

Recasting and joining of dataframes

In this lecture

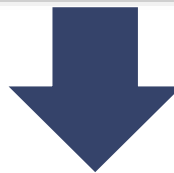
- Recasting
- Need to recast dataframes
- Recast in 2 steps
 - Melt
 - Cast
- Recast in 1 step –recast
- Joining of two dataframes
 - Left join, Right join, Inner join

Recasting dataframes

- Recasting is the process of manipulating a data frame in terms of its variables
- Reshaping the data
 - insights

Dataframe – “pd”

	Name	Month	BS	BP
1	Senthil	Jan	141.2	90
2	Senthil	Feb	139.3	78
3	Sam	Jan	135.2	80
4	Sam	Feb	160.1	81



	variable	Month	Sam	Senthil
1	BS	Feb	160.1	139.3
2	BS	Jan	135.2	141.2
3	BP	Feb	81.0	78.0
4	BP	Jan	80.0	90.0

Recast in two steps: Example

- Create the following example : dataframe 'pd'

	Name	Month	BS	BP
1	Senthil	Jan	141.2	90
2	Senthil	Feb	139.3	78
3	Sam	Jan	135.2	80
4	Sam	Feb	160.1	81

Code

```
# Data frame example 2
pd=data.frame("Name"=c("Senthil","
Senthil","Sam","Sam"),
"Month"=c("Jan","Feb","Jan","Feb"),
"BS" = c(141.2,139.3,135.2,160.1),
"BP" = c(90,78,80,81))
```

```
print(pd)
```

Console Output

```
> pd=data.frame("Name"=c("Senthil",
"1","Senthil","Sam","Sam"),
+               "Month"=c("Jan",
"Feb","Jan","Feb"),
+               "BS" = c(141.2,1
39.3,135.2,160.1),
+               "BP" = c(90,78,8
0,81))
> print(pd)
      Name Month      BS BP
1 Senthil   Jan 141.2  90
2 Senthil   Feb 139.3  78
3      Sam   Jan 135.2  80
4      Sam   Feb 160.1  81
> |
```

Recast in two steps: Example

- Two steps
 - Melt
 - Cast
- Identifier (Discrete type variables)
- Measurements (numeric variables)
- Categorical and Date variables can not be measurements

	Name Month		BS BP	
1	Senthil	Jan	141.2	90
2	Senthil	Feb	139.3	78
3	Sam	Jan	135.2	80
4	Sam	Feb	160.1	81

Identifier variables *measurement variables*

Step 1: Melt

Call the library 'reshape2' using the library() command
 melt (data, id.vars, measure.vars, variable.name = "variable", value.name = "value")

Code

```
# Data frame example 3

# melt operation sample code

library(reshape2)

Df = melt(pd, id.vars = c("Name","Month"),
measure.vars = c("BS", "BP"))

print(Df)
```

Console Output

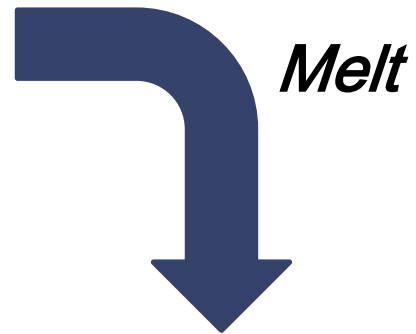
```
Console ~/
> # Data frame example 3
> # melt operation sample code
> library(reshape2)
> Df = melt(pd,
+ id.vars = c("Name","Month"),
+ measure.vars = c("BS", "BP"))
> print(Df)
      Name Month variable value
1 Senthil   Jan        BS  141.2
2 Senthil   Feb        BS  139.3
3      Sam   Jan        BS  135.2
4      Sam   Feb        BS  160.1
5 Senthil   Jan        BP   90.0
6 Senthil   Feb        BP   78.0
7      Sam   Jan        BP   80.0
8      Sam   Feb        BP   81.0
>
```

Step 1: melt

	Name Month		BS BP	
1	Senthil	Jan	141.2	90
2	Senthil	Feb	139.3	78
3	Sam	Jan	135.2	80
4	Sam	Feb	160.1	81

Identifier variables

measurement variables



	Name	Month	variable	value
1	Senthil	Jan	BS	141.2
2	Senthil	Feb	BS	139.3
3	Sam	Jan	BS	135.2
4	Sam	Feb	BS	160.1
5	Senthil	Jan	BP	90.0
6	Senthil	Feb	BP	78.0
7	Sam	Jan	BP	80.0
8	Sam	Feb	BP	81.0

Step 2: cast

- Applying the dcast() function
- dcast (data, formula, value.var = col. with values)

Code

```
# cast operation sample code
# continued from previous code
# we use dcast as we are working on
a dataframe
Df2 = dcast(Df,
variable+month ~ Name
value.var="value"
print(Df2)
```

Console Output

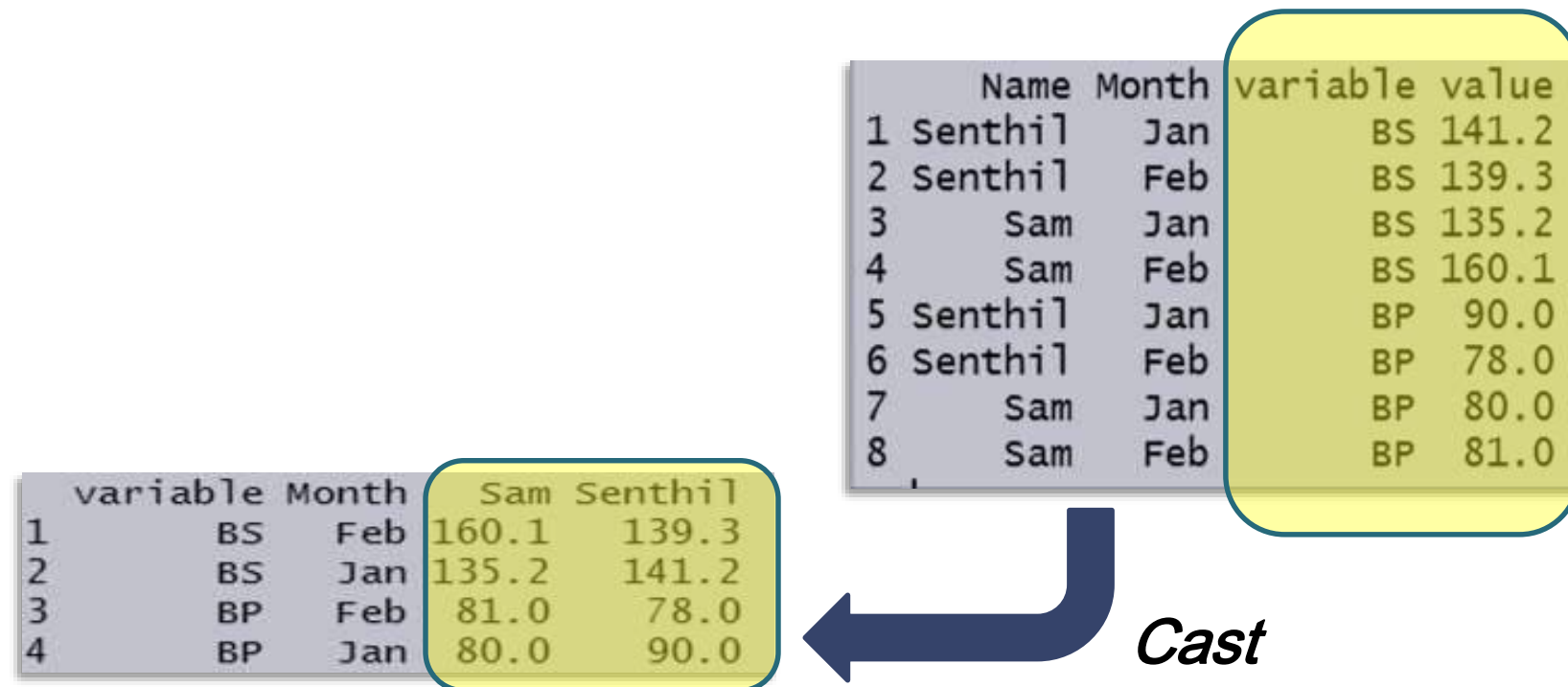
```
variable Month   Sam Senthil
1      BS    Feb 160.1   139.3
2      BS    Jan 135.2   141.2
3      BP    Feb  81.0    78.0
4      BP    Jan  80.0    90.0
>
```

Column of Df from which the values are to be taken from

Columns “variable” & “month” to remain as is.
Categories in column “Name” become new variables.

Step 2: cast

`Df2 = dcast(Df, variable+month ~ Name, value.var="value")`



Recasting in single step

- Applying the recast() function performs melt and cast in one command
- `recast(data, formula, ..., id.var, measure.var)`

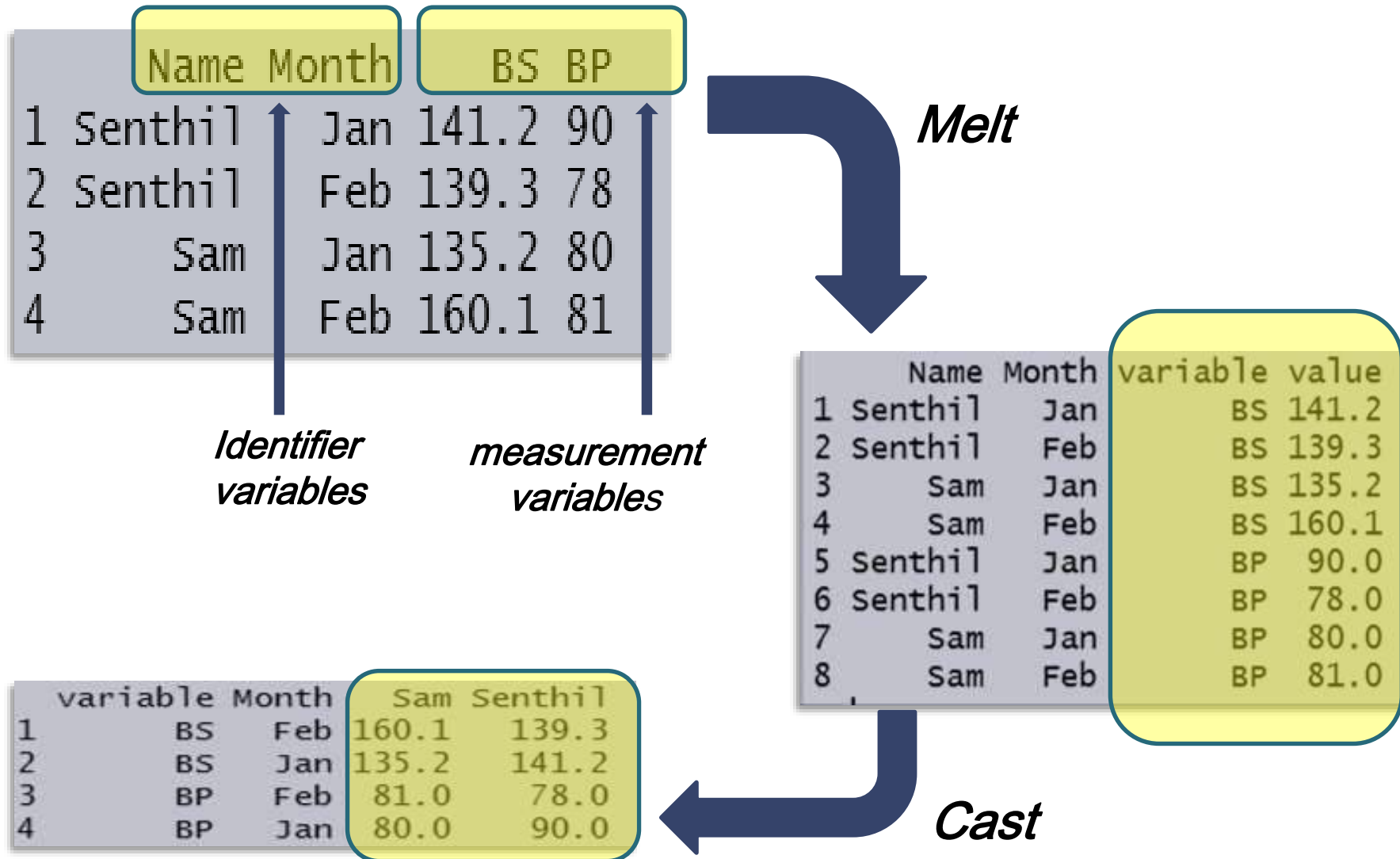
Command & console Output

Parameter refers to the “cast”
section of the command

Parameter refers to the “melt”
section of the command

```
Console ~/ 
> recast(pd, variable+Month~Name, id.var=c("Name", "Month"))
  variable Month  Sam Senthil
1      BS   Feb 160.1   139.3
2      BS   Jan 135.2   141.2
3      BP   Feb  81.0    78.0
4      BP   Jan  80.0    90.0
> |
```

recast()-melt and cast together




Add new variable to dataframe based on existing ones

- Call the library 'dplyr' command using the library() command
- mutate() command will add extra variable columns based on existing ones.

Code

```
# Adding new variables
#Continue from
#example on slide 3
library(dplyr)
pd2 <- mutate(pd, log_BP = log(BP))
print(pd2)
```



Console Output

```
> # Adding new variables
> #Continue from
> #example on slide 3
> library(dplyr)
> pd2 <- mutate(pd, log_BP = log(BP))
> print(pd2)
```

	Name	Month	BS	BP	log_BP
1	Senthil	Jan	141.2	90	4.499810
2	Senthil	Feb	139.3	78	4.356709
3	Sam	Jan	135.2	80	4.382027
4	Sam	Feb	160.1	81	4.394449

- original data frame 'pd' is the first argument
- multiple variables can be created as transformation of old variable
- here, new variable column is "log_BP" which is log of variable column "BP"

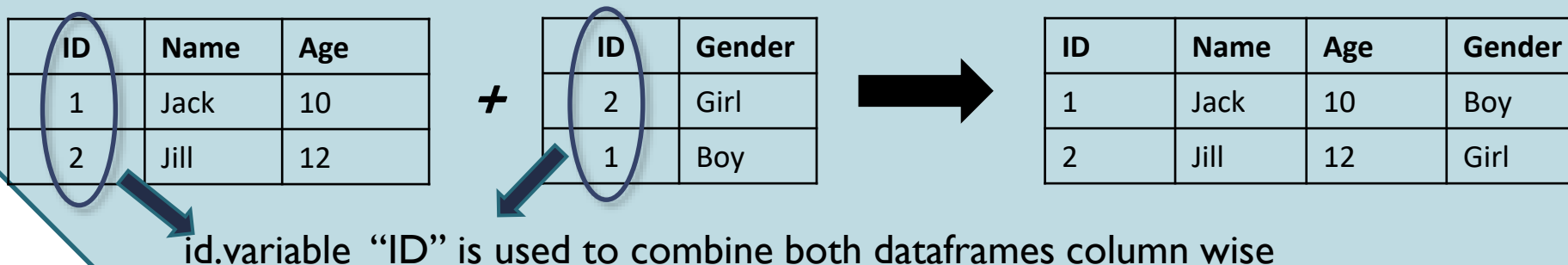
Joining of two frames

Combining two dataframes – dplyr package

The common syntax for “dplyr” functions used to combine dataframes:

“function(dataframe1, dataframe2, by = id.variable)”

- *The “id.variable” is common to both dataframes*
- *This variable provides the identifiers for combining the 2 dataframes*
- *The nature of combination depends on the function to be used*
- *Illustration Example : A possible combination*



Combining two dataframes

- Call the library 'dplyr' command using the library() command
- The following commands would be used to combine datasets:

❖ left_join()

❖ full_join()

❖ right_join()

❖ semi_join()

❖ inner_join()

❖ anti_join()

Example: create first dataframe

Create the data frame 'pd'

	Name	Month	BS	BP
1	Senthil	Jan	141.2	90
2	Senthil	Feb	139.3	78
3	Sam	Jan	135.2	80
4	Sam	Feb	160.1	81

Code

```
# Data frame example 2
pd=data.frame("Name"=c("Senthil",
Senthil,"Sam","Sam"),
"Month"=c("Jan","Feb","Jan","Feb"),
"BS" = c(141.2,139.3,135.2,160.1),
"BP" = c(90,78,80,81))

print(pd)
```

Console Output

```
> pd=data.frame("Name"=c("Senthil",
+ "Senthil","Sam","Sam"),
+ "Month"=c("Jan",
+ "Feb","Jan","Feb"),
+ "BS" = c(141.2,1
+ 39.3,135.2,160.1),
+ "BP" = c(90,78,8
+ 0,81))
> print(pd)
      Name Month    BS BP
1 Senthil   Jan 141.2  90
2 Senthil   Feb 139.3  78
3      Sam   Jan 135.2  80
4      Sam   Feb 160.1  81
> |
```


Create another dataframe

Create another data frame : 'pd_new'

	Name	Department
1	Senthil	PSE
2	Ramesh	Data Analytics
3	Sam	PSE

Code

```
# Data frame example 3
pd_new=data.frame("Name"=c("Senthil",
"Ramesh", "Sam"),
"Department"=c("PSE","Data
Analytics","PSE"))
print(pd_new)
```

Console Output

```
> # Data frame example 3
> pd_new=data.frame("Name"=c("Se
nthil","Ramesh","Sam"),
+                      "Department"
=c("PSE","Data Analytics","PSE")
)
> print(pd_new)
      Name      Department
1 Senthil          PSE
2  Ramesh Data Analytics
3     Sam          PSE
> |
```

left_join()

- joins matching rows of "dataframe2 " to "dataframe1" based on the "id.variable"
- In the example, only "Sam" and "Senthil" from id.variable "Name" are present in "pd" which is dataframe1.
- Only these two IDs & corresponding values in "pd_new" will be merged with "pd"
- The variable "Department" from "pd_new" would be merged to its *'left'* to *pd*

dataframe1 : pd

	Name	Month	BS	BP
1	Senthil	Jan	141.2	90
2	Senthil	Feb	139.3	78
3	Sam	Jan	135.2	80
4	Sam	Feb	160.1	81

dataframe2 : pd_new

	Name	Department
1	Senthil	PSE
2	Ramesh	Data Analytics
3	Sam	PSE

left_join()

USE DATAFRAMES 'pd' and pd_new

Code

```
#using left_join()
#to combine two dataframes
#Continue from
#example
library(dplyr)
pd_left_join1 <- left_join(pd, pd_new, by
="Name")
print(pd_left_join1)
```

dataframe1 : pd

	Name	Month	BS	BP
1	Senthil	Jan	141.2	90
2	Senthil	Feb	139.3	78
3	Sam	Jan	135.2	80
4	Sam	Feb	160.1	81

dataframe2 : pd_new

	Name	Department
1	Senthil	PSE
2	Ramesh	Data Analytics
3	Sam	PSE

pd_left_join1

	Name	Month	BS	BP	Department
1	Senthil	Jan	141.2	90	PSE
2	Senthil	Feb	139.3	78	PSE
3	Sam	Jan	135.2	80	PSE
4	Sam	Feb	160.1	81	PSE

right_join()

Joins matching rows of “dataframe1 ” to “dataframe2” based on the “id.variable”

Code

```
#using right_join() #using
right_join()
#to combine two data frames
#Continue from
#example
pd_right_join1 <- right_join
(pd, pd_new, by = "Name")
print(pd_right_join1)
```

dataframe1 : pd

	Name	Month	BS	BP
1	Senthil	Jan	141.2	90
2	Senthil	Feb	139.3	78
3	Sam	Jan	135.2	80
4	Sam	Feb	160.1	81

dataframe2 : pd_new

	Name	Department
1	Senthil	PSE
2	Ramesh	Data Analytics
3	Sam	PSE

pd_right_join1

	Name	Month	BS	BP	Department
1	Senthil	Jan	141.2	90	PSE
2	Senthil	Feb	139.3	78	PSE
3	Ramesh	<NA>	NA	NA	Data Analytics
4	Sam	Jan	135.2	80	PSE
5	Sam	Feb	160.1	81	PSE

right_join()

Joins matching rows of “dataframe1 ” to “dataframe2” based on the “id.variable”

Code

```
#using right_join() #using
right_join()
#to combine two data frames
#Continue from
#example
pd_right_join2 <- right_join
(pd_new, pd,
by = "Name")
print(pd_right_join2)
```

dataframe1 : pd_new

	Name	Month	BS	BP
1	Senthil	Jan	141.2	90
2	Senthil	Feb	139.3	78
3	Sam	Jan	135.2	80
4	Sam	Feb	160.1	81

dataframe2 : pd

	Name	Department
1	Senthil	PSE
2	Ramesh	Data Analytics
3	Sam	PSE

pd_right_join2

	Name	Department	Month	BS	BP
1	Senthil	PSE	Jan	141.2	90
2	Senthil	PSE	Feb	139.3	78
3	Sam	PSE	Jan	135.2	80
4	Sam	PSE	Feb	160.1	81

inner_join()

Merges and retains those rows with IDs present in both dataframes

Code

```
#using inner_join()
#to combine two data frames
#Continue from
#example
library(dplyr)
pd_inner_join1 <- inner_join
(pd_new, pd, by = "Name")
print(pd_inner_join1)
```

dataframe1 : pd_new

	Name	Department
1	Senthil	PSE
2	Ramesh	Data Analytics
3	Sam	PSE

dataframe2 : pd

	Name	Month	BS	BP
1	Senthil	Jan	141.2	90
2	Senthil	Feb	139.3	78
3	Sam	Jan	135.2	80
4	Sam	Feb	160.1	81

pd_inner_join1

	Name	Department	Month	BS	BP
1	Senthil	PSE	Jan	141.2	90
2	Senthil	PSE	Feb	139.3	78
3	Sam	PSE	Jan	135.2	80
4	Sam	PSE	Feb	160.1	81

Combining two dataframes: summary

SELF STUDY



`left_join()`



`right_join()`



`inner_join()`



❖ `full_join()`

❖ `semi_join()`

❖ `anti_join()`