











Tables

Tabulating: for
nominal data

					
	grey	brown	navy	white	pink
frequency	1	1	1	2	2

`table(colours)`

					
♀	1	0	1	1	1
♂	0	1	0	1	0
⚧	0	0	0	0	1

`table(genders,colours)`

```
tablename <- table(var1,var2)
```

Pipes

Take your data...
... Do one thing...
...Then do another...
... And then one more

```
gdata %>%  
  do_one_thing() %>%  
  then_do_another() %>%  
  then_one_more()
```

Data can be whatever
would go into the first
argument of the functions

```
c(1,5,3,2,5) %>%  
  mean() %>%  
  round() %>%  
  abs()
```

Can assign contents to a
new variable

```
absRoundedMean <- x %>%  
  sum() %>%  
  sqrt() %>%  
  round(digits=2)
```

Grouping

The name of
the data tibble
we want to use

`data %>%`

`group_by(var1, var2) %>%
summarise(mnV3 = mean(var3),
sdV3 = sd(var3),
nV3 = n(),
stdErrV3 = sdV3/sqrt(nV3)) %>%`

Always
ungroup!

`ungroup()`

We want to group the data
by `var1` and `var2` which are
variables in `data`


Calculations in
`summarise()`

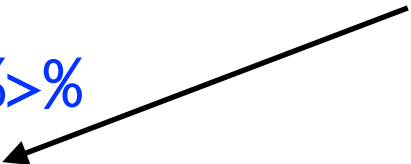
can be any functions that
operate on any of the
variables (columns) in the
data like `var3` (or the groups
as a whole, like `n()`). The
result of the calculations are
put in the variable names like
`mnV3`, etc. Later calculations
can refer to variables created
by `summarise` just upstream.


Later calculations can refer
to variables created by
`summarise` just upstream.

Filter and arrange

General idea: `filter()` lets you select a subset of the rows of the tibble, and `arrange()` tells you how to sort them

The name of the data tibble we want to use  `data %>%
filter(LOGICALVAL) %>%
arrange(var)`

We only want rows that satisfy LOGICALVAL 

Arranges the rows by `var` 

Simple example of `filter()` includes only rows of `data` for which the value of `var1` is `x`

```
data %>%  
  filter(var1=="x")
```

More complicated can include multiple comparisons

```
data %>%  
  filter(var1=="x" & var2>N)
```

Select and mutate

`select()` lets you select a subset of the columns of the tibble

The name of
the data tibble → `data %>%`
we want to use

`select(var1, var2, var3)`

We only want to
keep `var1`, `var2`,
and `var3`

`mutate()` lets you calculate new ones

`data %>%`
`mutate(var4 = var3/var2)`

We are creating a new
variable called `var4` which is
`var3` divided by `var2`)

Pivot wider and longer

`pivot_longer()`

Converts from wide to long format by decreasing the number of columns and increasing the number of rows

`data %>%`

`pivot_longer(cols, names_to="key", values_to="val")`

Says which columns
to combine into the
new one

Defines the name
of the new column

Defines what to call the new
column of values

long

id	key	val
1	x	a
2	x	b
1	y	c
2	y	d
1	z	e
2	z	f

`pivot_wider()`

Converts from long to wide format by increasing the number of columns and decreasing the number of rows

`data %>%`

`pivot_wider(names_from="key", values_from="val")`

Defines where the
names for the new
columns come from

Defines where the
values in the new
columns come from

wide

id	x	y	z
1	a	c	e
2	b	d	f