STOR 320 Tidy Data

Lecture 13

Yao Li
Department of Statistics and Operations Research
UNC Chapel Hill

Introduction

- Read Chapter 12

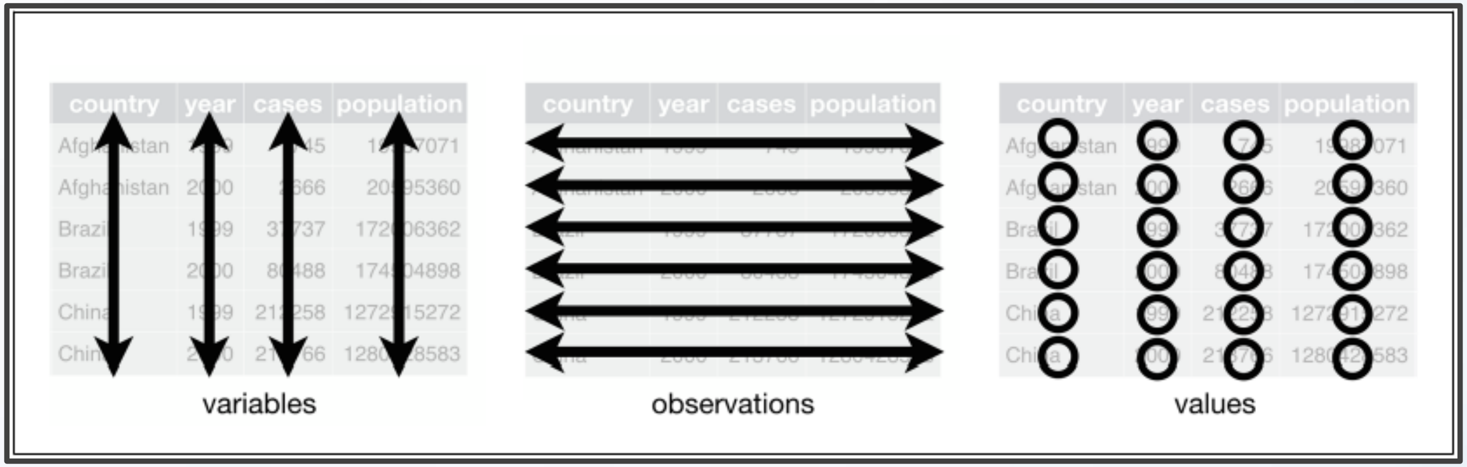
- Functions From tidyr Package

```
>library(tidyr)
```

  - pivot_longer()
  - pivot_wider()
  - separate()
  - unite()
  - complete()

Tidy Data Definition

- For Tidy Data:
  - Each Variable Must Have Its Own Column
  - Each Observation Must Have Its Own Row
  - Each Value Must Have Its Own Cell



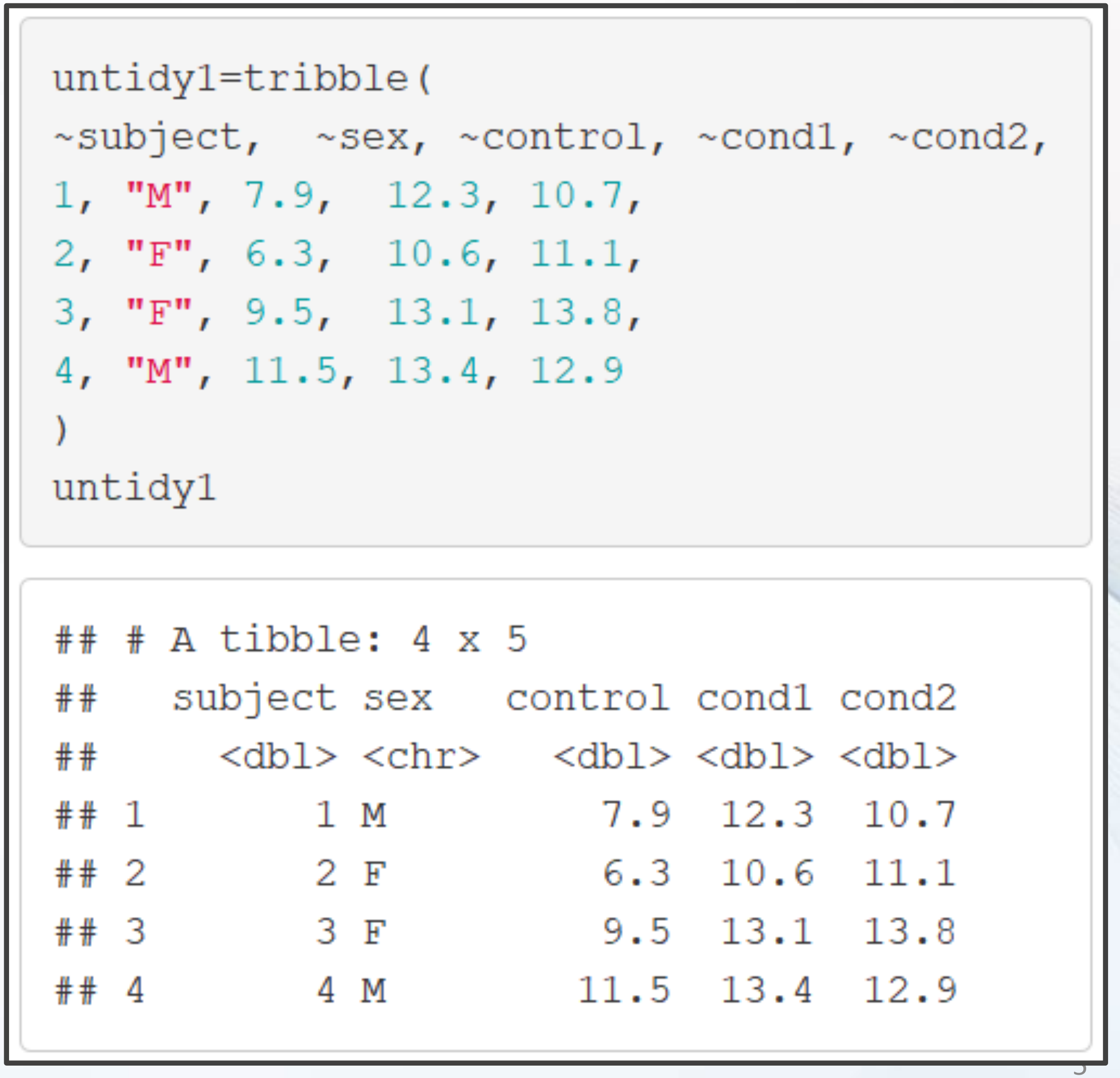
Problem

- Most Data is Not Tidy
- Reason: Data Collectors Often Don’t Know How Data Should Be Recorded Since They Don’t Analyze the Data
- Common Problems
  - A Variable Spread Across Multiple Columns
  - A Observation is Spread Across Multiple Rows

“Tidy datasets are all alike, but every messy dataset is messy in its own way.” — Hadley Wickham

Untidy Data Example 1

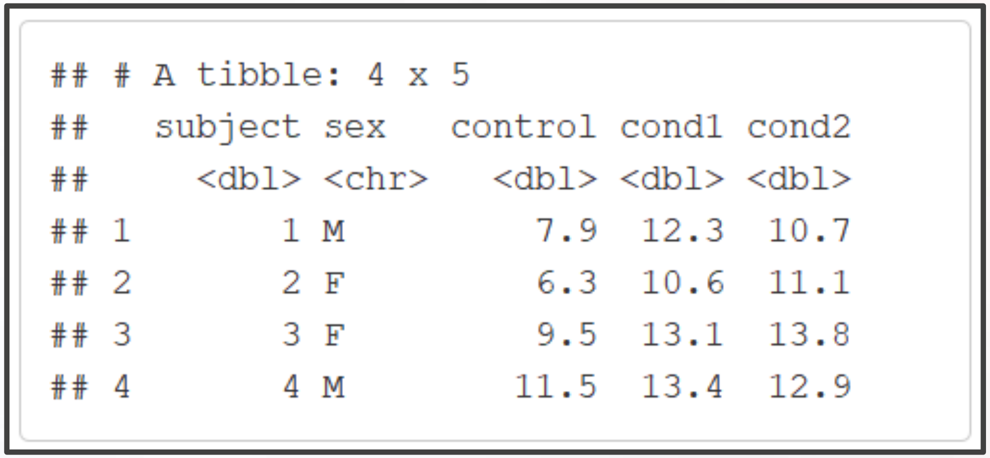
- Multiple Columns for One Variable



```
untidy1=tribble(
~subject,  ~sex, ~control, ~cond1, ~cond2,
1, "M", 7.9,  12.3, 10.7,
2, "F", 6.3,  10.6, 11.1,
3, "F", 9.5,  13.1, 13.8,
4, "M", 11.5, 13.4, 12.9
)
untidy1
```

```
## # A tibble: 4 x 5
##   subject sex   control cond1 cond2
##     <dbl> <chr>   <dbl> <dbl> <dbl>
## 1       1 M         7.9  12.3  10.7
## 2       2 F         6.3  10.6  11.1
## 3       3 F         9.5  13.1  13.8
## 4       4 M        11.5  13.4  12.9
```

Problem



```
## # A tibble: 4 x 5
##   subject sex   control cond1 cond2
##     <dbl> <chr>   <dbl> <dbl> <dbl>
## 1       1 M         7.9  12.3  10.7
## 2       2 F         6.3  10.6  11.1
## 3       3 F         9.5  13.1  13.8
## 4       4 M        11.5  13.4  12.9
```

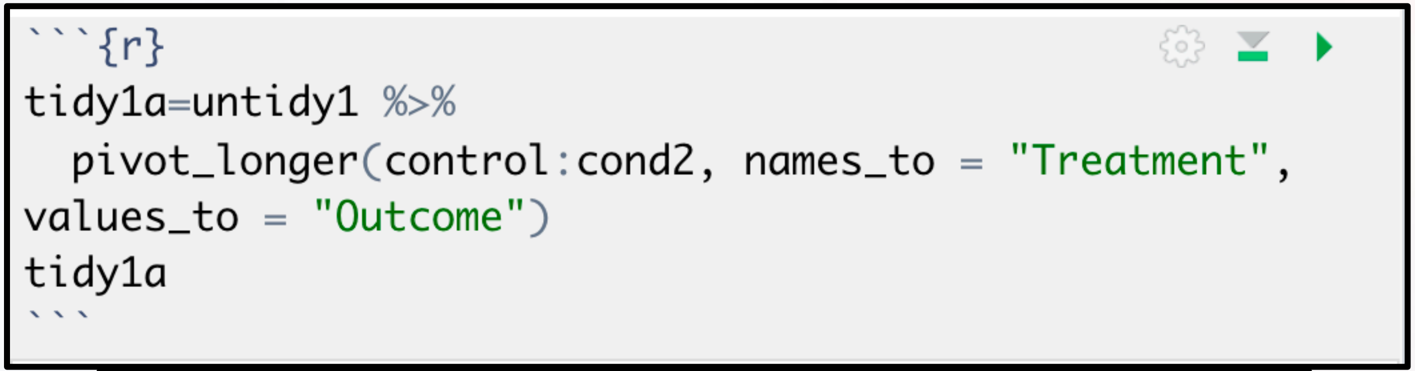
- Multiple Treatment Data

- Variables “Control”, “Cond1”, and “Cond2” are Measuring the Same Thing Under Different Treatments

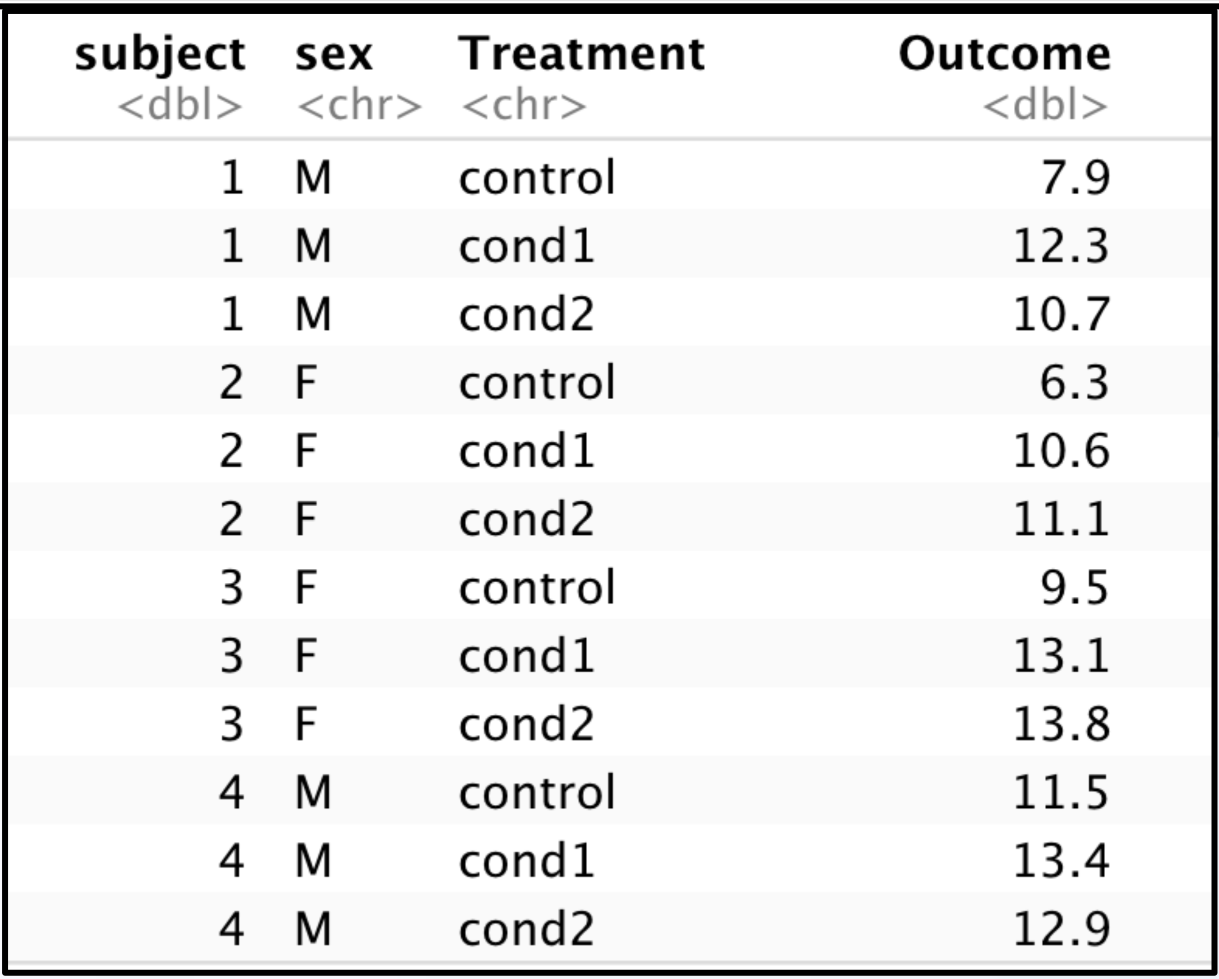
- The Name of the Variable Whose Values Form the Column Names Can Be Called “Treatment”

- The Name of the Variable Whose Values are Spread Over the Cells Can Be Called “Outcome”

Longer

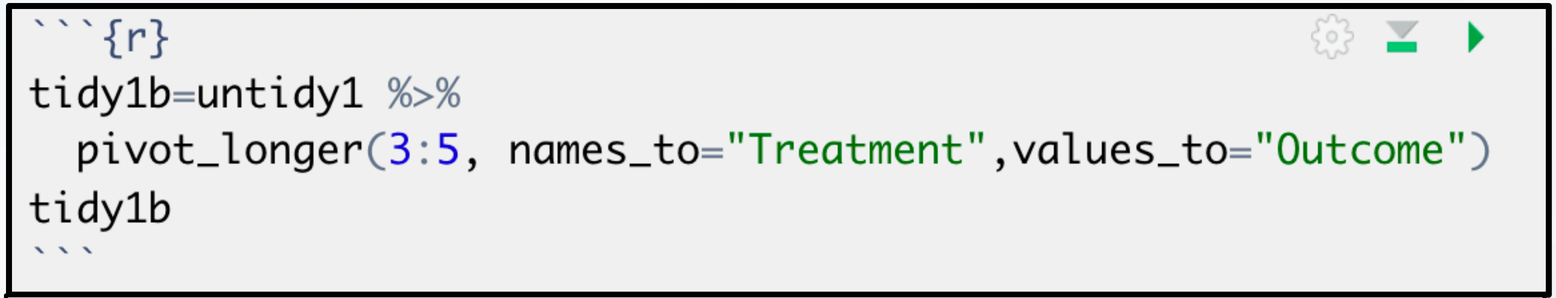


~~~
```{r}
tidy1a=untidy1 %>%
  pivot_longer(control:cond2, names_to = "Treatment",
values_to = "Outcome")
tidy1a
```
~~~

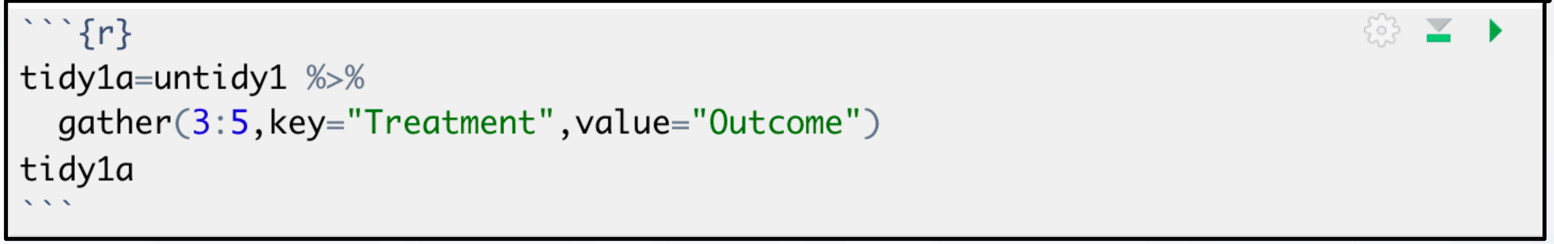


subject <dbl>	sex <chr>	Treatment <chr>	Outcome <dbl>
1	M	control	7.9
1	M	cond1	12.3
1	M	cond2	10.7
2	F	control	6.3
2	F	cond1	10.6
2	F	cond2	11.1
3	F	control	9.5
3	F	cond1	13.1
3	F	cond2	13.8
4	M	control	11.5
4	M	cond1	13.4
4	M	cond2	12.9

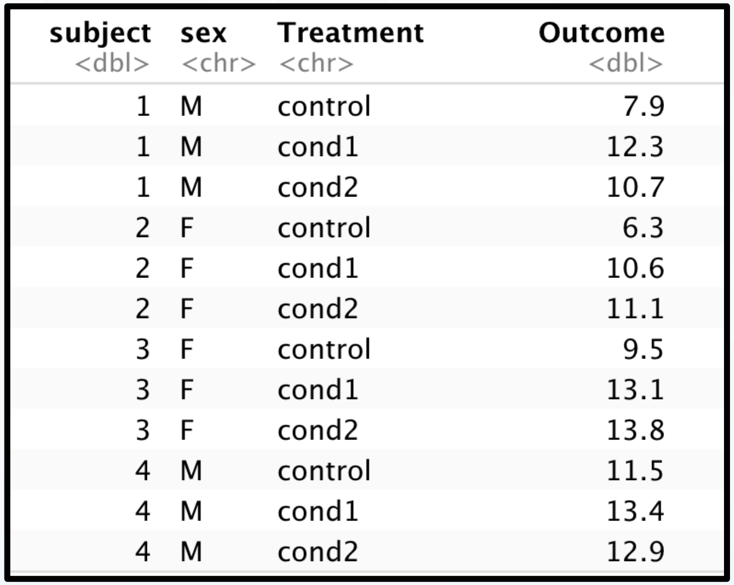
Longer by index



~~~
```{r}
tidy1b=untidy1 %>%
  pivot_longer(3:5, names_to="Treatment",values_to="Outcome")
tidy1b
```
~~~

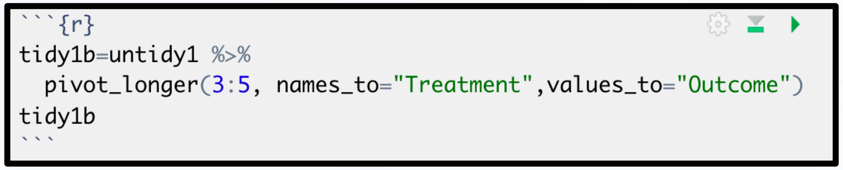


~~~
```{r}
tidy1a=untidy1 %>%
  gather(3:5,key="Treatment",value="Outcome")
tidy1a
```
~~~

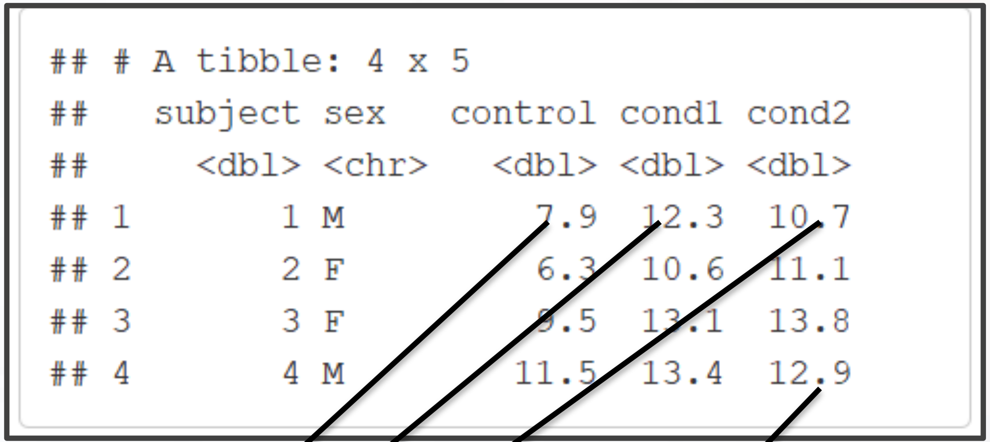


subject <dbl>	sex <chr>	Treatment <chr>	Outcome <dbl>
1	M	control	7.9
1	M	cond1	12.3
1	M	cond2	10.7
2	F	control	6.3
2	F	cond1	10.6
2	F	cond2	11.1
3	F	control	9.5
3	F	cond1	13.1
3	F	cond2	13.8
4	M	control	11.5
4	M	cond1	13.4
4	M	cond2	12.9

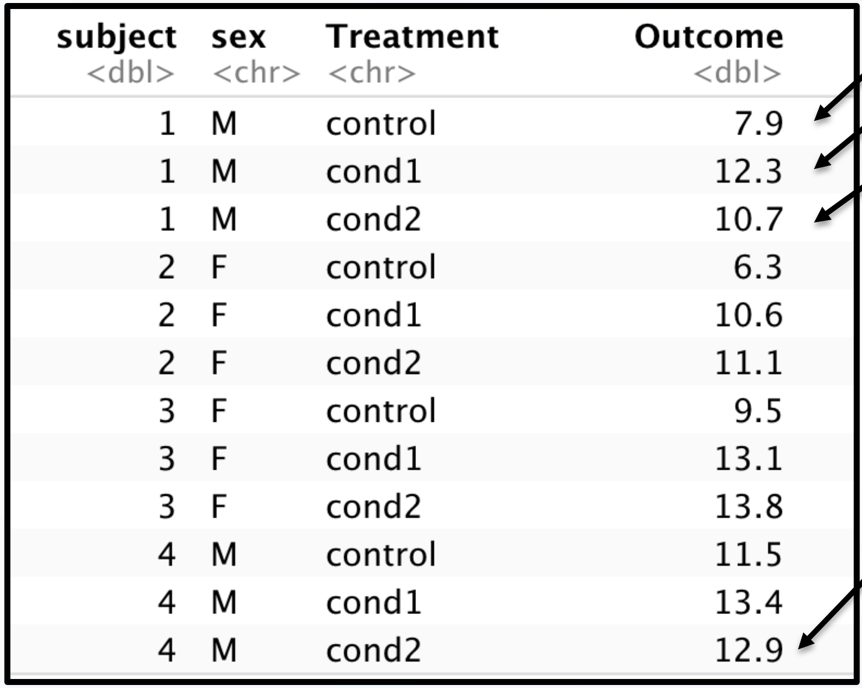
Process



~~~
```{r}
tidy1b=untidy1 %>%
  pivot_longer(3:5, names_to="Treatment",values_to="Outcome")
tidy1b
```
~~~

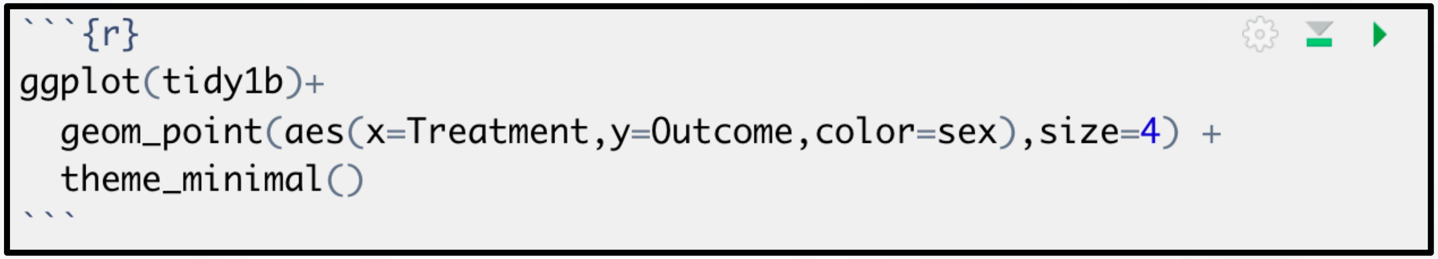


```
## # A tibble: 4 x 5
##   subject sex   control cond1 cond2
##     <dbl> <chr>   <dbl> <dbl> <dbl>
## 1       1 M         7.9  12.3  10.7
## 2       2 F         6.3  10.6  11.1
## 3       3 F         9.5  13.1  13.8
## 4       4 M        11.5  13.4  12.9
```

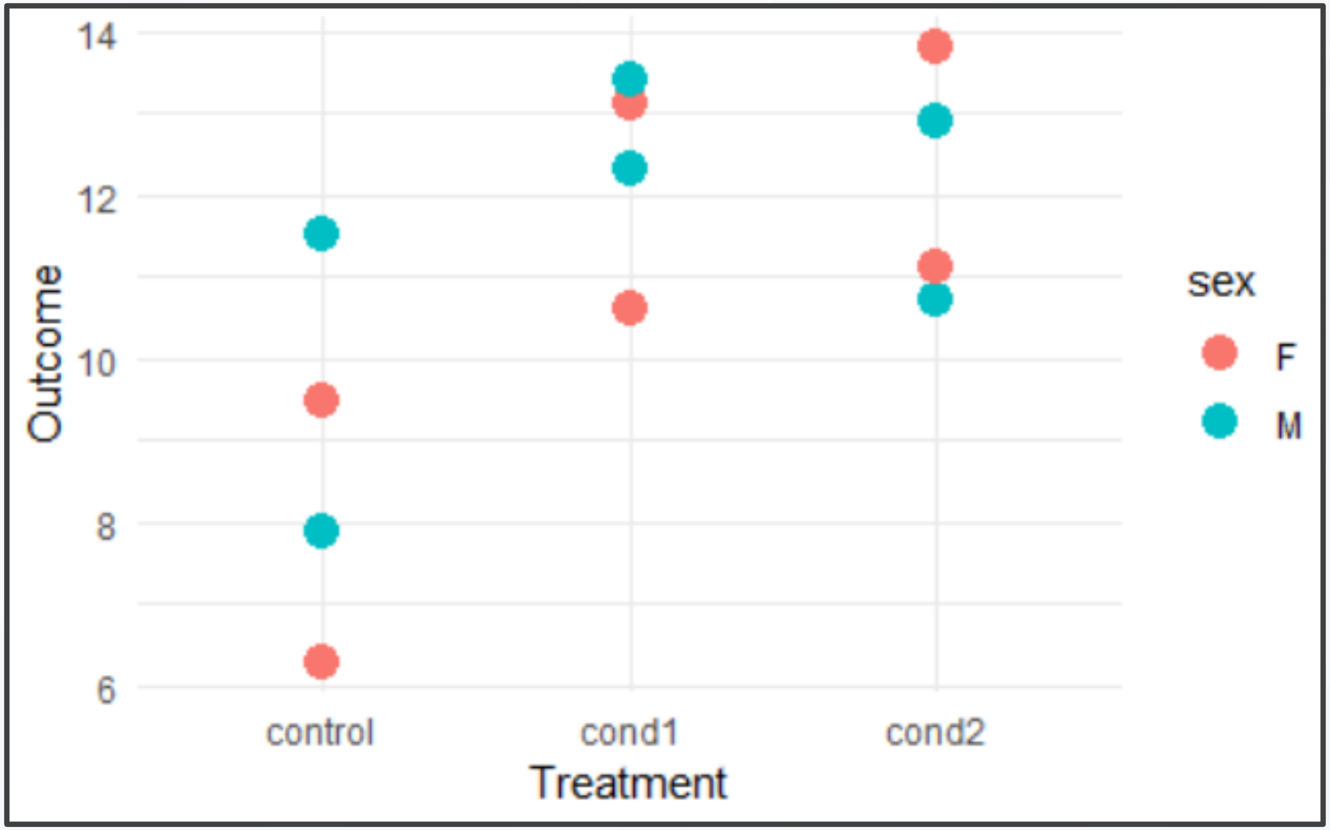


subject <dbl>	sex <chr>	Treatment <chr>	Outcome <dbl>
1	M	control	7.9
1	M	cond1	12.3
1	M	cond2	10.7
2	F	control	6.3
2	F	cond1	10.6
2	F	cond2	11.1
3	F	control	9.5
3	F	cond1	13.1
3	F	cond2	13.8
4	M	control	11.5
4	M	cond1	13.4
4	M	cond2	12.9

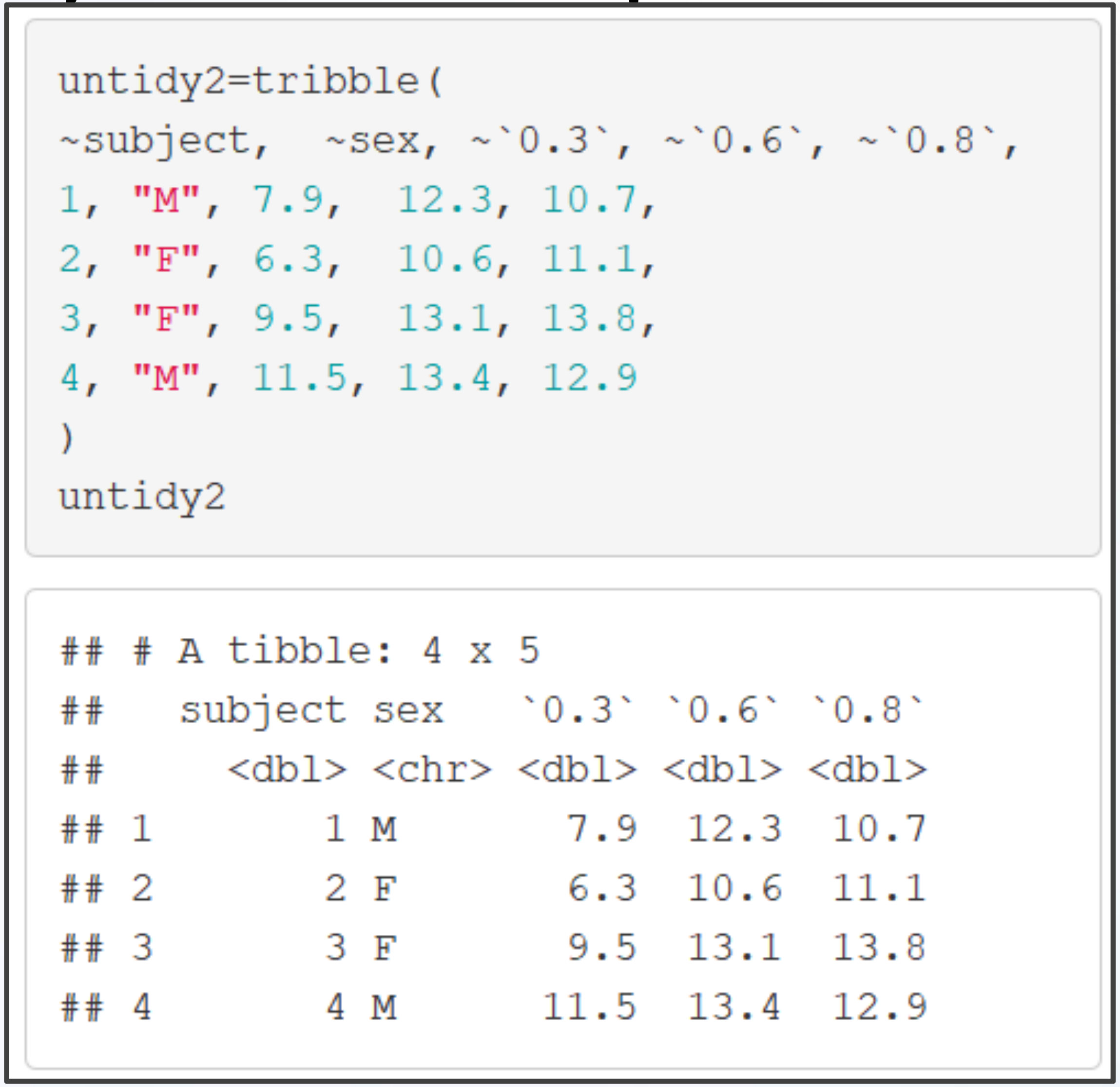
Longer



~~~
```{r}
ggplot(tidy1b)+
  geom_point(aes(x=Treatment,y=Outcome,color=sex),size=4) +
  theme_minimal()
```
~~~



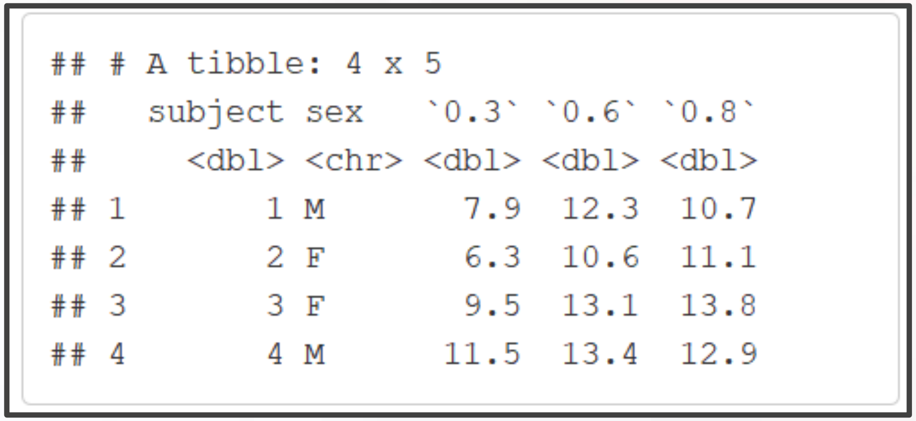
Untidy Data Example 2



```
untidy2=tribble(
~subject,  ~sex, ~`0.3`, ~`0.6`, ~`0.8`,
1, "M", 7.9,  12.3, 10.7,
2, "F", 6.3,  10.6, 11.1,
3, "F", 9.5,  13.1, 13.8,
4, "M", 11.5, 13.4, 12.9
)
untidy2
```

```
## # A tibble: 4 x 5
##   subject sex   `0.3` `0.6` `0.8`
##     <dbl> <chr> <dbl> <dbl> <dbl>
## 1       1 M       7.9  12.3  10.7
## 2       2 F       6.3  10.6  11.1
## 3       3 F       9.5  13.1  13.8
## 4       4 M      11.5  13.4  12.9
```

Problem



```
## # A tibble: 4 x 5
##   subject sex   `0.3` `0.6` `0.8`
##     <dbl> <chr> <dbl> <dbl> <dbl>
## 1       1 M       7.9  12.3  10.7
## 2       2 F       6.3  10.6  11.1
## 3       3 F       9.5  13.1  13.8
## 4       4 M      11.5  13.4  12.9
```

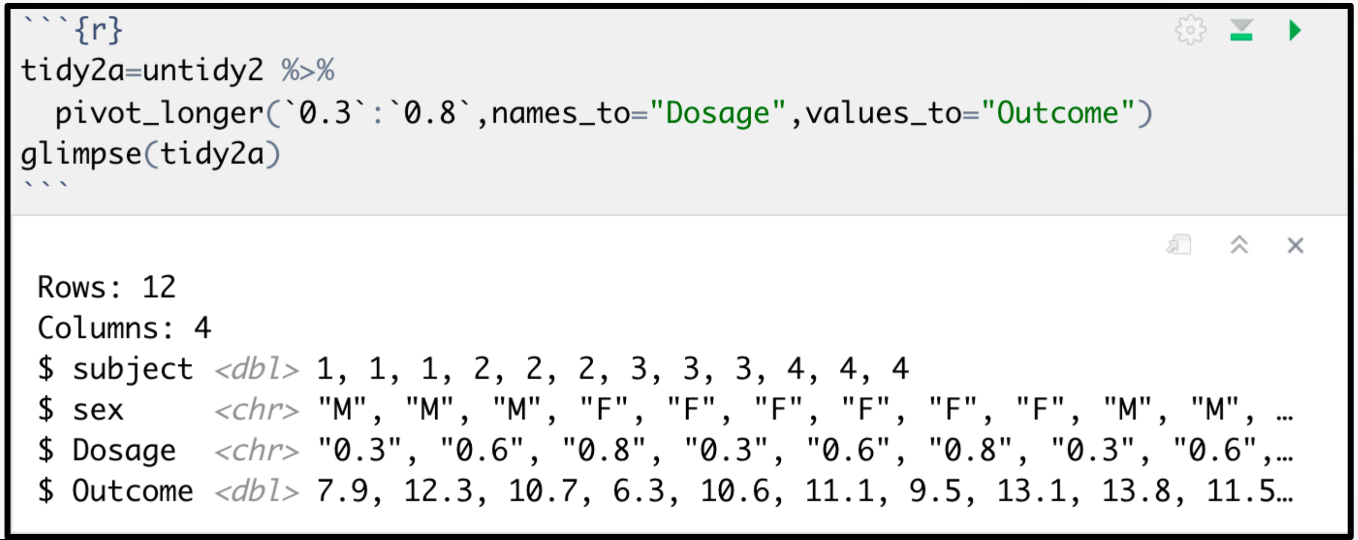
- Repeated Measures Data

- Variables “0.3”, “0.6”, and “0.8” are Measuring the Same Thing Under Different Drug Strengths

- The Name of the Variable Whose Values Form the Column Names Can Be Called “Dosage”

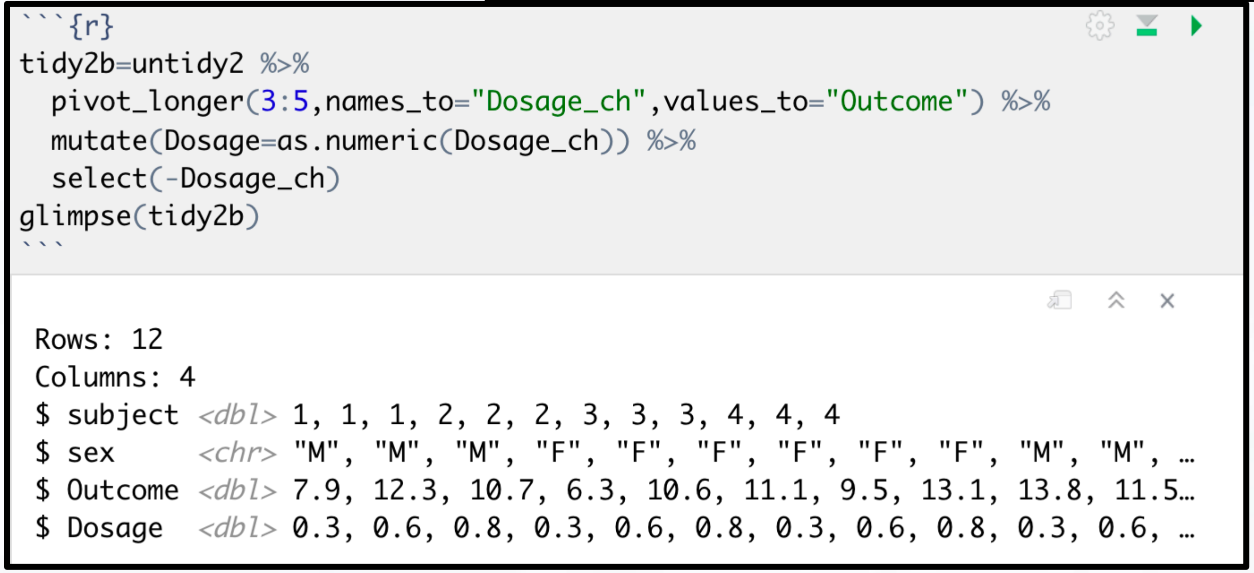
- The Name of the Variable Whose Values are Spread Over the Cells Can Be Called “Outcome”

Longer



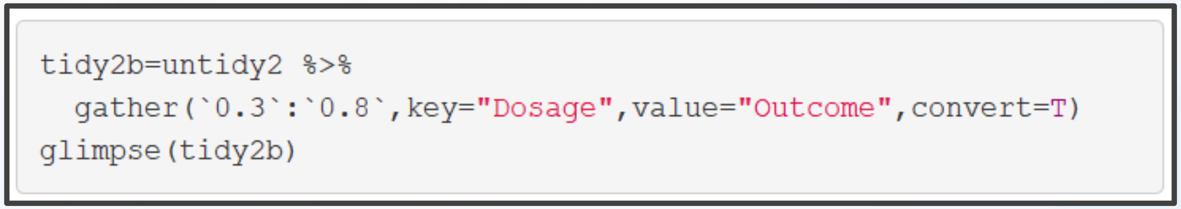
~~~
```{r}
tidy2a=untidy2 %>%
  pivot_longer(`0.3`:`0.8`,names_to="Dosage",values_to="Outcome")
glimpse(tidy2a)
```
~~~

```
Rows: 12
Columns: 4
$ subject <dbl> 1, 1, 1, 2, 2, 2, 3, 3, 3, 4, 4, 4
$ sex     <chr> "M", "M", "M", "F", "F", "F", "F", "F", "F", "M", "M", …
$ Dosage  <chr> "0.3", "0.6", "0.8", "0.3", "0.6", "0.8", "0.3", "0.6",…
$ Outcome <dbl> 7.9, 12.3, 10.7, 6.3, 10.6, 11.1, 9.5, 13.1, 13.8, 11.5…
```



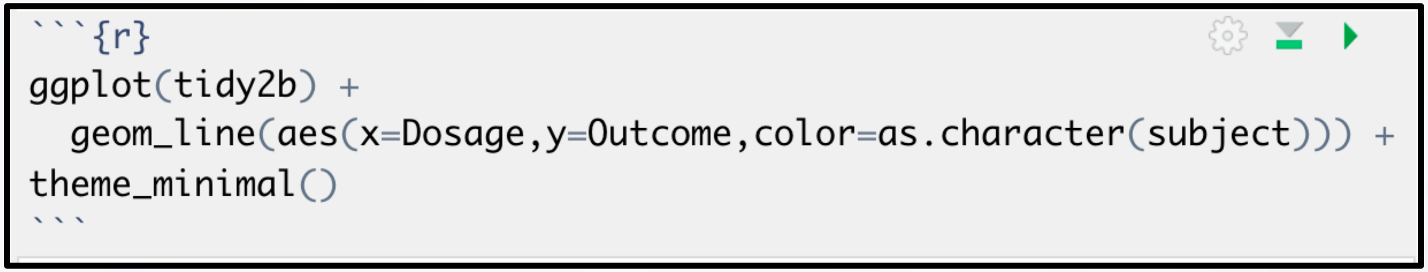
~~~
```{r}
tidy2b=untidy2 %>%
  pivot_longer(3:5,names_to="Dosage_ch",values_to="Outcome") %>%
  mutate(Dosage=as.numeric(Dosage_ch)) %>%
  select(-Dosage_ch)
glimpse(tidy2b)
```
~~~

```
Rows: 12
Columns: 4
$ subject <dbl> 1, 1, 1, 2, 2, 2, 3, 3, 3, 4, 4, 4
$ sex     <chr> "M", "M", "M", "F", "F", "F", "F", "F", "F", "M", "M", …
$ Outcome <dbl> 7.9, 12.3, 10.7, 6.3, 10.6, 11.1, 9.5, 13.1, 13.8, 11.5…
$ Dosage  <dbl> 0.3, 0.6, 0.8, 0.3, 0.6, 0.8, 0.3, 0.6, 0.8, 0.3, 0.6, …
```

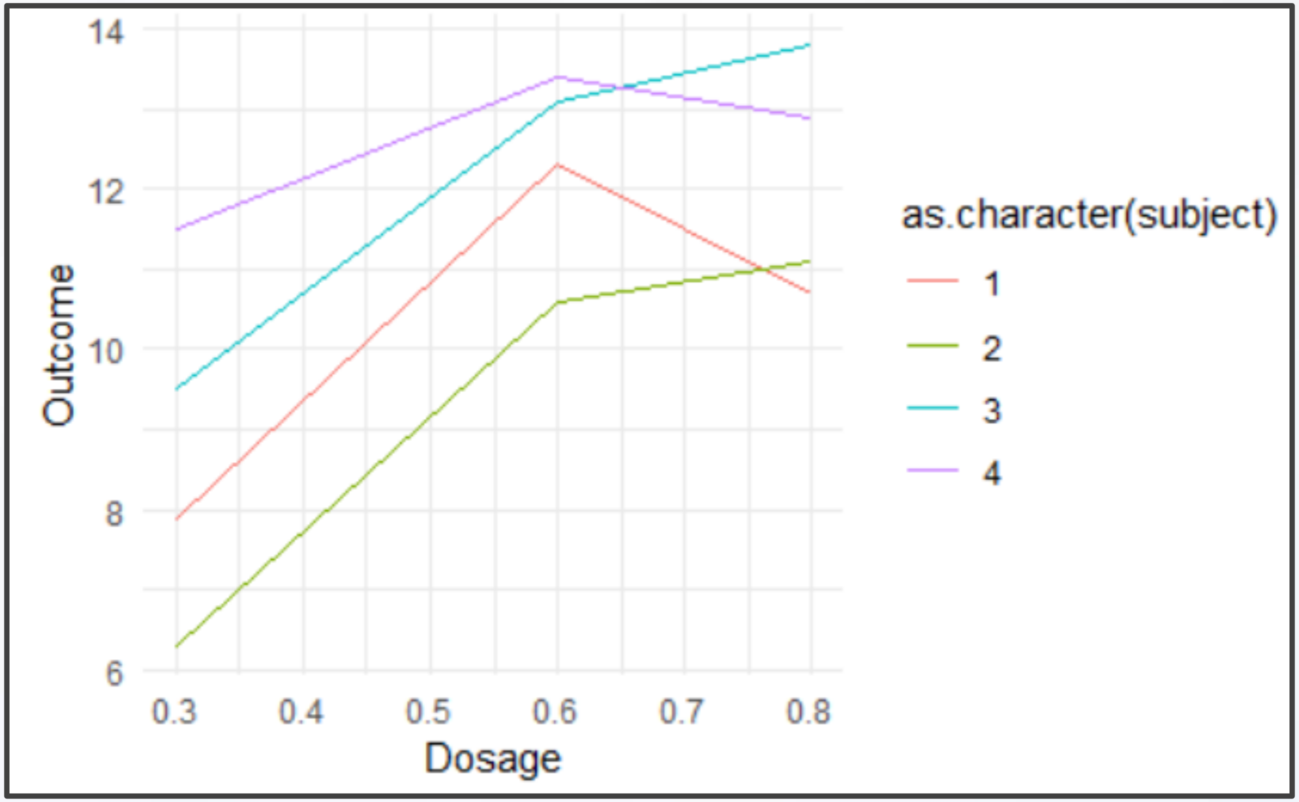


```
tidy2b=untidy2 %>%
  gather(`0.3`:`0.8`,key="Dosage",value="Outcome",convert=T)
glimpse(tidy2b)
```

Longer

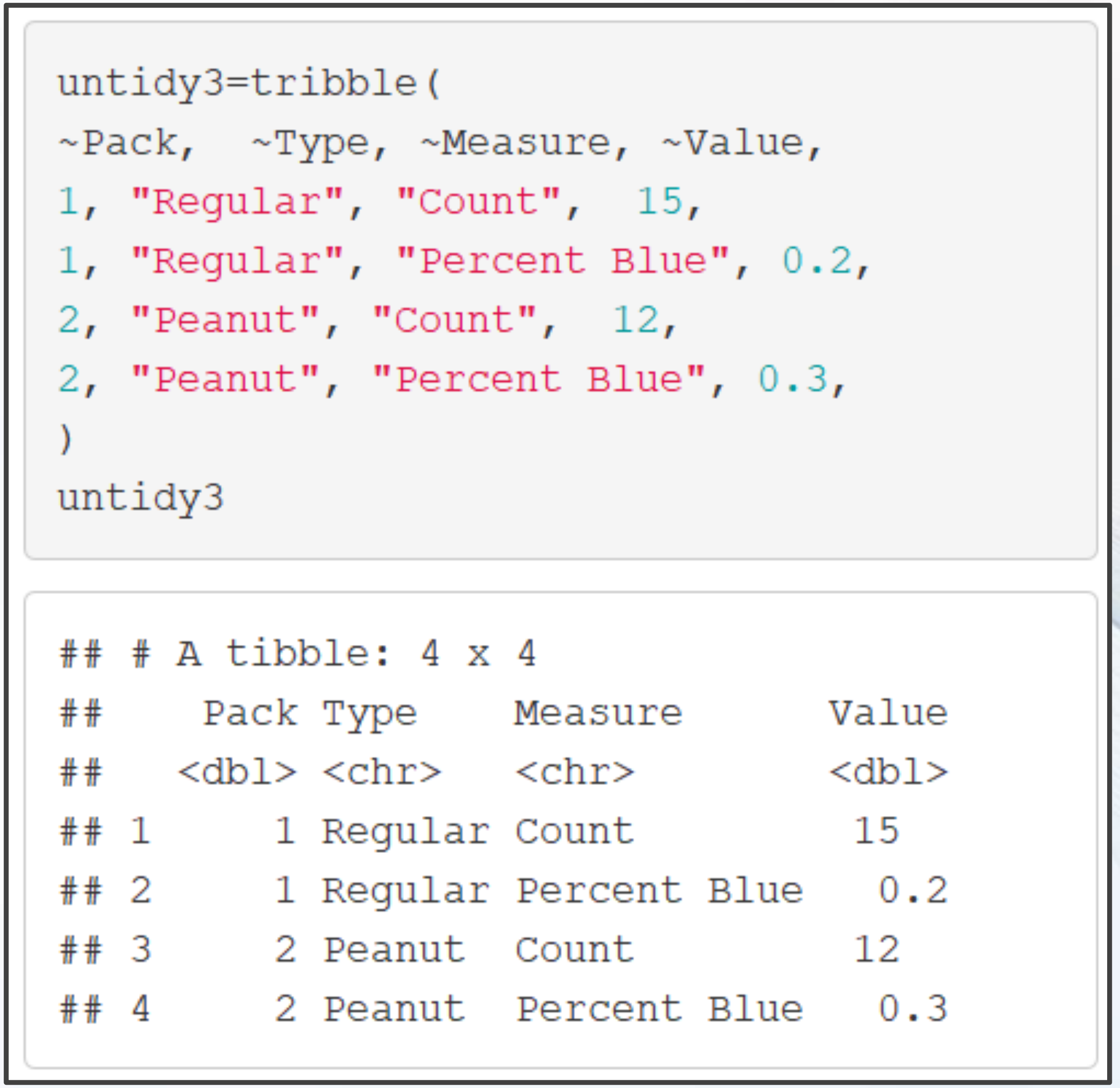


~~~
```{r}
ggplot(tidy2b) +
  geom_line(aes(x=Dosage,y=Outcome,color=as.character(subject))) +
theme_minimal()
```
~~~



Untidy Data Example 3

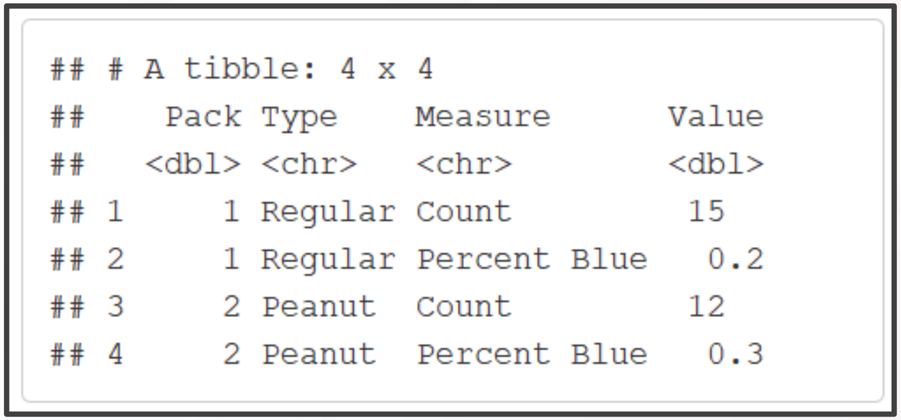
- Multiple rows



```
untidy3=tribble(
~Pack,  ~Type, ~Measure, ~Value,
1, "Regular", "Count",  15,
1, "Regular", "Percent Blue", 0.2,
2, "Peanut", "Count",  12,
2, "Peanut", "Percent Blue", 0.3,
)
untidy3
```

```
## # A tibble: 4 x 4
##    Pack Type    Measure       Value
##   <dbl> <chr>   <chr>         <dbl>
## 1     1 Regular Count          15  
## 2     1 Regular Percent Blue    0.2
## 3     2 Peanut  Count          12  
## 4     2 Peanut  Percent Blue    0.3
```

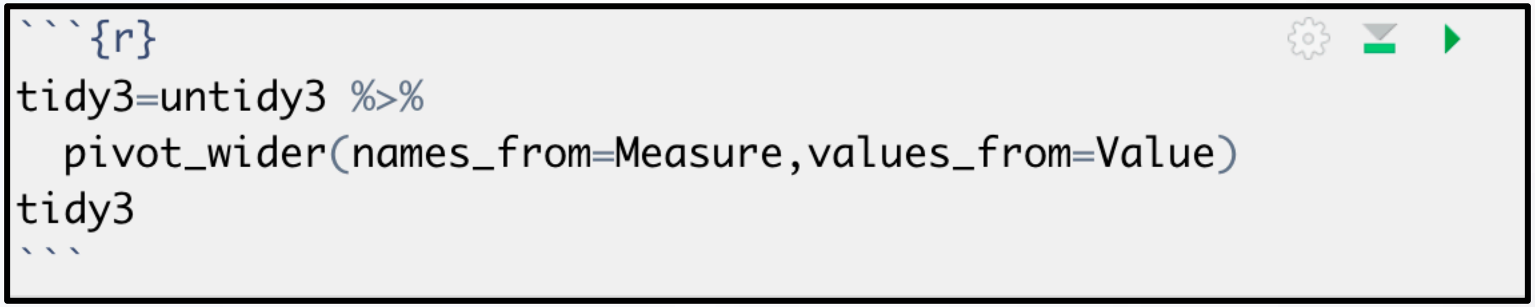
Problem



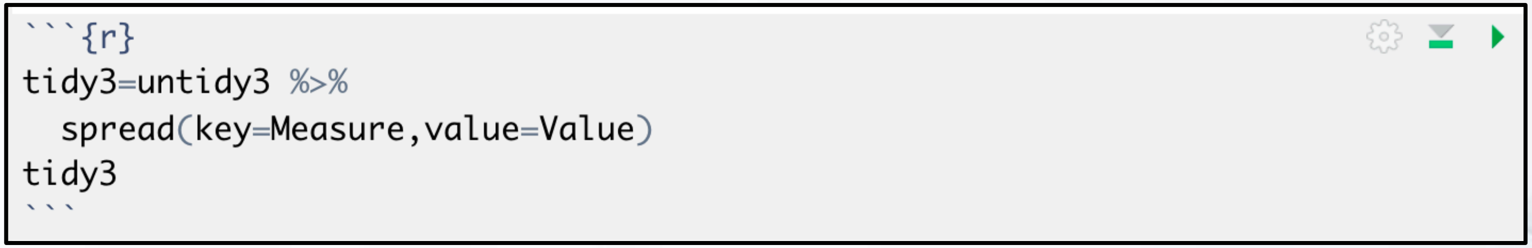
```
## # A tibble: 4 x 4
##    Pack Type    Measure      Value
##   <dbl> <chr>   <chr>        <dbl>
## 1     1 Regular Count         15  
## 2     1 Regular Percent Blue   0.2
## 3     2 Peanut  Count         12  
## 4     2 Peanut  Percent Blue   0.3
```

- Less Common
- Column “Measures” Contains Variable Names
- Column “Value” Contains the Output of the Different Variables
- Notice Values are of Different Units (Count vs Percentage)
- Wider Does the Opposite of Longer

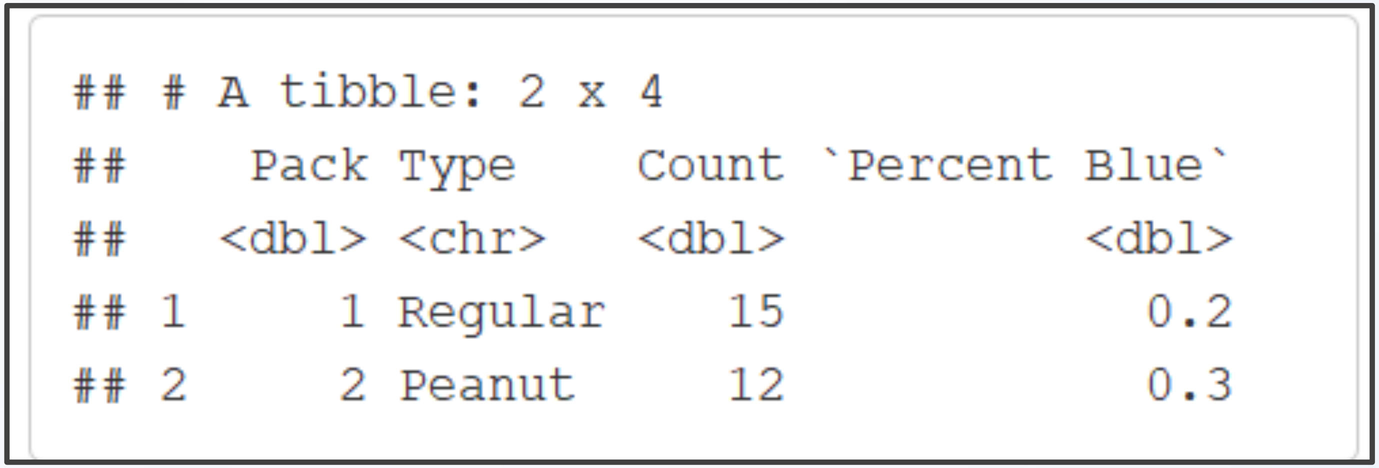
Wider



~~~
```{r}
tidy3=untidy3 %>%
  pivot_wider(names_from=Measure,values_from=Value)
tidy3
```
~~~

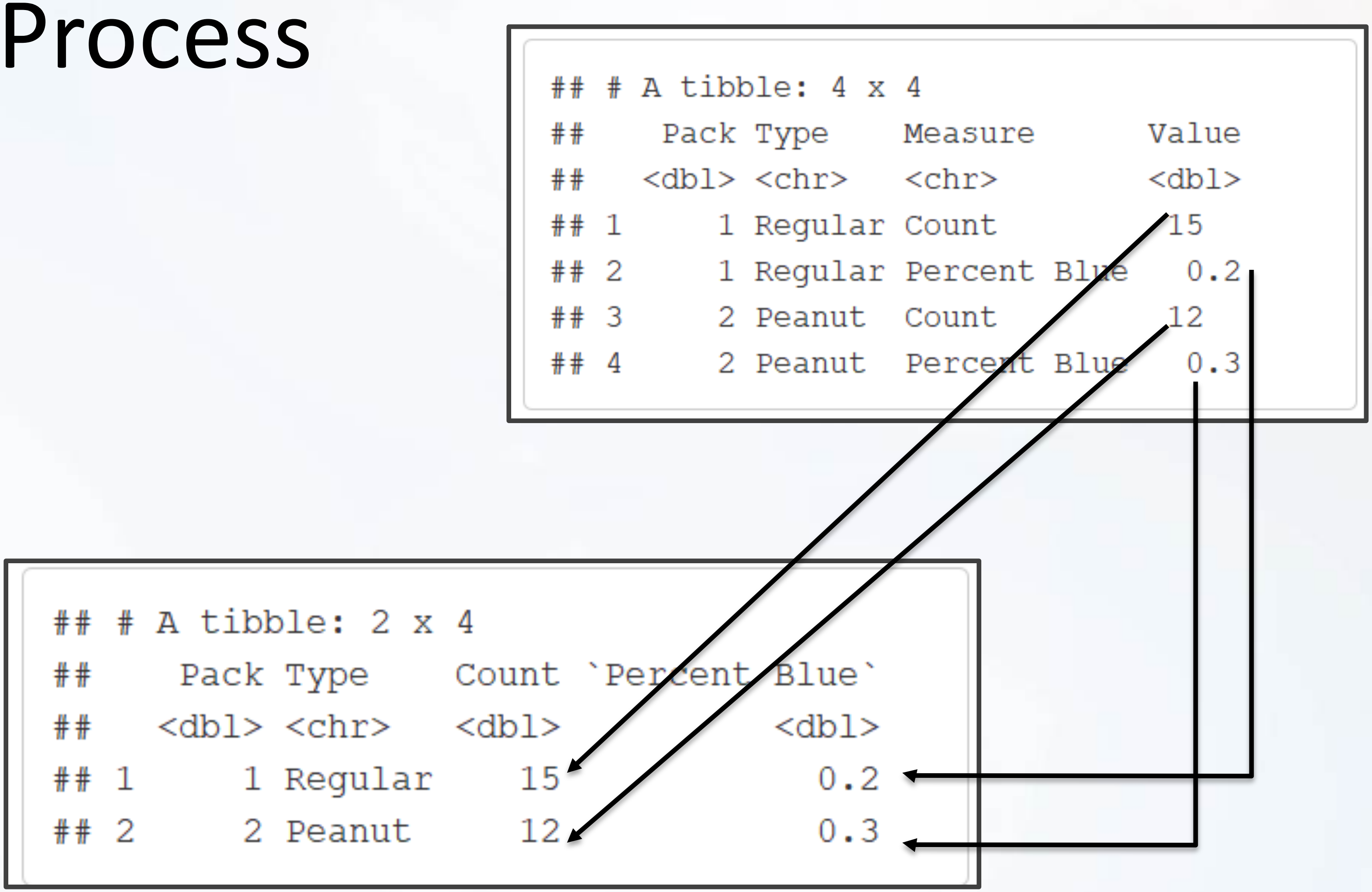


~~~
```{r}
tidy3=untidy3 %>%
  spread(key=Measure,value=Value)
tidy3
```
~~~



```
## # A tibble: 2 x 4
##    Pack Type    Count `Percent Blue`
##   <dbl> <chr>   <dbl>          <dbl>
## 1     1 Regular    15            0.2
## 2     2 Peanut     12            0.3
```

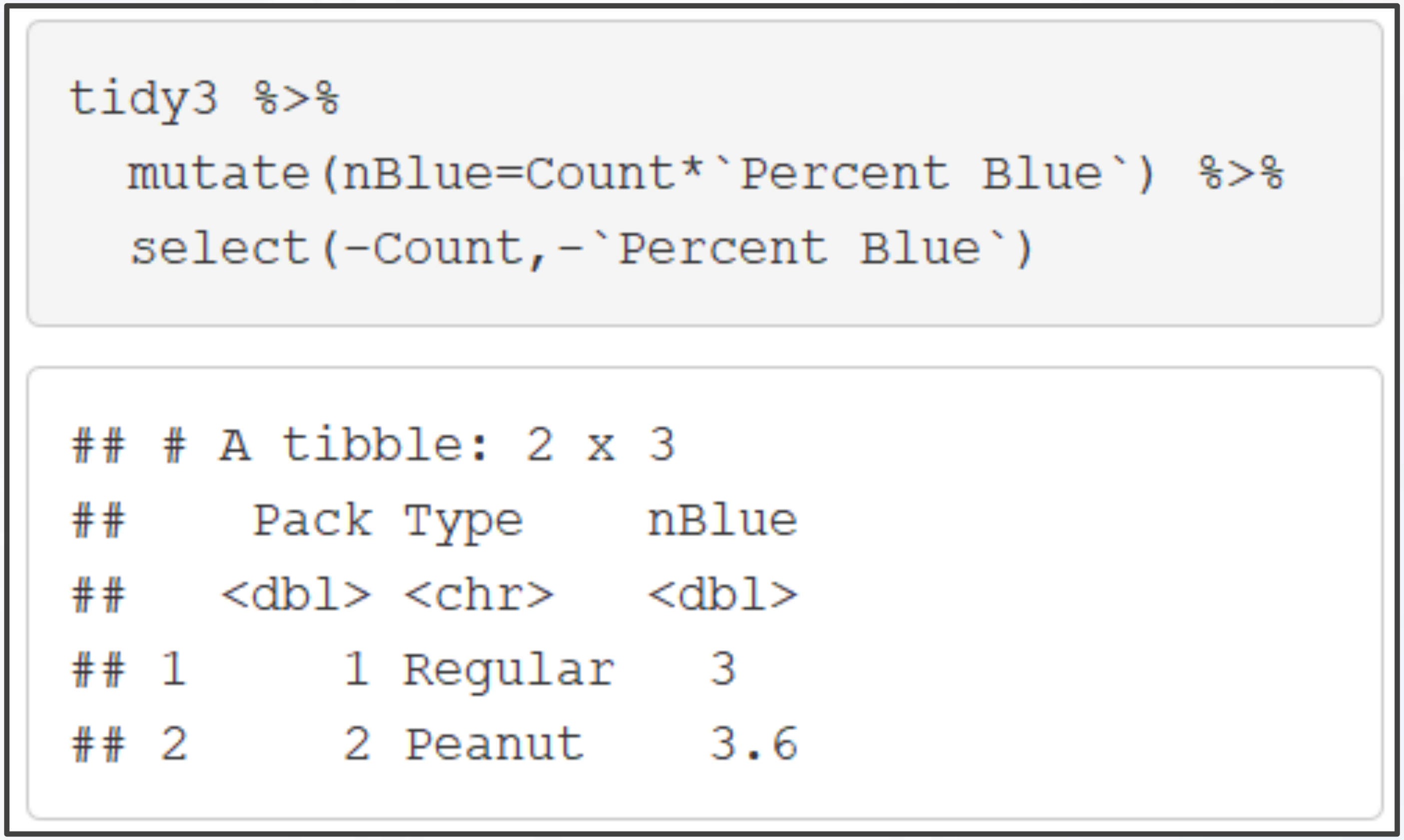
Process



```
## # A tibble: 4 x 4
##    Pack Type    Measure      Value
##   <dbl> <chr>   <chr>        <dbl>
## 1     1 Regular Count         15
## 2     1 Regular Percent Blue   0.2
## 3     2 Peanut  Count         12
## 4     2 Peanut  Percent Blue   0.3
```

```
## # A tibble: 2 x 4
##    Pack Type    Count `Percent Blue`
##   <dbl> <chr>   <dbl>          <dbl>
## 1     1 Regular    15            0.2
## 2     2 Peanut     12            0.3
```

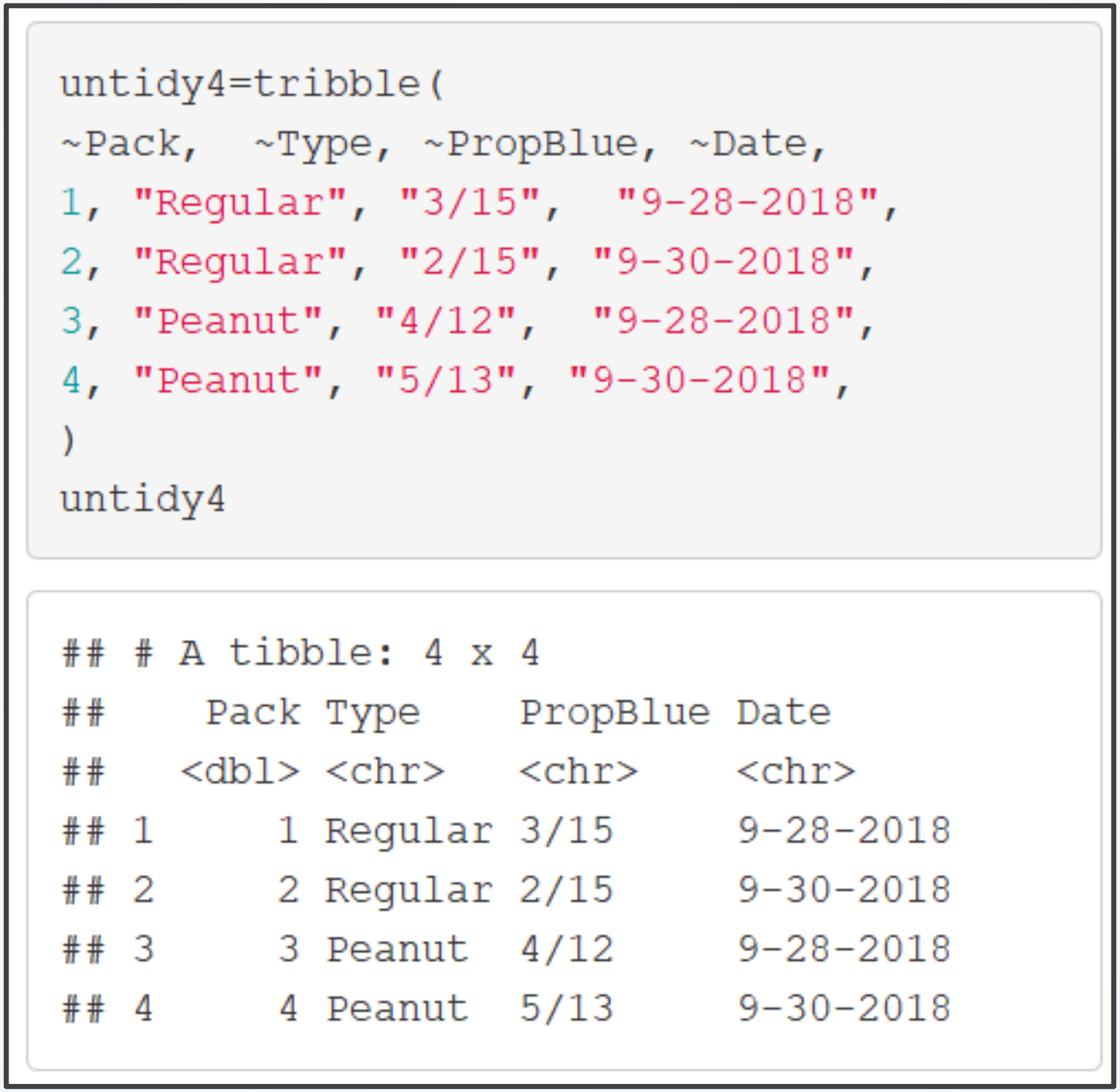
Wider



```
tidy3 %>%
  mutate(nBlue=Count*`Percent Blue`) %>%
  select(-Count,-`Percent Blue`)
```

```
## # A tibble: 2 x 3
##    Pack Type    nBlue
##   <dbl> <chr>   <dbl>
## 1     1 Regular   3
## 2     2 Peanut    3.6
```

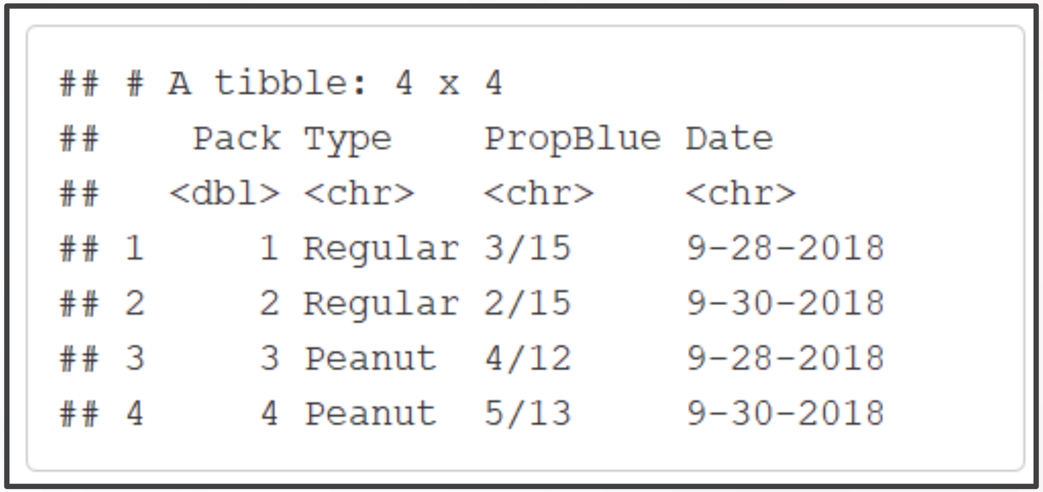
Untidy Data Example 4



```
untidy4=tribble(
~Pack,  ~Type, ~PropBlue, ~Date,
1, "Regular", "3/15",  "9-28-2018",
2, "Regular", "2/15", "9-30-2018",
3, "Peanut", "4/12",  "9-28-2018",
4, "Peanut", "5/13", "9-30-2018",
)
untidy4
```

```
## # A tibble: 4 x 4
##     Pack Type    PropBlue Date
##    <dbl> <chr>   <chr>    <chr>
## 1      1 Regular 3/15     9-28-2018
## 2      2 Regular 2/15     9-30-2018
## 3      3 Peanut  4/12     9-28-2018
## 4      4 Peanut  5/13     9-30-2018
```

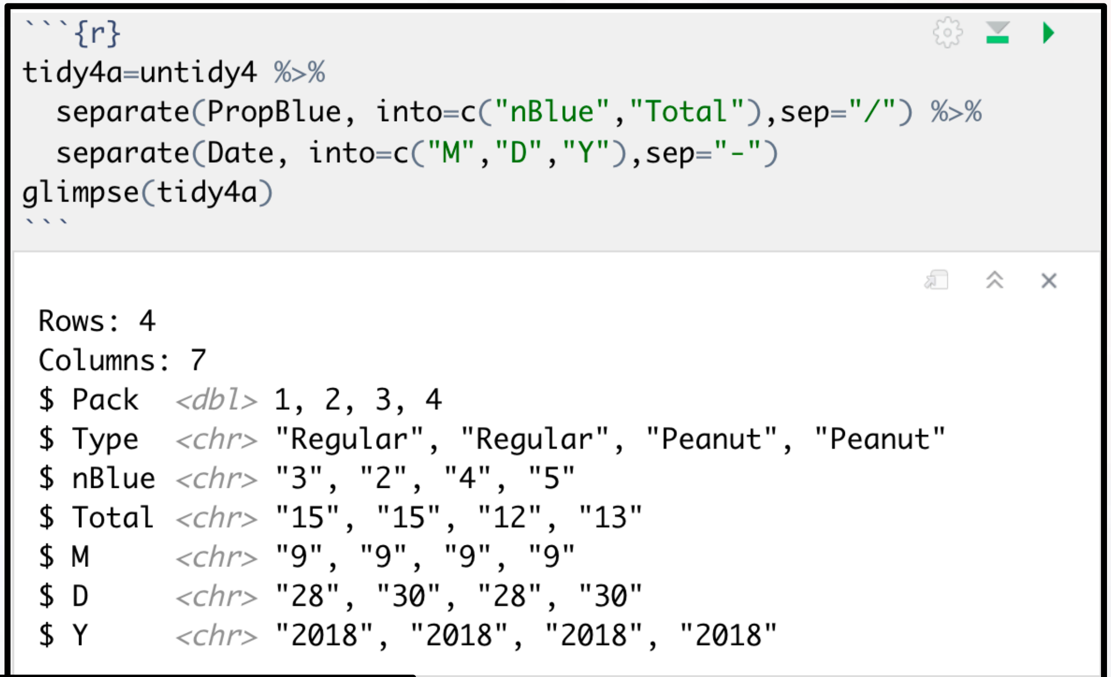
Problem



```
## # A tibble: 4 x 4
##    Pack Type    PropBlue Date
##   <dbl> <chr>   <chr>    <chr>
## 1     1 Regular 3/15     9-28-2018
## 2     2 Regular 2/15     9-30-2018
## 3     3 Peanut  4/12     9-28-2018
## 4     4 Peanut  5/13     9-30-2018
```

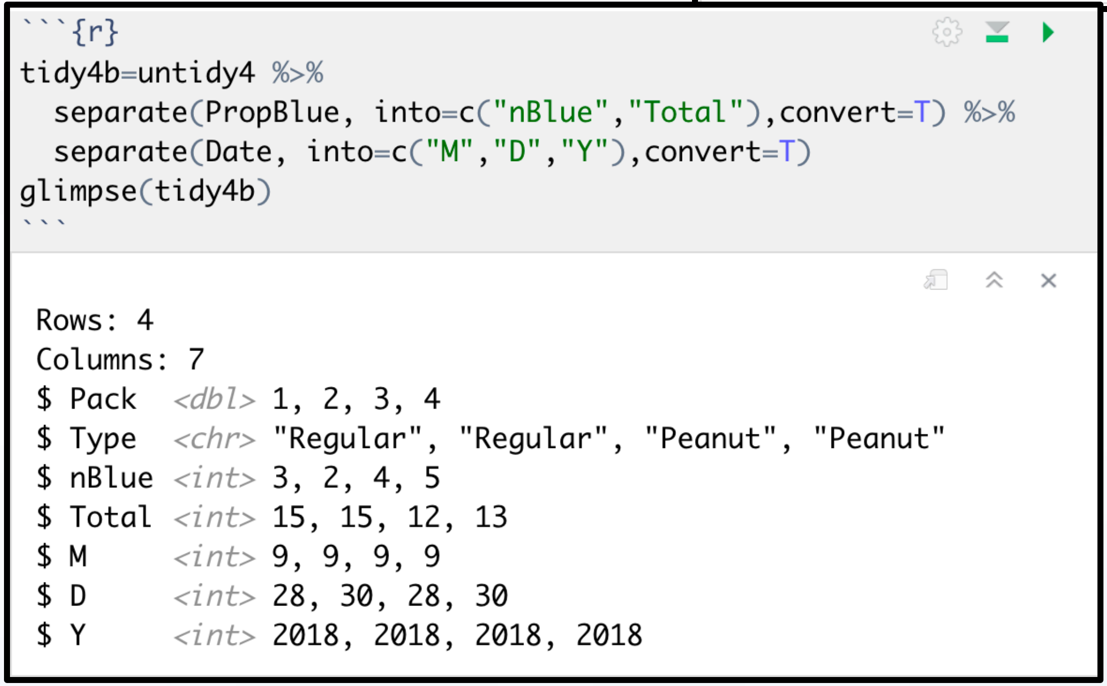
- Very Uncommon
- The Variable “PropBlue” Contains Two Numeric Variables
- The Variable “Date” Contains Three Numeric Variables
- We Must Separate Both of These Variables Into Multiple Columns

Separating



~~~
```{r}
tidy4a=untidy4 %>%
  separate(PropBlue, into=c("nBlue","Total"),sep="/") %>%
  separate(Date, into=c("M","D","Y"),sep="-")
glimpse(tidy4a)
```
~~~

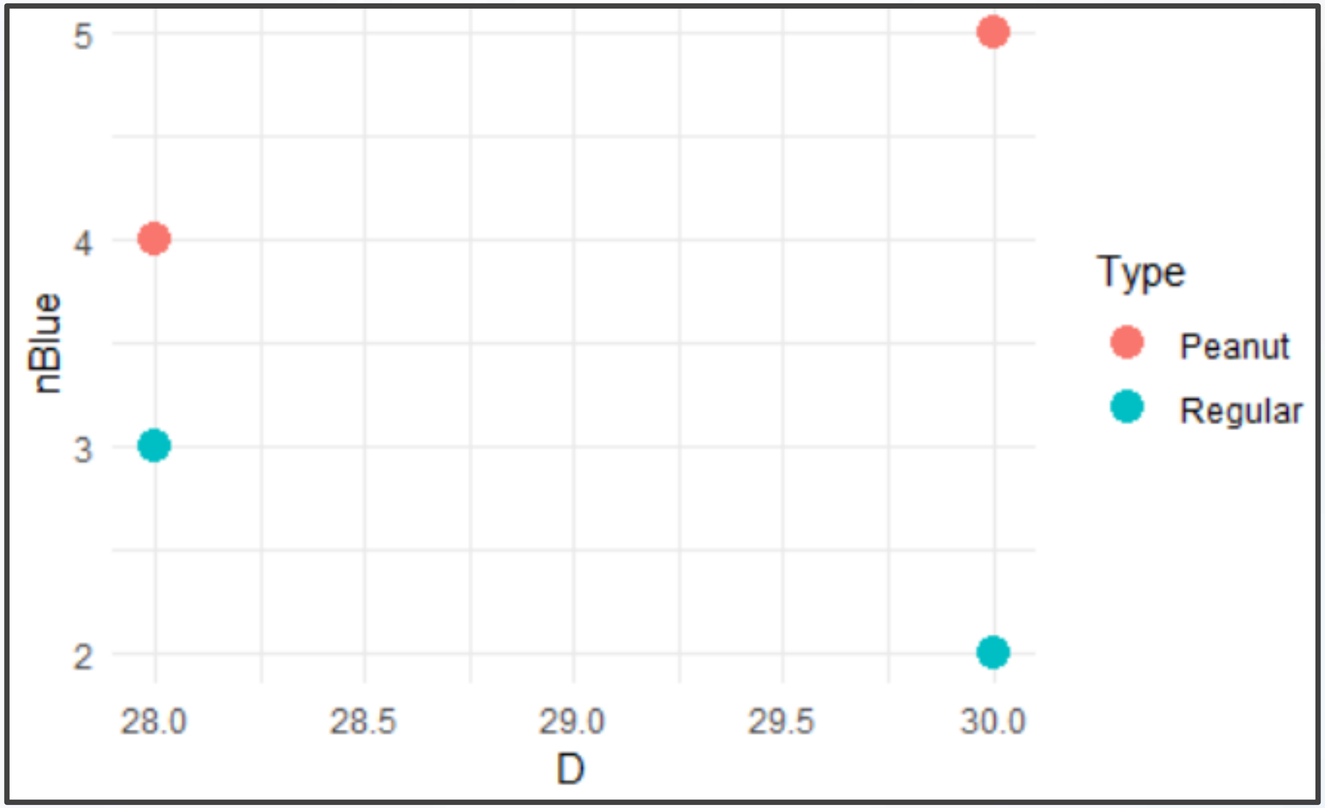
```
Rows: 4
Columns: 7
$ Pack  <dbl> 1, 2, 3, 4
$ Type  <chr> "Regular", "Regular", "Peanut", "Peanut"
$ nBlue <chr> "3", "2", "4", "5"
$ Total <chr> "15", "15", "12", "13"
$ M     <chr> "9", "9", "9", "9"
$ D     <chr> "28", "30", "28", "30"
$ Y     <chr> "2018", "2018", "2018", "2018"
```



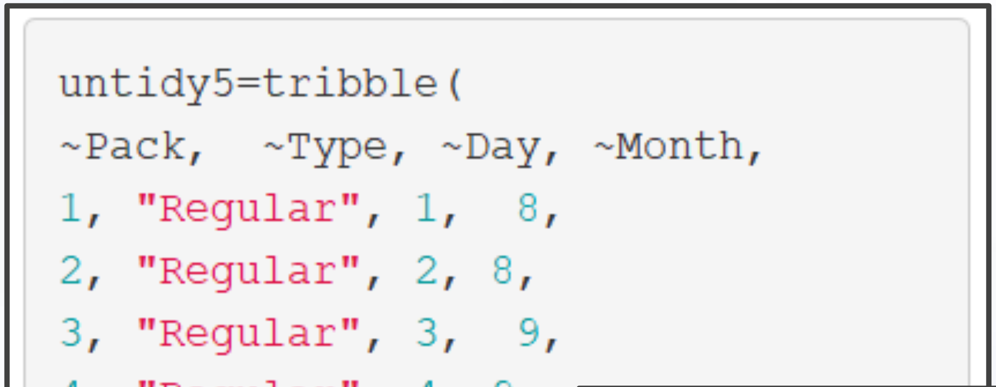
~~~
```{r}
tidy4b=untidy4 %>%
  separate(PropBlue, into=c("nBlue","Total"),convert=T) %>%
  separate(Date, into=c("M","D","Y"),convert=T)
glimpse(tidy4b)
```
~~~

```
Rows: 4
Columns: 7
$ Pack  <dbl> 1, 2, 3, 4
$ Type  <chr> "Regular", "Regular", "Peanut", "Peanut"
$ nBlue <int> 3, 2, 4, 5
$ Total <int> 15, 15, 12, 13
$ M     <int> 9, 9, 9, 9
$ D     <int> 28, 30, 28, 30
$ Y     <int> 2018, 2018, 2018, 2018
```

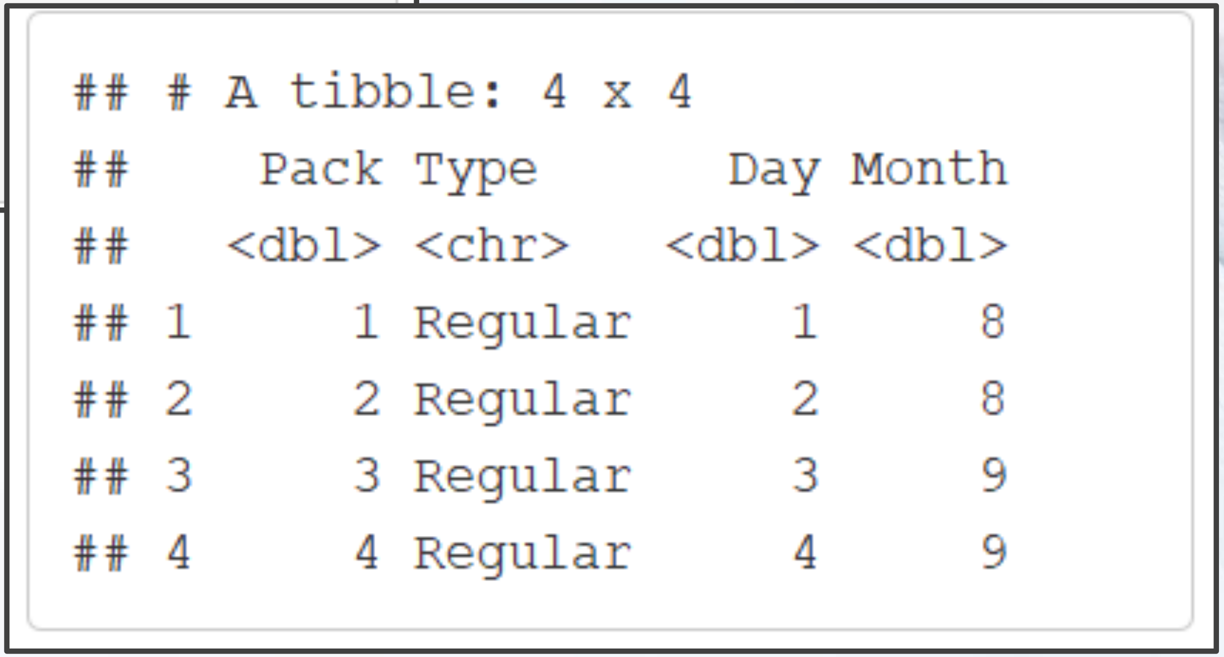
Separating



Untidy Data Example 5



```
untidy5=tribble(
~Pack,  ~Type, ~Day, ~Month,
1, "Regular", 1,   8,
2, "Regular", 2, 8,
3, "Regular", 3,   9,
4, "Regular", 4, 9,
)
untidy5
```

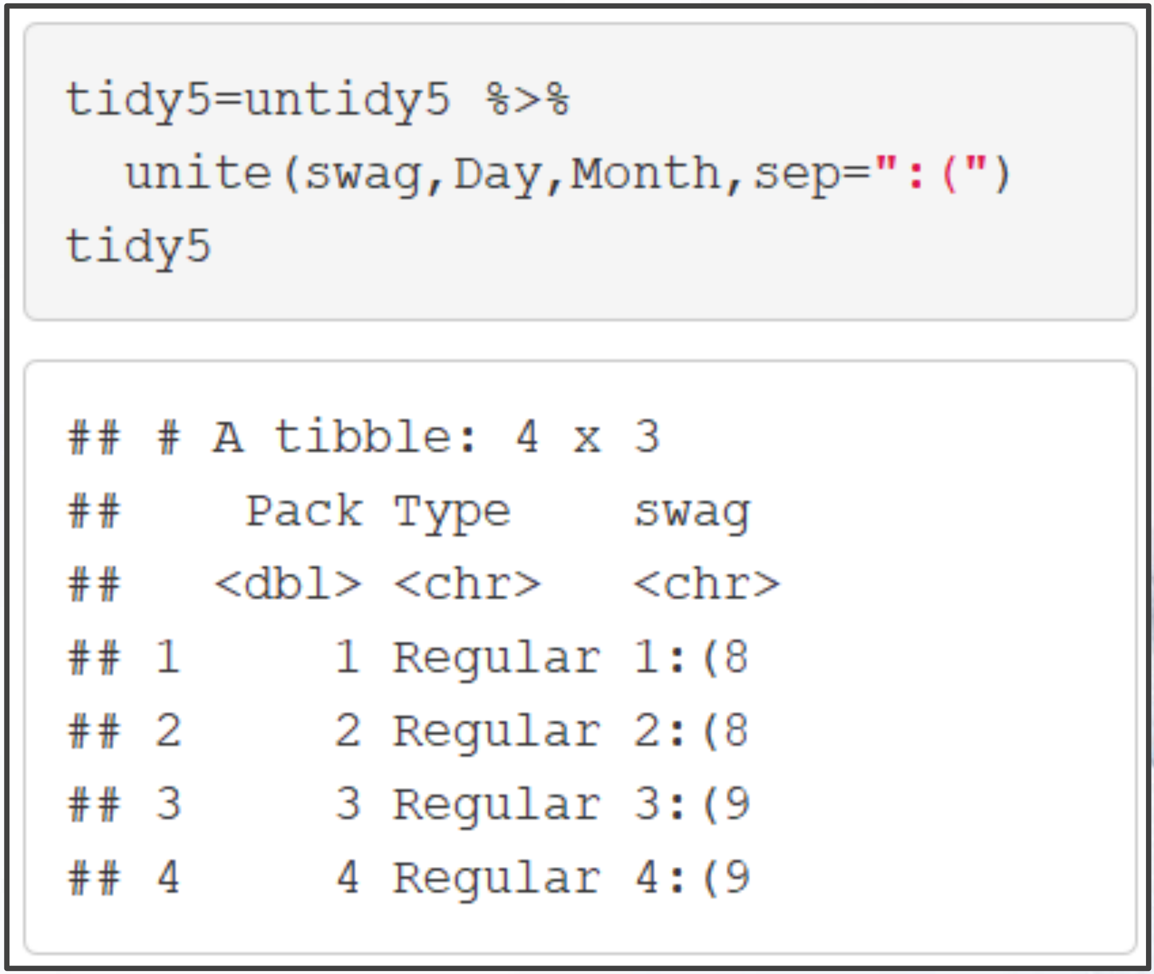


```
## # A tibble: 4 x 4
##    Pack Type      Day Month
##   <dbl> <chr>   <dbl> <dbl>
## 1     1 Regular     1     8
## 2     2 Regular     2     8
## 3     3 Regular     3     9
## 4     4 Regular     4     9
```

Uniting

- Absolutely Silly

- Uniting Does the Opposite of Separating



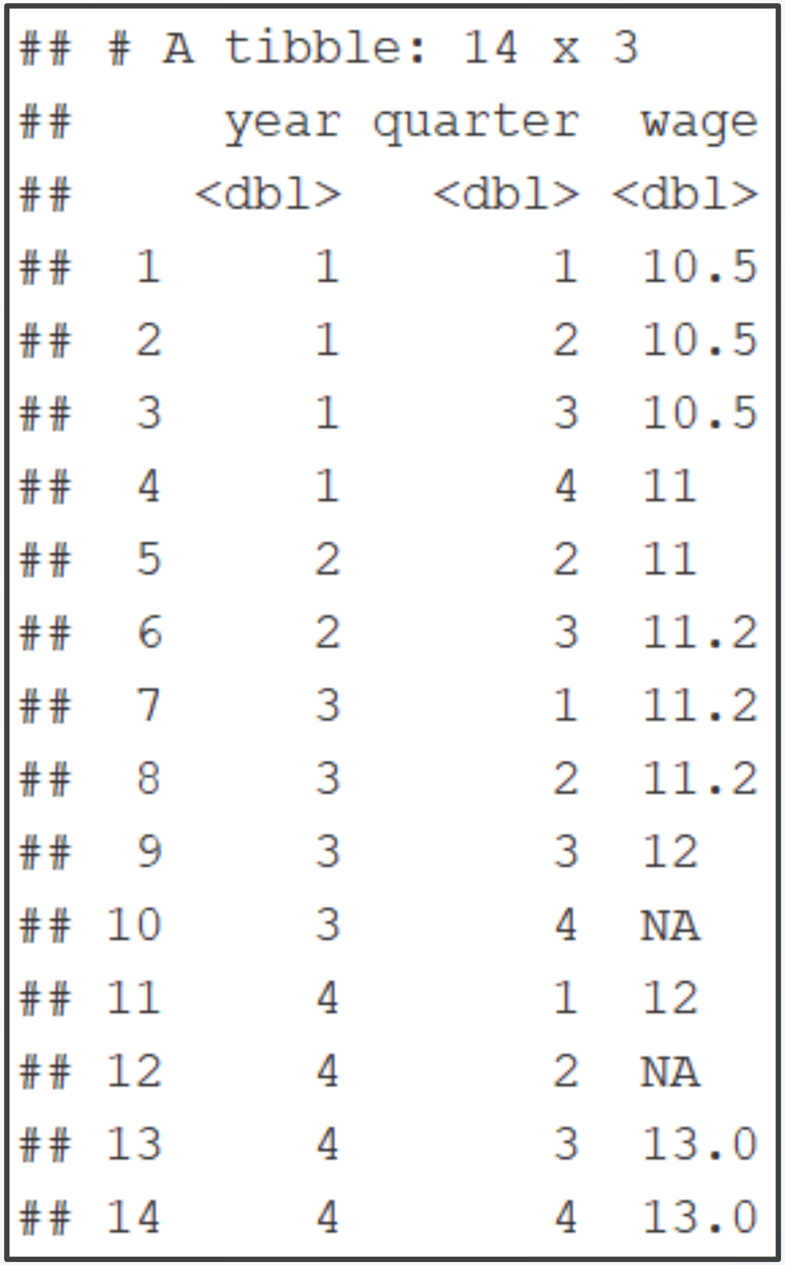
```
tidy5=untidy5 %>%
  unite(swag,Day,Month,sep=":(")
tidy5
```

```
## # A tibble: 4 x 3
##     Pack Type    swag
##    <dbl> <chr>   <chr>
## 1      1 Regular 1:(8
## 2      2 Regular 2:(8
## 3      3 Regular 3:(9
## 4      4 Regular 4:(9
```

Missing Values

- Two Ways
  - Explicitly: Defined to Be Missing Using NA
  - Implicitly: Absent From Data
- There is not a Uniform Way to Handle Either of These Problems
- Rule: Either Convert All Explicitly Missing to Implicitly Missing or Convert All Implicitly Missing to Explicitly Missing

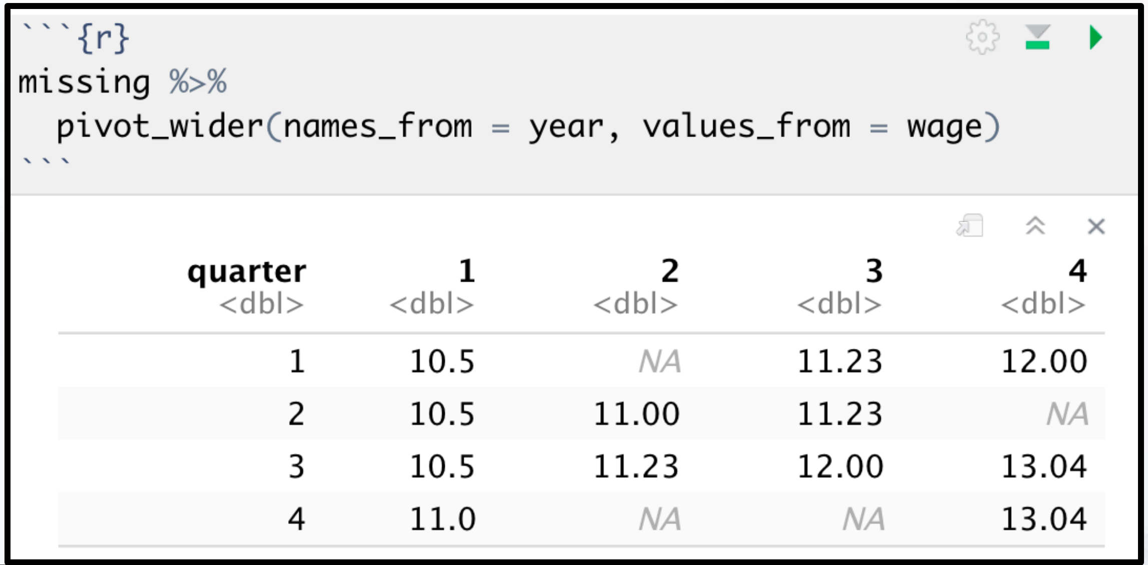
Example



```
## # A tibble: 14 x 3
##     year quarter  wage
##    <dbl>   <dbl> <dbl>
##  1     1       1  10.5
##  2     1       2  10.5
##  3     1       3  10.5
##  4     1       4  11
##  5     2       2  11
##  6     2       3  11.2
##  7     3       1  11.2
##  8     3       2  11.2
##  9     3       3  12
## 10     3       4  NA
## 11     4       1  12
## 12     4       2  NA
## 13     4       3  13.0
## 14     4       4  13.0
```

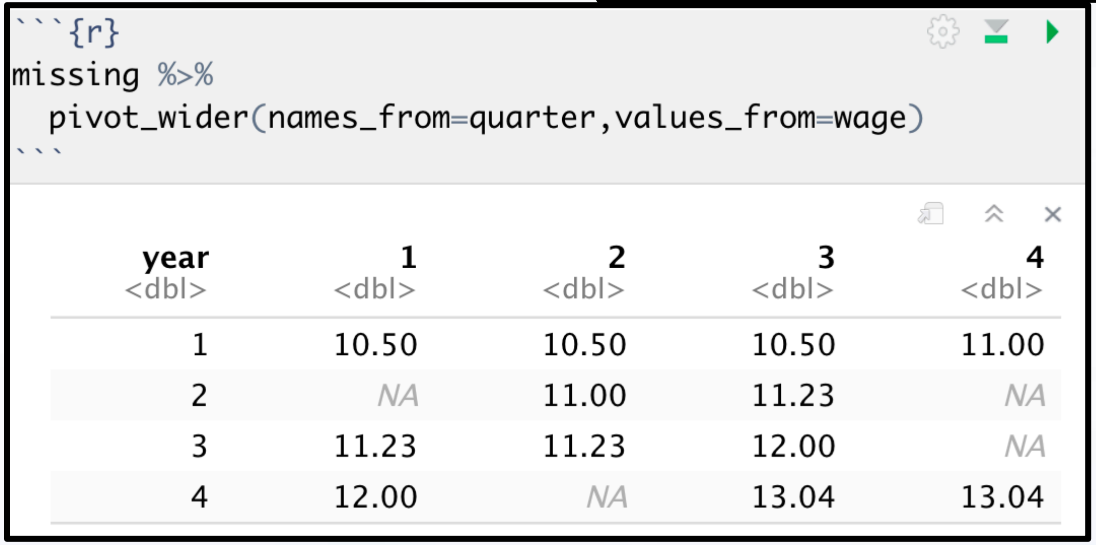
Missing Values

- Notice:



~~~
```{r}
missing %>%
  pivot_wider(names_from = year, values_from = wage)
```
~~~

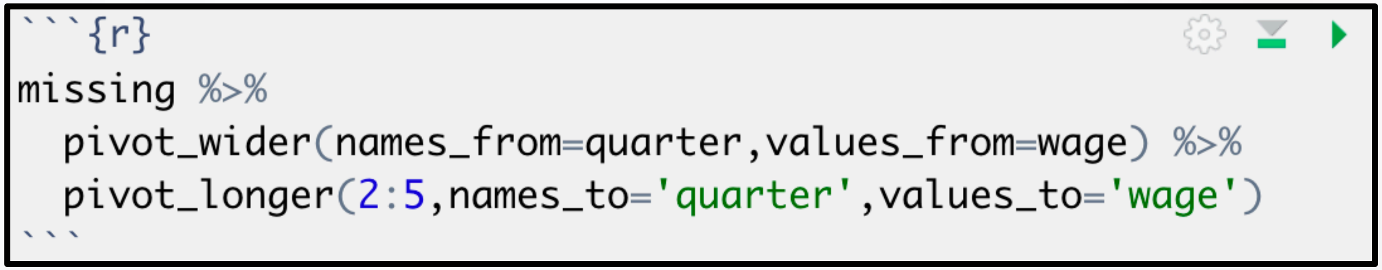
quarter <dbl>	1 <dbl>	2 <dbl>	3 <dbl>	4 <dbl>
1	10.5	*NA*	11.23	12.00
2	10.5	11.00	11.23	*NA*
3	10.5	11.23	12.00	13.04
4	11.0	*NA*	*NA*	13.04



~~~
```{r}
missing %>%
  pivot_wider(names_from=quarter,values_from=wage)
```
~~~

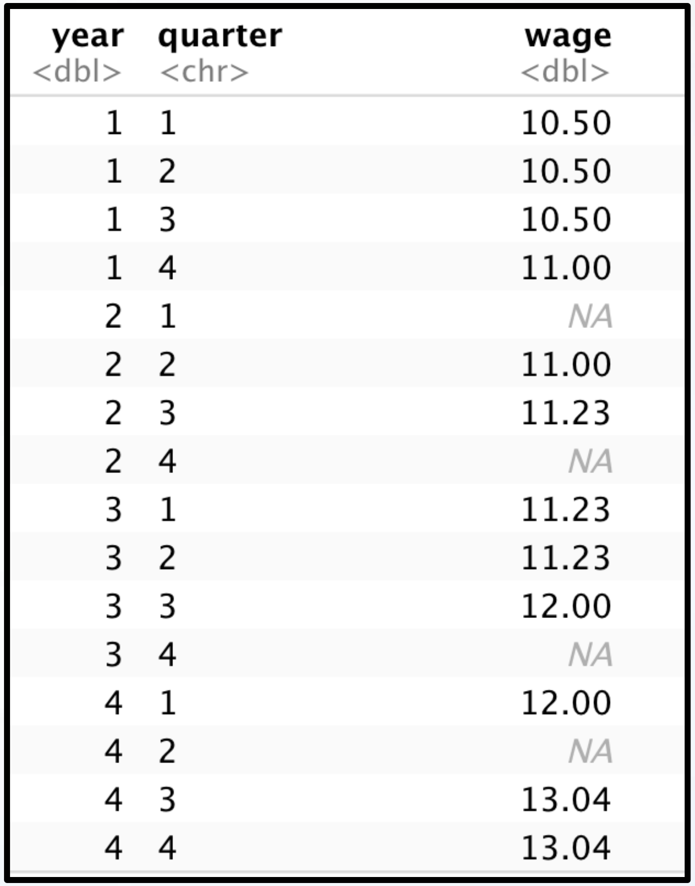
year <dbl>	1 <dbl>	2 <dbl>	3 <dbl>	4 <dbl>
1	10.50	10.50	10.50	11.00
2	*NA*	11.00	11.23	*NA*
3	11.23	11.23	12.00	*NA*
4	12.00	*NA*	13.04	13.04

Missing Values



~~~
```{r}
missing %>%
  pivot_wider(names_from=quarter,values_from=wage) %>%
  pivot_longer(2:5,names_to='quarter',values_to='wage')
```
~~~

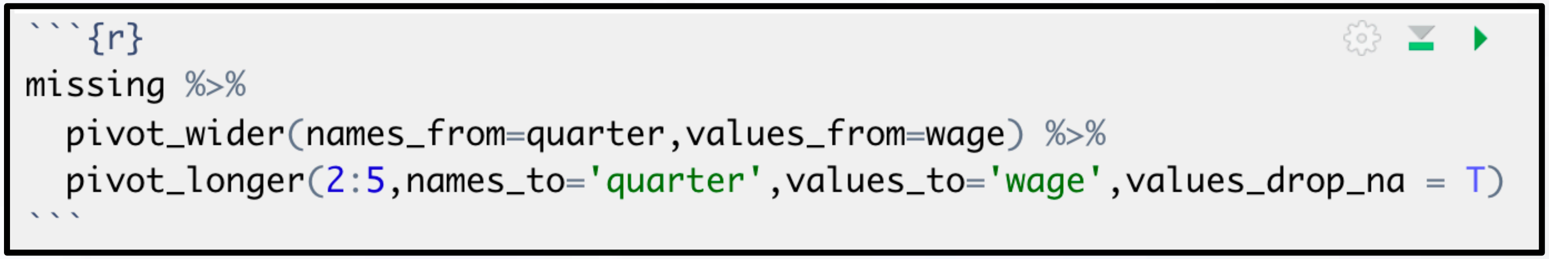
- Implicit to Explicit



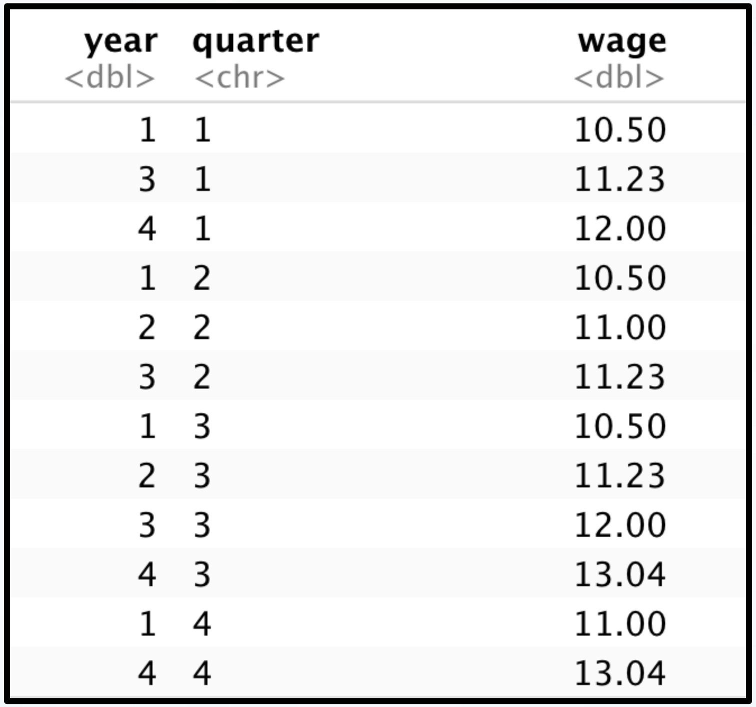
year	quarter	wage
<dbl>	<chr>	<dbl>
1	1	10.50
1	2	10.50
1	3	10.50
1	4	11.00
2	1	NA
2	2	11.00
2	3	11.23
2	4	NA
3	1	11.23
3	2	11.23
3	3	12.00
3	4	NA
4	1	12.00
4	2	NA
4	3	13.04
4	4	13.04

Missing Values

- Explicit to Implicit



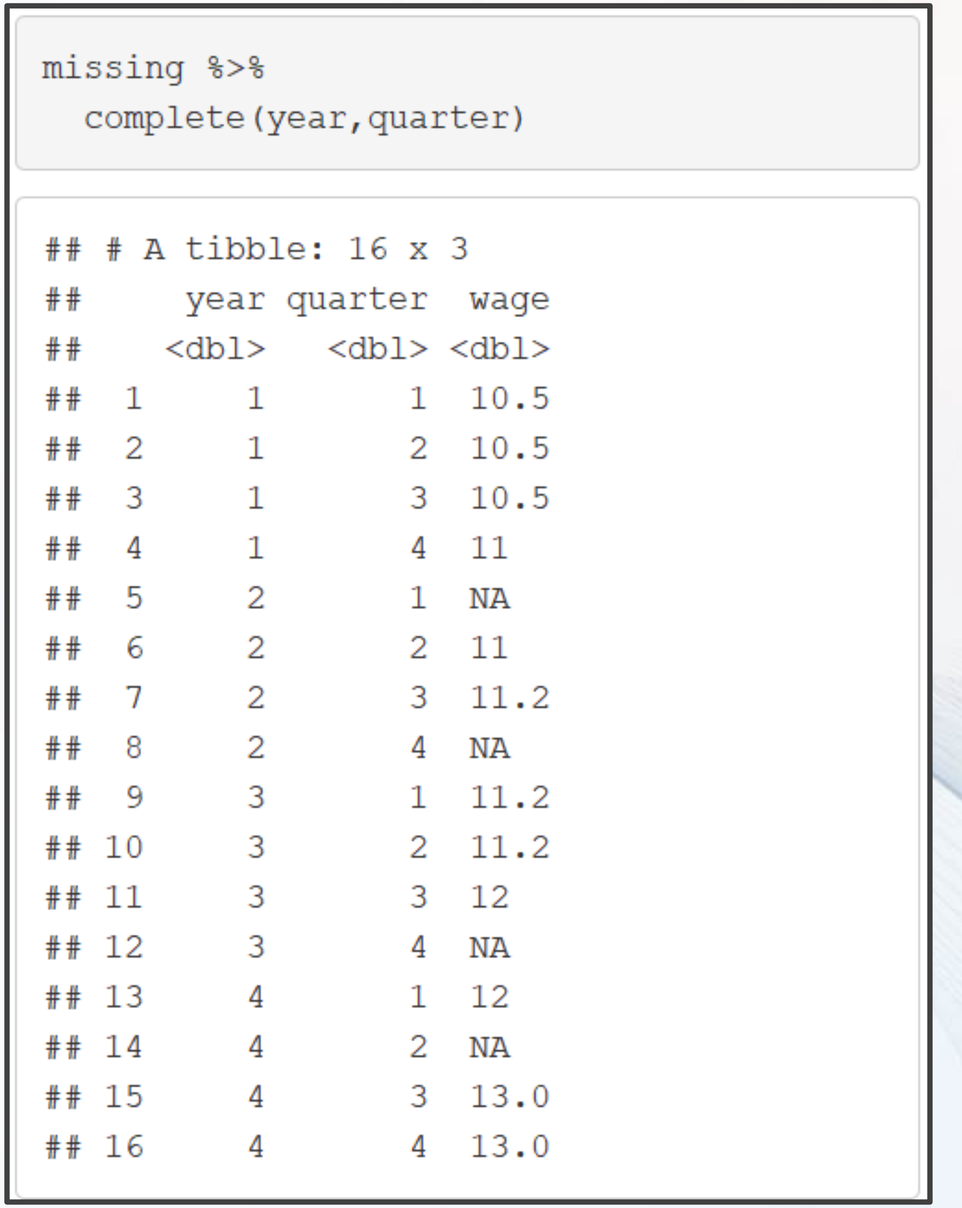
~~~
```{r}
missing %>%
  pivot_wider(names_from=quarter,values_from=wage) %>%
  pivot_longer(2:5,names_to='quarter',values_to='wage',values_drop_na = T)
```
~~~



year	quarter	wage
<dbl>	<chr>	<dbl>
1	1	10.50
3	1	11.23
4	1	12.00
1	2	10.50
2	2	11.00
3	2	11.23
1	3	10.50
2	3	11.23
3	3	12.00
4	3	13.04
1	4	11.00
4	4	13.04

Missing Values

- Complete Function



```
missing %>%
  complete(year,quarter)
```

```
## # A tibble: 16 x 3
##     year quarter  wage
##    <dbl>   <dbl> <dbl>
##  1     1       1  10.5
##  2     1       2  10.5
##  3     1       3  10.5
##  4     1       4  11
##  5     2       1  NA
##  6     2       2  11
##  7     2       3  11.2
##  8     2       4  NA
##  9     3       1  11.2
## 10     3       2  11.2
## 11     3       3  12
## 12     3       4  NA
## 13     4       1  12
## 14     4       2  NA
## 15     4       3  13.0
## 16     4       4  13.0
```