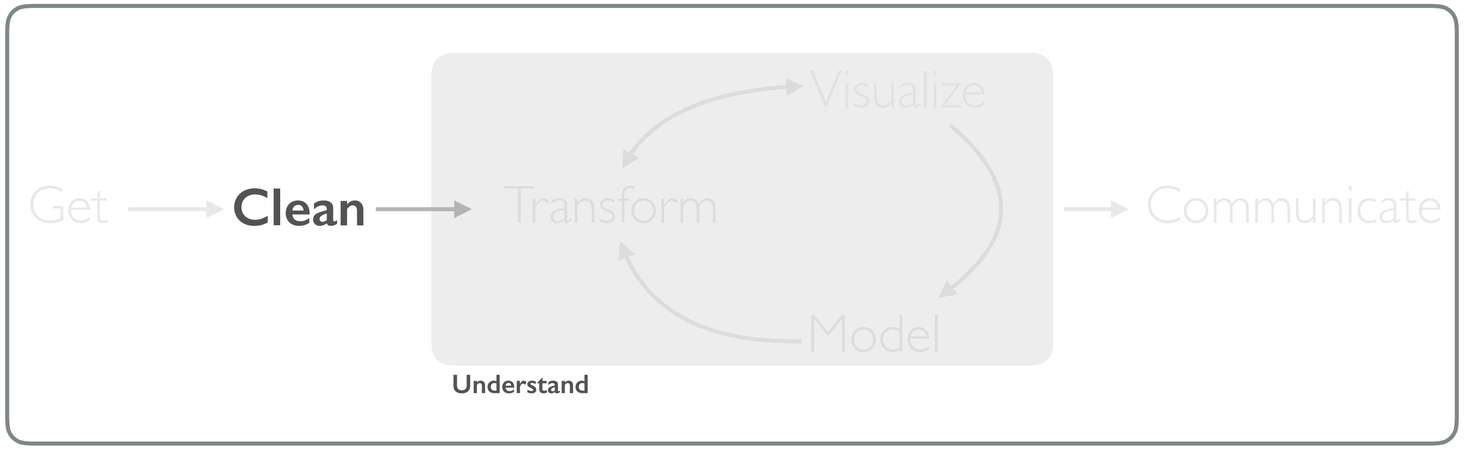
TIDY DATA



†A modified version of Hadley Wickham's analytic process

"Classroom data are like teddy bears and real data are like a grizzly bear with salmon blood dripping out its mouth." - Jenny Bryan

"Up to 80% of data analysis is spent on the process of cleaning and preparing data."

- cf. Wickham, 2014 and Dasu & Johnson, 2003

"Cannot emphasize enough how much time you save by putting analysis efforts into tidying data first." - Hilary Parker

WHAT IS TIDY DATA?

- One variable per column
- One observation per row

```
##        userid age dob_day dob_year dob_month gender tenure friend_count
## 1     2094382  14      19     1999        11   male    266            0
## 2     1192601  14       2     1999        11 female      6            0
## 3     2083884  14      16     1999        11   male     13            0
## 4     1203168  14      25     1999        12 female     93            0
## 5     1733186  14       4     1999        12   male     82            0
## 6     1524765  14       1     1999        12   male     15            0
## 7     1136133  13      14     2000         1   male     12            0
## 8     1680361  13       4     2000         1 female      0            0
## 9     1365174  13       1     2000         1   male     81            0
## 10    1712567  13       2     2000         2   male    171            0
## 11    1612453  13      22     2000         2   male     98            0
## 12    2104073  13       1     2000         2   male     55            0
```

IS THIS TIDY?

```
##                            State   X1980   X1990   X2000   X2005   X2006
## 1      United States ..... 2725285 2320337 2553844 2799250 2815544
## 2    Alabama .............   44894   40485   37819   37453   37918
## 3    Alaska ..............    5343    5386    6615    6909    7361
## 4    Arizona .............   28416   32103   38304   59498   54091
## 5    Arkansas ............   29577   26475   27335   26621   28790
## 6    California ..........  242172  236291  309866  355217  343515
## 7    Colorado ............   35897   32967   38924   44532   44424
## 8    Connecticut .........   38369   27878   31562   35515   36222
## 9    Delaware ............    7349    5550    6108    6934    7275
## 10   Florida .............   88755   88934  106708  133318  134686
## 11   Georgia .............   62963   56605   62563   70834   73498
## 12   Hawaii ..............   11472   10325   10437   10813   10922
## 13   Idaho ...............   12679   11971   16170   15768   16096
## 14   Illinois ............  136795  108119  111835  123615  126817
## 15   Indiana .............   73381   60012   57012   55444   57920
## 16   Iowa ................   42635   31796   33926   33547   33693
## 17   Kansas ..............   29397   25367   29102   30355   29818
## 18   Kentucky ............   41714   38005   36830   38399   38449
## 19   Louisiana ...........   46199   36053   38430   36009   33275
## 20   Maine ...............   15554   13839   12211   13077   12950
## 21   Maryland ............   54050   41566   47849   54170   55536
```

IS THIS TIDY?

```
##    Year White_unemployment Black_unemployment White_hs Black_hs
## 1  1972                5.1           10.400000     60.4     36.6
## 2  1973                4.3            9.425000     61.9     39.2
## 3  1974                5.1           10.541667     63.3     40.8
## 4  1975                7.8           14.808333     64.5     42.5
## 5  1976                7.0           13.950000     66.1     43.8
## 6  1977                6.2           14.033333     67.0     45.5
## 7  1978                5.2           12.741667     67.9     47.6
## 8  1979                5.1           12.341667     69.7     49.4
## 9  1980                6.3           14.291667     70.5     51.2
## 10 1981                6.7           15.625000     71.6     52.9
## 11 1982                8.6           18.908333     72.8     54.9
## 12 1983                8.4           19.500000     73.8     56.8
## 13 1984                6.5           15.925000     75.0     58.5
## 14 1985                6.2           15.091667     75.5     59.8
## 15 1986                6.0           14.558333     76.2     62.3
## 16 1987                5.3           12.966667     77.0     63.4
## 17 1988                4.7           11.708333     77.7     63.5
## 18 1989                4.5           11.466667     78.4     64.6
## 19 1990                4.8           11.408333     79.1     66.2
## 20 1991                6.1           12.491667     79.9     66.7
## 21 1992                6.6           14.200000     80.9     67.7
```

IS THIS TIDY?

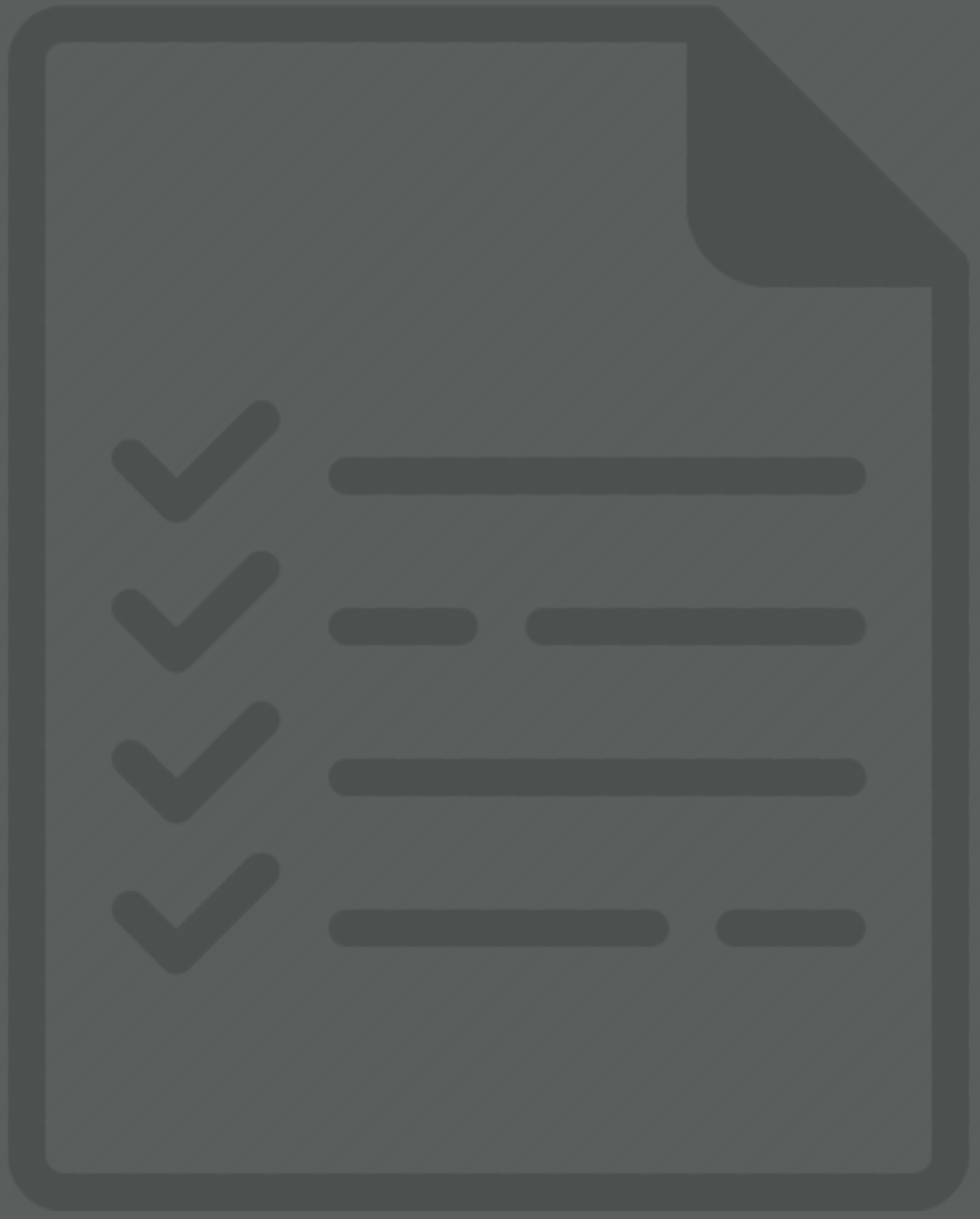
```
##                       state division
## 1          Connecticut (CT)        1
## 2                Maine (ME)        1
## 3        Massachusetts (MA)        1
## 4        New Hampshire (NH)        1
## 5         Rhode Island (RI)        1
## 6              Vermont (VT)        1
## 7          New Jersey (NJ)        2
## 8            New York (NY)        2
## 9         Pennsylvania (PA)        2
## 10           Illinois (IL)        3
## 11            Indiana (IN)        3
## 12           Michigan (MI)        3
## 13               Ohio (OH)        3
## 14          Wisconsin (WI)        3
## 15               Iowa (IA)        4
## 16             Kansas (KS)        4
## 17          Minnesota (MN)        4
## 18           Missouri (MO)        4
## 19           Nebraska (NE)        4
## 20         North Dakota (ND)        4
## 21         South Dakota (SD)        4
```

tidyr

You are going to learn four key `tidyr` functions that allow you to solve the vast majority of your data tidying challenges:

- `gather:` transforms data from wide to long
- `spread:` transforms data from long to wide
- `separate:` splits a single column into multiple columns
- `unite:` combines multiple columns into a single column

PREREQUISITES



PREREQUISITES

- Re-start your R session
  - **Windows:** Ctrl+Shift+F10
  - **Mac:** Command+Shift+F10
- Make sure your working directory is set to the course folder
- We will be using the various data sets that are in the data folder
- Data to follow along with the examples: `load("data/tidy_data.RData")`

PACKAGE PREREQUISITE

```
library(tidyverse)
#> Loading tidyverse: ggplot2
#> Loading tidyverse: tibble
#> Loading tidyverse: tidyr
#> Loading tidyverse: readr
#> Loading tidyverse: purrr
#> Loading tidyverse: dplyr
#> Conflicts with tidy packages ----------------------------------------------
#> filter(): dplyr, stats
#> lag():    dplyr, stats
```

`gather()`

Transform data from wide to long

gather()

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

```
cases %>% gather(Year, n, 2:4)
```

dataframe to reshape

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000
FR	2013	7000
DE	2013	6200
US	2013	13000

gather()

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

```
cases %>% gather(Year, n, 2:4)
```

name of the new "key" column

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000
FR	2013	7000
DE	2013	6200
US	2013	13000

gather()

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

```
cases %>% gather(Year, n, 2:4)
```

name of the new "value" column

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000
FR	2013	7000
DE	2013	6200
US	2013	13000

gather()

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

key

value

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000
FR	2013	7000
DE	2013	6200
US	2013	13000

```
cases %>% gather(Year, n, 2:4)
```

columns to collapse

gather()

Code alternatives:

```
# These all produce the same results:
        cases %>% gather(Year, n, `2011`:`2013`)
        cases %>% gather(Year, n, `2011`, `2012`, `2013`)
        cases %>% gather(Year, n, 2:4)
        cases %>% gather(Year, n, -Country)

# Also note that if you do not supply arguments for na.rm or convert values then the defaults are used
```

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

gather()

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000
FR	2013	7000
DE	2013	6200
US	2013	13000

YOUR TURN!

1. *Import the* **`bomber_wide.rds`** *file in the data folder*

2. *Reshape this data from wide to long*

SOLUTION

```
read_rds("data/bomber_wide.rds") %>%
  gather(Year, Value, -c(Type, MD))
     Type   MD Year Value
1  Bomber  B-1 1996 26914
2  Bomber  B-2 1996  2364
3  Bomber B-52 1996 28511
4  Bomber  B-1 1997 25219
5  Bomber  B-2 1997  2776
6  Bomber B-52 1997 26034
7  Bomber  B-1 1998 24205
8  Bomber  B-2 1998  2166
9  Bomber B-52 1998 25639
10 Bomber  B-1 1999 23306
11 Bomber  B-2 1999  3672
12 Bomber B-52 1999 24500
13 Bomber  B-1 2000 25013
```

`spread()`

Transform data from long to wide

spread()

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000
FR	2013	7000
DE	2013	6200
US	2013	13000

```
cases %>% spread(Year, n)
```

dataframe to reshape

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

spread()

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000
FR	2013	7000
DE	2013	6200
US	2013	13000

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

```
cases %>% spread(Year, n)
```

column to use as keys (new column names)

spread()

Country	Year	n
FR	2011	7000
DE	2011	5800
US	2011	15000
FR	2012	6900
DE	2012	6000
US	2012	14000
FR	2013	7000
DE	2013	6200
US	2013	13000

Country	2011	2012	2013
FR	7000	6900	7000
DE	5800	6000	6200
US	15000	14000	13000

```
cases %>% spread(Year, n)
```

column to use as values
(new column cells)

YOUR TURN!

1. *Import the* **`bomber_long.rds`** *file in the data folder*

2. *Reshape this data from long to wide*

SOLUTION

```
read_rds("data/bomber_long.rds") %>%
  spread(Output, Value)
      Type   MD   FY      Cost    FH   Gallons
1   Bomber  B-1 1996  72753781 26914  88594449
2   Bomber  B-1 1997  71297263 25219  85484074
3   Bomber  B-1 1998  84026805 24205  85259038
4   Bomber  B-1 1999  71848336 23306  79323816
5   Bomber  B-1 2000  58439777 25013  86230284
6   Bomber  B-1 2001  94946077 25059  86892432
7   Bomber  B-1 2002  96458536 26581  89198262
8   Bomber  B-1 2003  68650070 21491  74485788
9   Bomber  B-1 2004 101895634 28118 101397707
10  Bomber  B-1 2005 124816690 21859  78410415
11  Bomber  B-1 2006 174627869 20163  69984142
12  Bomber  B-1 2007 204486404 24629  85112485
13  Bomber  B-1 2008 266109848 23024  78084791
14  Bomber  B-1 2009 185902082 23065  81030579
```

`separate()`

Split a single column into multiple columns

separate()

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

storm	wind	pressure	year	month	day
Alberto	110	1007	2000	08	12
Alex	45	1009	1998	07	30
Allison	65	1005	1995	06	04
Ana	40	1013	1997	07	1
Arlene	50	1010	1999	06	13
Arthur	45	1010	1996	06	21

```
storms %>% separate(date, c("year", "month", "day"), sep = "-")
```

dataframe to reshape

separate()

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

storm	wind	pressure	year	month	day
Alberto	110	1007	2000	08	12
Alex	45	1009	1998	07	30
Allison	65	1005	1995	06	04
Ana	40	1013	1997	07	1
Arlene	50	1010	1999	06	13
Arthur	45	1010	1996	06	21

```
storms %>% separate(date, c("year", "month", "day"), sep = "-")
```

column to split into multiple columns

separate()

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

storm	wind	pressure	year	month	day
Alberto	110	1007	2000	08	12
Alex	45	1009	1998	07	30
Allison	65	1005	1995	06	04
Ana	40	1013	1997	07	1
Arlene	50	1010	1999	06	13
Arthur	45	1010	1996	06	21

```
storms %>% separate(date, c("year", "month", "day"), sep = "-")
```

names of the new variable columns

separate()

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

storm	wind	pressure	year	month	day
Alberto	110	1007	2000	08	12
Alex	45	1009	1998	07	30
Allison	65	1005	1995	06	04
Ana	40	1013	1997	07	1
Arlene	50	1010	1999	06	13
Arthur	45	1010	1996	06	21

```
storms %>% separate(date, c("year", "month", "day"), sep = "-")
```

how to separate current variable

separate()

Code alternatives:

```
# These all produce the same results:
        storms %>% separate(date, c("year", "month", “day"))
        storms %>% separate(date, c("year", "month", "day"), sep = "-")

# By default, if no separator is specified, will separate by any regular expression that matches
# any sequence of non-alphanumeric values
```

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

separate()

storm	wind	pressure	year	month	day
Alberto	110	1007	2000	08	12
Alex	45	1009	1998	07	30
Allison	65	1005	1995	06	04
Ana	40	1013	1997	07	1
Arlene	50	1010	1999	06	13
Arthur	45	1010	1996	06	21

YOUR TURN!

1. *Import the* **`bomber_combined.rds`** *file in the data folder*

2. *Separate the AC variable into "Type" and "MD"*

SOLUTION

```
read_rds("data/bomber_combined.rds") %>%
  separate(AC, into = c("Type", "MD"), sep = " ")
     Type   MD   FY      Cost    FH   Gallons
1  Bomber  B-1 1996  72753781 26914  88594449
2  Bomber  B-1 1997  71297263 25219  85484074
3  Bomber  B-1 1998  84026805 24205  85259038
4  Bomber  B-1 1999  71848336 23306  79323816
5  Bomber  B-1 2000  58439777 25013  86230284
6  Bomber  B-1 2001  94946077 25059  86892432
7  Bomber  B-1 2002  96458536 26581  89198262
8  Bomber  B-1 2003  68650070 21491  74485788
9  Bomber  B-1 2004 101895634 28118 101397707
10 Bomber  B-1 2005 124816690 21859  78410415
11 Bomber  B-1 2006 174627869 20163  69984142
12 Bomber  B-1 2007 204486404 24629  85112485
13 Bomber  B-1 2008 266109848 23024  78084791
```

`unite()`

Combine multiple columns into a single column

unite()

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

storm	wind	pressure	year	month	day
Alberto	110	1007	2000	08	12
Alex	45	1009	1998	07	30
Allison	65	1005	1995	06	04
Ana	40	1013	1997	07	1
Arlene	50	1010	1999	06	13
Arthur	45	1010	1996	06	21

```
storms %>% unite(date, year, month, day, sep = "-")
```

dataframe to reshape

unite()

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

storm	wind	pressure	year	month	day
Alberto	110	1007	2000	08	12
Alex	45	1009	1998	07	30
Allison	65	1005	1995	06	04
Ana	40	1013	1997	07	1
Arlene	50	1010	1999	06	13
Arthur	45	1010	1996	06	21

```
storms %>% unite(date, year, month, day, sep = "-")
```

name of new "merged" column

unite()

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

storm	wind	pressure	year	month	day
Alberto	110	1007	2000	08	12
Alex	45	1009	1998	07	30
Allison	65	1005	1995	06	04
Ana	40	1013	1997	07	1
Arlene	50	1010	1999	06	13
Arthur	45	1010	1996	06	21

```
storms %>% unite(date, year, month, day, sep = "-")
```

columns to merge

unite()

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

storm	wind	pressure	year	month	day
Alberto	110	1007	2000	08	12
Alex	45	1009	1998	07	30
Allison	65	1005	1995	06	04
Ana	40	1013	1997	07	1
Arlene	50	1010	1999	06	13
Arthur	45	1010	1996	06	21

```
storms %>% unite(date, year, month, day, sep = "-")
```

separator to use btwn merged values

unite()

Code alternatives:

```
# These all produce the same results:
        storms %>% unite(date, year, month, day, sep = “_")
        storms %>% unite(date, year, month, day)

# If no separator is identified, "_" will automatically be used
```

storm	wind	pressure	date
Alberto	110	1007	2000-08-12
Alex	45	1009	1998-07-30
Allison	65	1005	1995-06-04
Ana	40	1013	1997-07-01
Arlene	50	1010	1999-06-13
Arthur	45	1010	1996-06-21

storm	wind	pressure	year	month	day
Alberto	110	1007	2000	08	12
Alex	45	1009	1998	07	30
Allison	65	1005	1995	06	04
Ana	40	1013	1997	07	1
Arlene	50	1010	1999	06	13
Arthur	45	1010	1996	06	21

YOUR TURN!

1. *Import the* **`bomber_prefix.rds`** *file in the data folder*

2. *Unite the prefix and number columns into a "MD" variable with "-" separator*

SOLUTION

```
read_rds("data/bomber_prefix.rds") %>%
  unite(MD, prefix, number, sep = "-")
      Type   MD   FY  Output     Value
1   Bomber  B-1 1996      FH     26914
2   Bomber  B-1 1997      FH     25219
3   Bomber  B-1 1998      FH     24205
4   Bomber  B-1 1999      FH     23306
5   Bomber  B-1 2000      FH     25013
6   Bomber  B-1 2001      FH     25059
7   Bomber  B-1 2002      FH     26581
8   Bomber  B-1 2003      FH     21491
9   Bomber  B-1 2004      FH     28118
10  Bomber  B-1 2005      FH     21859
11  Bomber  B-1 2006      FH     20163
12  Bomber  B-1 2007      FH     24629
13  Bomber  B-1 2008      FH     23024
```

CHALLENGE

1. *Import the* **`bomber_mess.rds`** *file in the data folder*

2. *Clean this data up so it looks like:*

```
# A tibble: 57 × 6
     Type    MD    FY       Cost    FH   Gallons
*   <chr> <chr> <chr>      <int> <int>     <int>
1  Bomber   B-1  1996   72753781 26914  88594449
2  Bomber   B-1  1997   71297263 25219  85484074
3  Bomber   B-1  1998   84026805 24205  85259038
4  Bomber   B-1  1999   71848336 23306  79323816
5  Bomber   B-1  2000   58439777 25013  86230284
6  Bomber   B-1  2001   94946077 25059  86892432
7  Bomber   B-1  2002   96458536 26581  89198262
8  Bomber   B-1  2003   68650070 21491  74485788
9  Bomber   B-1  2004  101895634 28118 101397707
10 Bomber   B-1  2005  124816690 21859  78410415
# ... with 47 more rows
```

SOLUTION

```
read_rds("data/bomber_mess.rds") %>%
  unite(col = MD, prefix:number, sep = "-") %>%
  separate(Metric, into = c("FY", "Output")) %>%
  spread(Output, Value) %>%
  as_tibble()
# A tibble: 57 × 6
     Type    MD    FY      Cost    FH   Gallons
*    <chr> <chr> <chr>     <int> <int>     <int>
1  Bomber   B-1  1996  72753781 26914  88594449
2  Bomber   B-1  1997  71297263 25219  85484074
3  Bomber   B-1  1998  84026805 24205  85259038
4  Bomber   B-1  1999  71848336 23306  79323816
5  Bomber   B-1  2000  58439777 25013  86230284
6  Bomber   B-1  2001  94946077 25059  86892432
7  Bomber   B-1  2002  96458536 26581  89198262
8  Bomber   B-1  2003  68650070 21491  74485788
```

WHAT TO REMEMBER

FUNCTIONS TO REMEMBER

Operator/Function	Description
`gather`	transform data from wide to long
`spread`	transform data from long to wide
`unite`	unite multiple columns into a single column
`separate`	separate one column into multiple columns