#> # i 20,160 more rows
```

Case study

```
batters |>
  filter(n > 100) |>
  ggplot(aes(x = n, y = performance)) +
  geom_point(alpha = 1 / 10) +
  geom_smooth(se = FALSE)
```



- The variation in performance is larger among players with fewer at-bats.
- There's a positive correlation between skill (performance) and opportunities to hit the ball (n) because teams want to give their best batters the most opportunities to hit the ball.

Finding the best batters

```
batters |>
  arrange(desc(performance))
#> # A tibble: 20,166 × 3
#>   playerID  performance     n
#>   <chr>           <dbl> <int>
#> 1 abramge01           1     1
#> 2 alberan01           1     1
#> 3 banisje01           1     1
#> 4 bartocl01           1     1
#> 5 bassdo01            1     1
#> 6 birasst01           1     2
```

- http://varianceexplained.org/r/empirical_bayes_baseball/
- https://www.evanmiller.org/how-not-to-sort-by-average-rating.html

pivot_longer()

```
billboard
#> # A tibble: 317 × 79
#>   artist       track                date.entered   wk1   wk2   wk3   wk4   wk
#>   <chr>        <chr>                <date>       <dbl> <dbl> <dbl> <dbl> <dbl
#> 1 2 Pac        Baby Don't Cry (Ke... 2000-02-26      87    82    72    77
#> 2 2Ge+her      The Hardest Part O... 2000-09-02      91    87    92    NA
#> 3 3 Doors Down Kryptonite           2000-04-08      81    70    68    67     6
#> 4 3 Doors Down Loser                2000-10-21      76    76    72    69     6
#> 5 504 Boyz     Wobble Wobble        2000-04-15      57    34    25    17     1
#> 6 98^0         Give Me Just One N... 2000-08-19      51    39    34    26
#> # i 311 more rows
#> # i 71 more variables: wk6 <dbl>, wk7 <dbl>, wk8 <dbl>, wk9 <dbl>, ...
```

pivot_longer()

```
billboard |>
  pivot_longer(
    cols = starts_with("wk"),
    names_to = "week",
    values_to = "rank"
  )
#> # A tibble: 24,092 × 5
#>    artist track                   date.entered week   rank
#>    <chr>  <chr>                   <date>       <chr> <dbl>
#>  1 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk1      87
#>  2 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk2      82
#>  3 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk3      72
#>  4 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk4      77
#>  5 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk5      87
#>  6 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk6      94
#>  7 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk7      99
#>  8 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk8      NA
#>  9 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk9      NA
#> 10 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk10     NA
#> # i 24,082 more rows
```

pivot_longer()

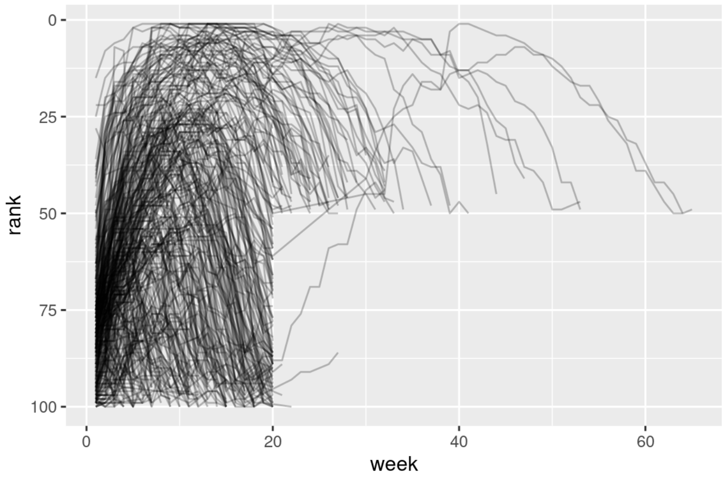
```
billboard |>
  pivot_longer(
    cols = starts_with("wk"),
    names_to = "week",
    values_to = "rank",
    values_drop_na = TRUE
  )
#> # A tibble: 5,307 × 5
#>   artist track                   date.entered week   rank
#>   <chr>  <chr>                   <date>       <chr> <dbl>
#> 1 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk1      87
#> 2 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk2      82
#> 3 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk3      72
#> 4 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk4      77
#> 5 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk5      87
#> 6 2 Pac  Baby Don't Cry (Keep... 2000-02-26   wk6      94
#> # i 5,301 more rows
```

pivot_longer()

```
billboard_longer <- billboard |>
  pivot_longer(
    cols = starts_with("wk"),
    names_to = "week",
    values_to = "rank",
    values_drop_na = TRUE
  ) |>
  mutate(
    week = parse_number(week)
  )
billboard_longer
#> # A tibble: 5,307 × 5
#>   artist track                   date.entered  week  rank
#>   <chr>  <chr>                   <date>       <dbl> <dbl>
#> 1 2 Pac  Baby Don't Cry (Keep... 2000-02-26       1    87
#> 2 2 Pac  Baby Don't Cry (Keep... 2000-02-26       2    82
#> 3 2 Pac  Baby Don't Cry (Keep... 2000-02-26       3    72
#> 4 2 Pac  Baby Don't Cry (Keep... 2000-02-26       4    77
#> 5 2 Pac  Baby Don't Cry (Keep... 2000-02-26       5    87
#> 6 2 Pac  Baby Don't Cry (Keep... 2000-02-26       6    94
#> # i 5,301 more rows
```

pivot_longer()

```
billboard_longer |>
  ggplot(aes(x = week, y = rank, group = track)) +
  geom_line(alpha = 0.25) +
  scale_y_reverse()
```

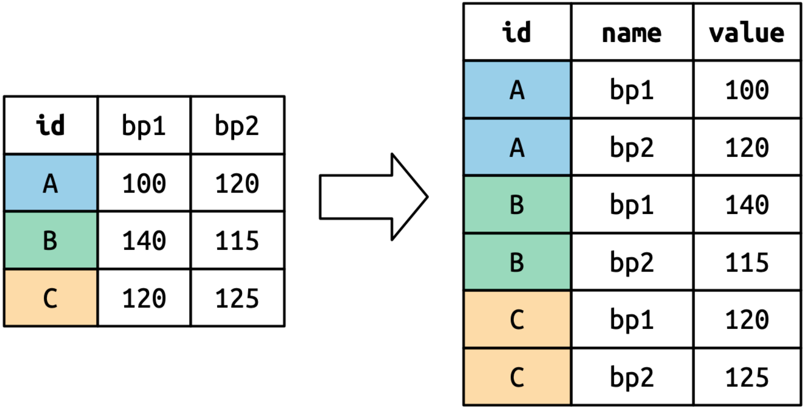


pivot_longer()

```
df <- tribble(
  ~id,  ~bp1, ~bp2,
   "A",  100,  120,
   "B",  140,  115,
   "C",  120,  125
)
df |>
  pivot_longer(
    cols = bp1:bp2,
    names_to = "measurement",
    values_to = "value"
  )
#> # A tibble: 6 × 3
#>   id    measurement value
#>   <chr> <chr>       <dbl>
#> 1 A     bp1           100
#> 2 A     bp2           120
#> 3 B     bp1           140
#> 4 B     bp2           115
#> 5 C     bp1           120
#> 6 C     bp2           125
```

pivot_longer()

```
df |>
  pivot_longer(
    cols = bp1:bp2,
    names_to = "measurement",
    values_to = "value"
  )
```

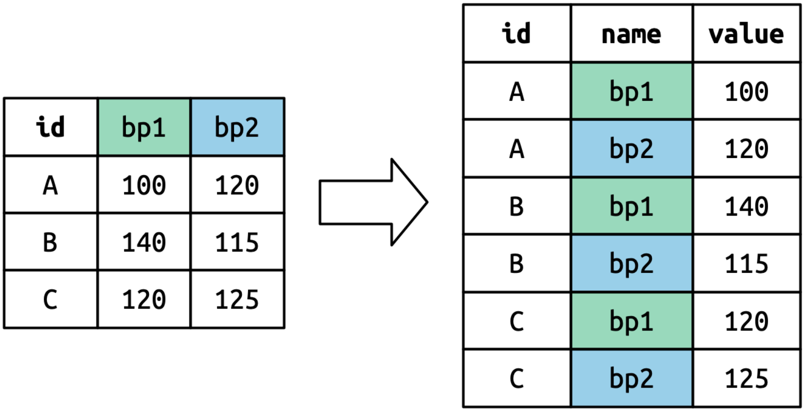


id	bp1	bp2
A	100	120
B	140	115
C	120	125

id	name	value
A	bp1	100
A	bp2	120
B	bp1	140
B	bp2	115
C	bp1	120
C	bp2	125

pivot_longer()

```
df |>
  pivot_longer(
    cols = bp1:bp2,
    names_to = "measurement",
    values_to = "value"
  )
```

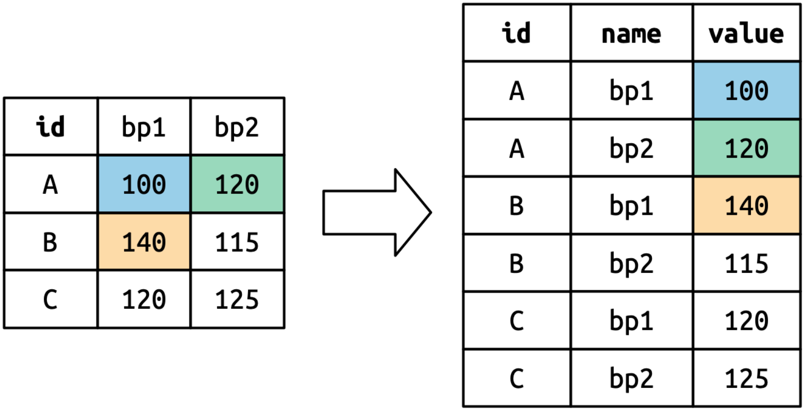


id	bp1	bp2
A	100	120
B	140	115
C	120	125

id	name	value
A	bp1	100
A	bp2	120
B	bp1	140
B	bp2	115
C	bp1	120
C	bp2	125

pivot_longer()

```
df |>
  pivot_longer(
    cols = bp1:bp2,
    names_to = "measurement",
    values_to = "value"
  )
```



id	bp1	bp2
A	100	120
B	140	115
C	120	125

id	name	value
A	bp1	100
A	bp2	120
B	bp1	140
B	bp2	115
C	bp1	120
C	bp2	125