THE UNIVERSITY of EDINBURGH | usher institute

HDS Tutorial 2
Week 4

| Brittany Blankinship | 3 & 5 May 2022 |

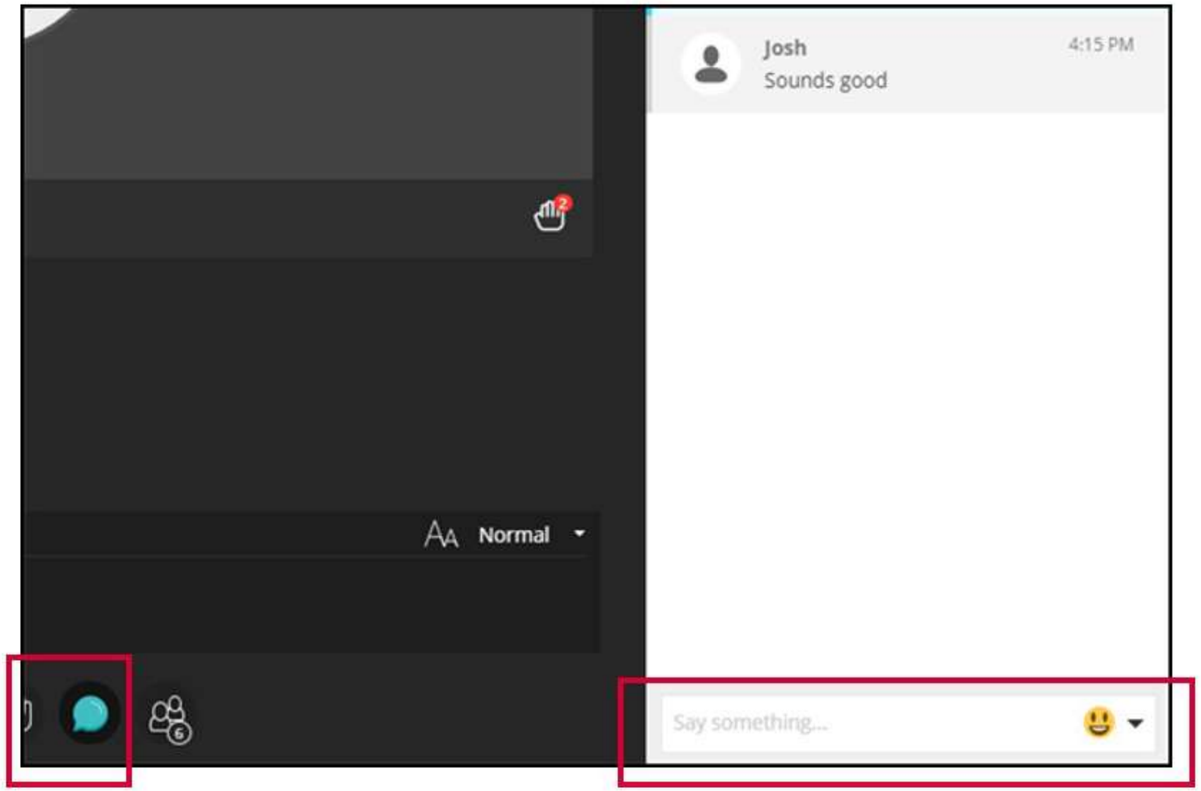
Audio check

Can you hear the presenter talking?

Please type **yes** or **no** in the “Text chat area”

If you can’t hear:

- Check your Audio/Visual settings in the Collaborate Panel
- Try signing out and signing back into the session
- Type into the chat box and a moderator will try to assist you



ddi.hsc.talent@ed.ac.uk

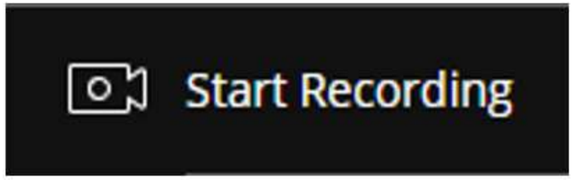
Supported by
THE UNIVERSITY of EDINBURGH
Data-Driven Innovation

THE UNIVERSITY of EDINBURGH | usher institute

Recording

Open to the world

This session will now be recorded. Any further information that you provide during a session is optional and in doing so you give us consent to process this information.



These sessions will be stored by the University of Edinburgh for one year and published for 30 days after the event. Schools or Services may use the recordings for up to a year on relevant websites.

By taking part in a session, you give us your consent to process any information you provide during it.

ddi.hsc.talent@ed.ac.uk

Supported by THE UNIVERSITY of EDINBURGH | DDI Data-Driven Innovation

THE UNIVERSITY of EDINBURGH | usher institute

HDS Tutorial 2
Week 4

| Brittany Blankinship | 3 & 5 May 2022 |

Agenda

- **Plots vs Tables**
- `ggplot2` **for making plots in R**
- `gt` **for making tables in R**
- **Q&A**

Are you using R on your own device, Noteable, or both?

Do you think figures or tables are more useful for data visualisation?

Why do graphs and tables matter?

Both graphs AND tables are tools for communication

Informative **and well-presented graphs & tables ARE better communication**

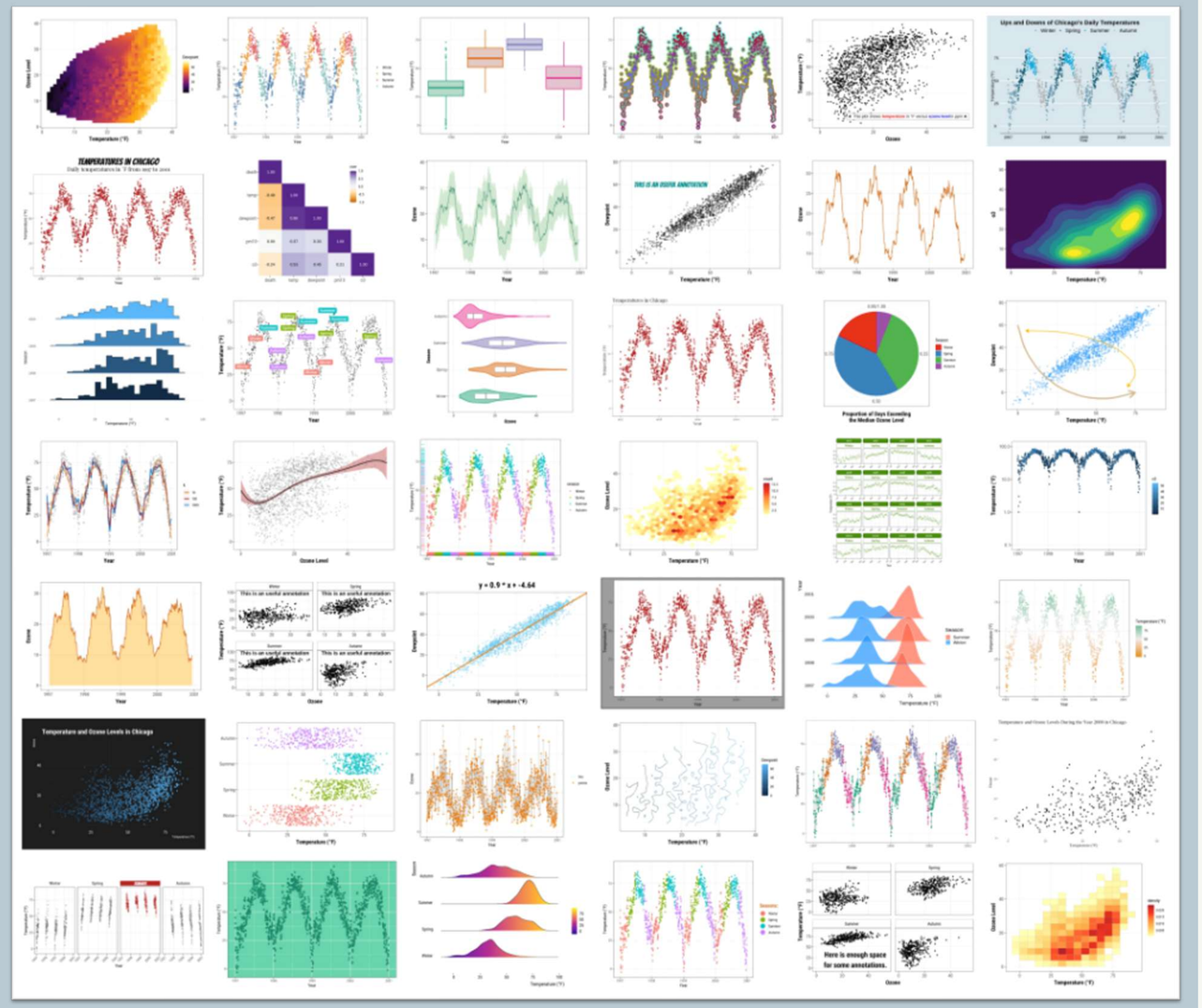
When to use tables vs graphs?

Use Tables When	Use Graphs When
• The display will be used to look up individual values	• The display will be used to reveal relationships among whole sets of values
• It will be used to compare individual values	• The message is contained in the shape of the values (e.g., patterns, trends, exceptions)
• Precise values are required	
• Quantitative values include more than one unit of measure	
• Both detail and summary values are included	

Adapted from:
Few, Stephen. (2012). *Show Me the Numbers: Designing Tables and Graphs to Enlighten*.(4)57

ggplot2

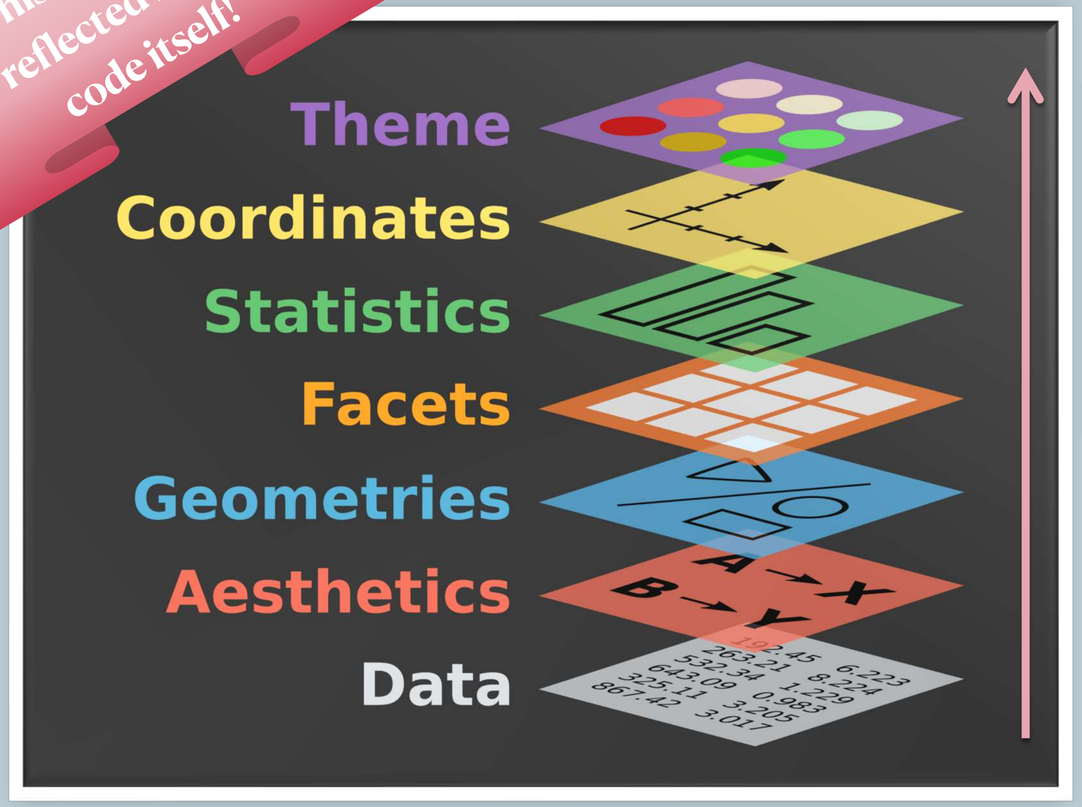
- **Application of Leland Wilkinson's [grammar of graphics](#) for R**
- **A system for iteratively and declaratively creating graphics**
- **Just like grammar in a language constructs sentences, grammar of graphics constructs data visualisations**



Credit: Cédric Sherer
ggplot2 in-depth tutorial

Building blocks of a `ggplot`

This layering is reflected in the code itself!



Element	Description	Code
Data	The dataset being plotted	`ggplot(data = ,`
Aesthetics	The scales onto which we map our data	`aes(x, y, fill, color))`
Geometries	The visual elements used for our data	`+ geom_()`
Facets	Plotting small multiples	`+ facet_()`
Statistics	Representations of our data to aid understanding	`+ stat_()`
Coordinates	The space on which the data will be plotted	`+ coord_()`
Themes	All non-data ink, design elements	`+ theme_()`

The key ingredients

```
ggplot(data = cancelled ,
```

data layer | use the data cancelled | THEN

The key ingredients

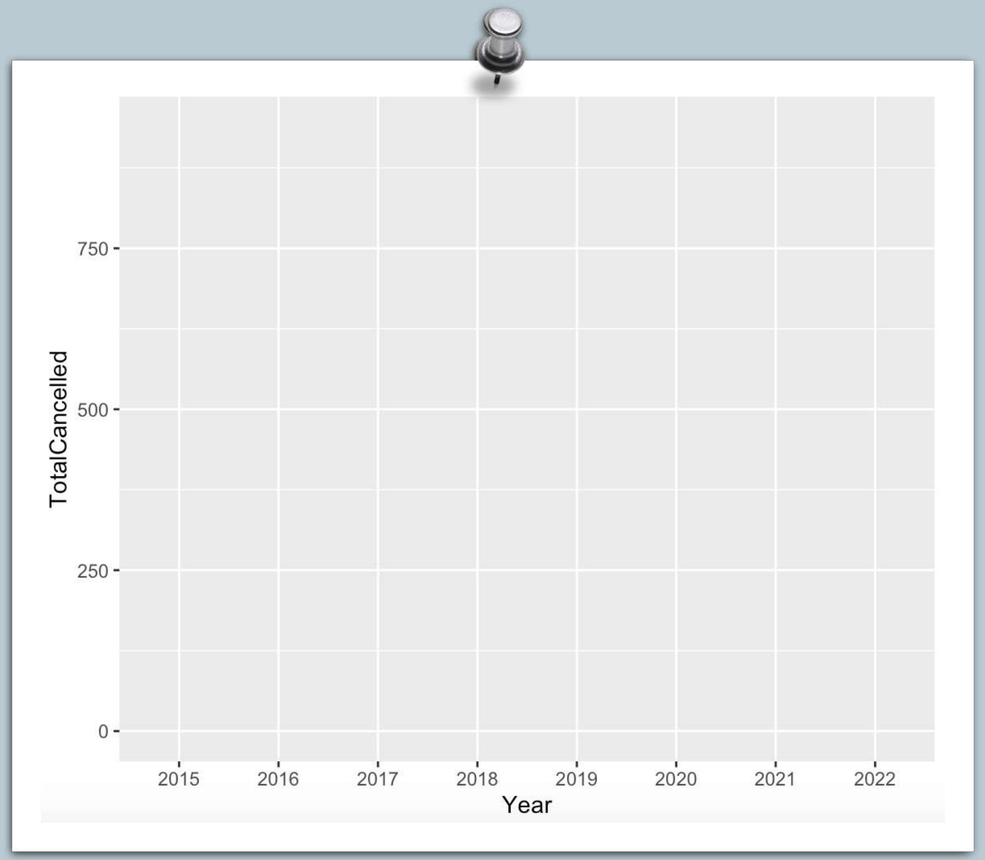
```
ggplot(data = cancelled ,
```

data layer — use the data cancelled — THEN

```
aes(x = Year, y = TotalCancelled))   +
```

use the X axis to show “Year” then the Y axis to show ”Total Cancelled” AND

aesthetics layer



The key ingredients

```
ggplot(data = cancelled ,
```

data layer — use the data cancelled — THEN

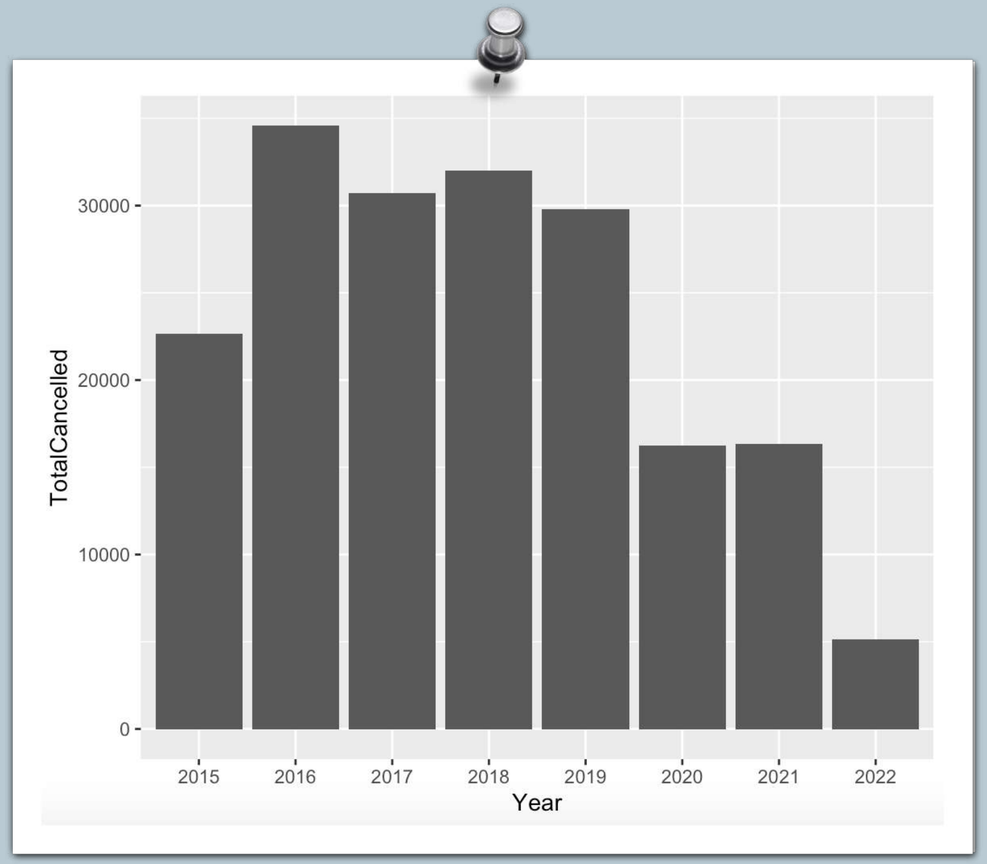
```
aes(x = Year, y = TotalCancelled))   +
```

use the X axis to show “Year” then the Y axis to show ”Total Canceled” AND

aesthetics layer

```
geom_bar()
```

geometry layer .... take all that and make a bar chart



The key ingredients

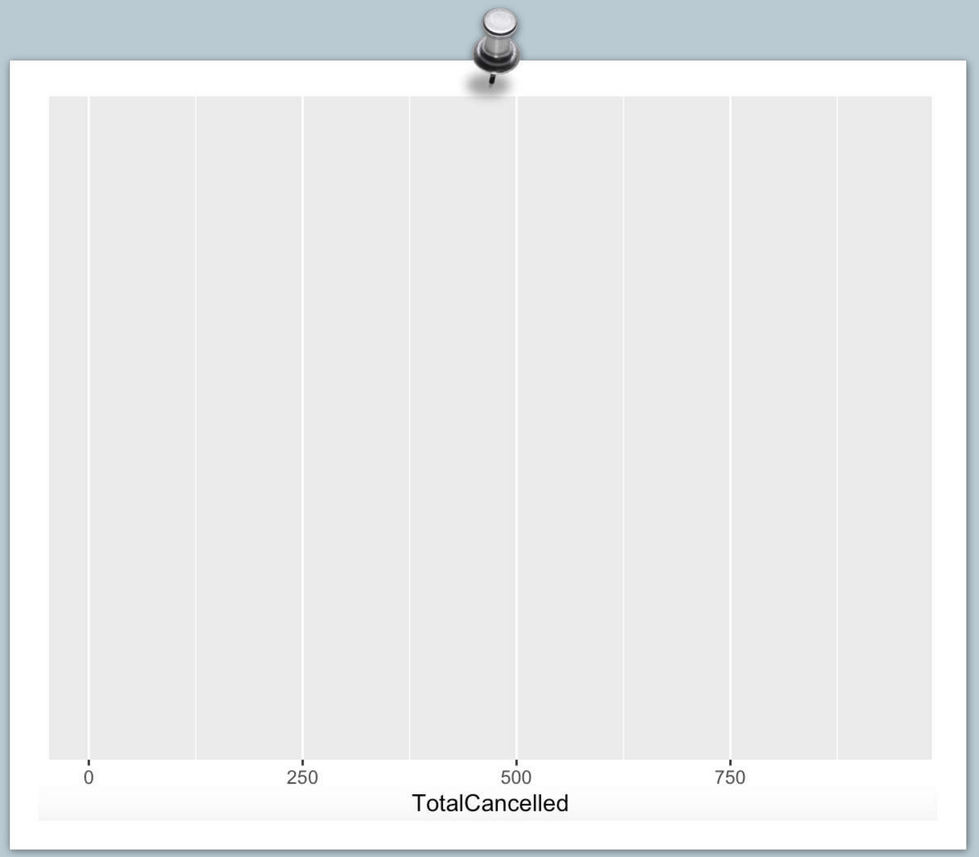
```
ggplot(data = cancelled ,
```

data layer — use the data cancelled — THEN

```
aes(x = TotalCancelled))   +
```

use the X axis to show "Total Cancelled" — AND

aesthetics layer



The key ingredients

```
ggplot(data = cancelled ,
```

data layer — use the data cancelled — THEN

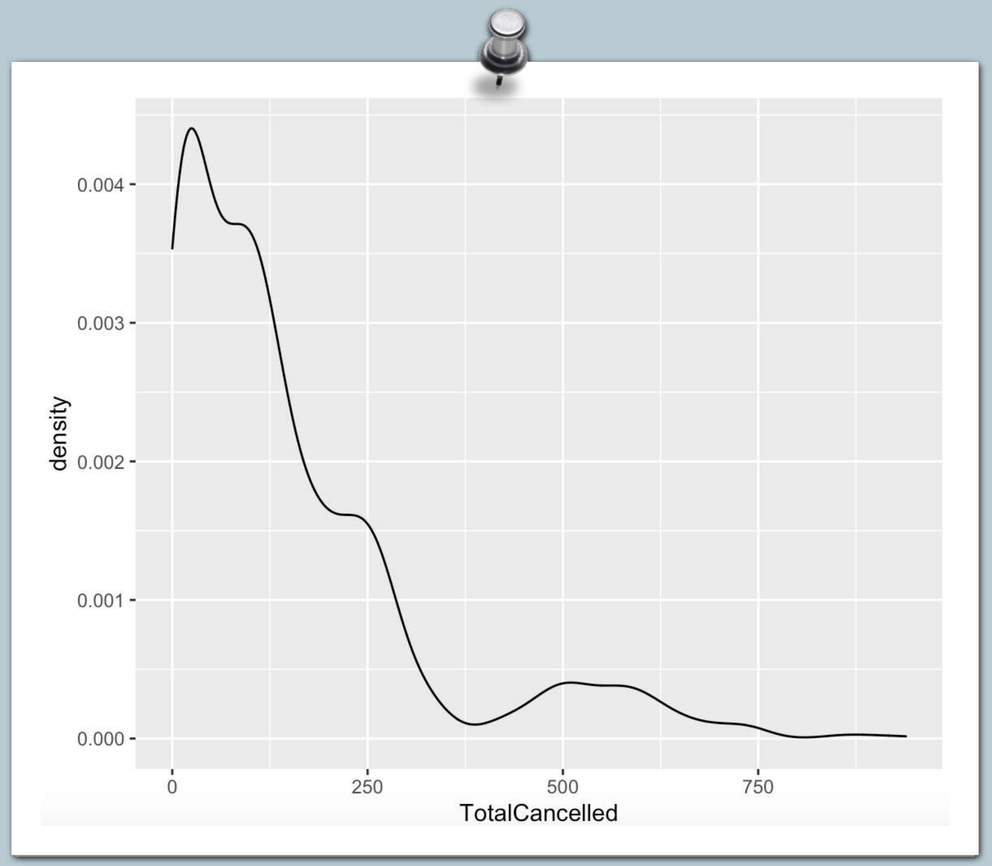
```
aes(x = TotalCancelled))   +
```

use the X axis to show "Total Cancelled" — AND

aesthetics layer

```
geom_density()
```

geometry layer .... take all that and make a density chart



The key ingredients

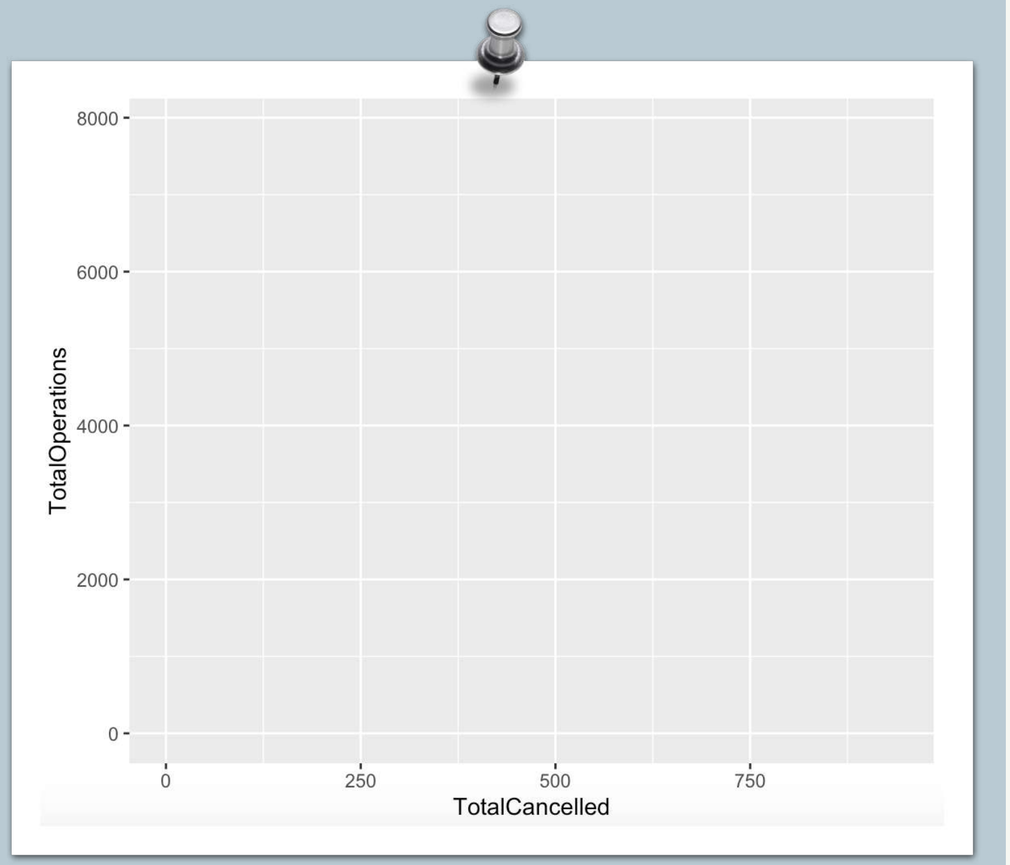
```
ggplot(data = cancelled ,
```

data layer use the data cancelled THEN

```
aes(x = TotalCancelled, y = TotalOperations))  +
```

use the X axis to show “Total Cancelled” then the Y axis to show ”Total Operations” AND

aesthetics layer



The key ingredients

```
ggplot(data = cancelled ,
```

data layer — use the data cancelled — THEN

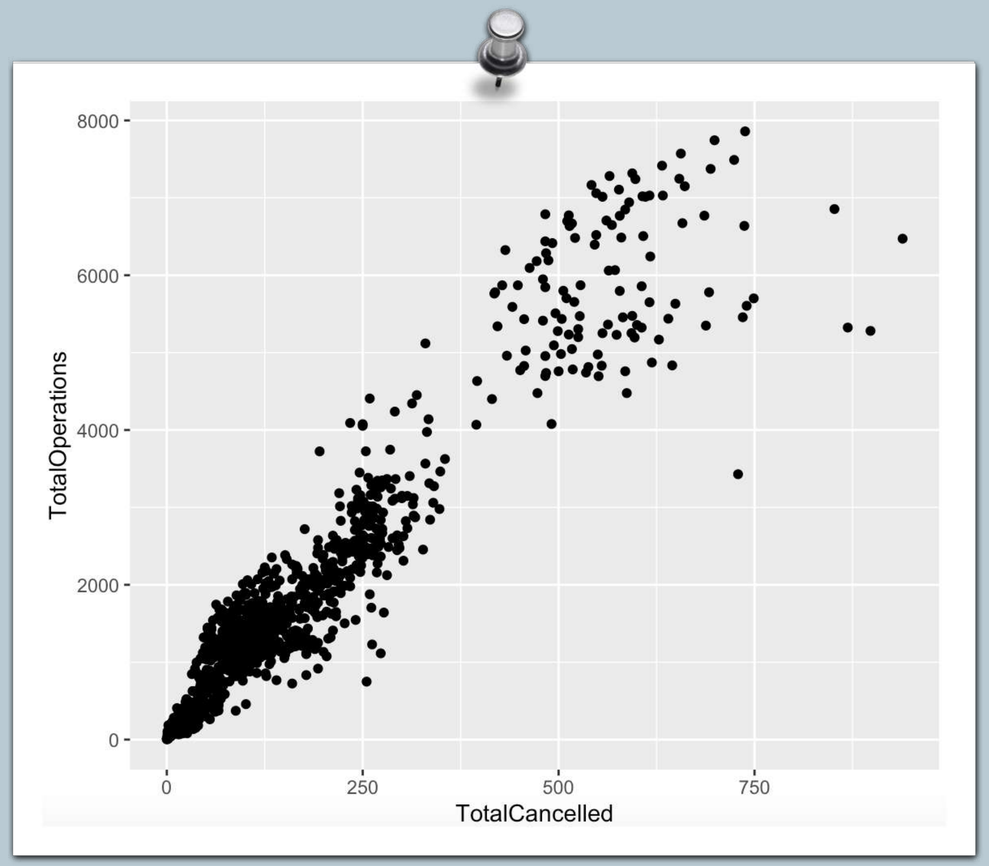
```
aes(x = TotalCancelled, y = TotalOperations))  +
```

use the X axis to show “Total Cancelled” then the Y axis to show ”Total Operations” AND

aesthetics layer

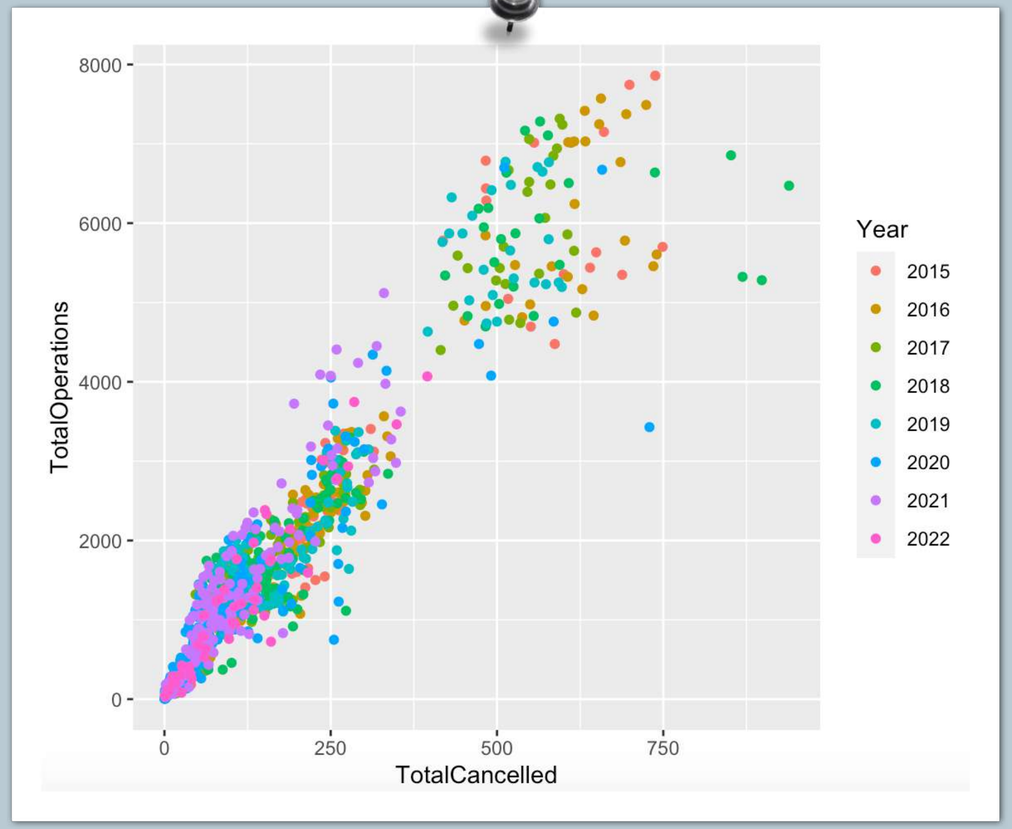
```
geom_point()
```

geometry layer .... take all that and make a scatter plot

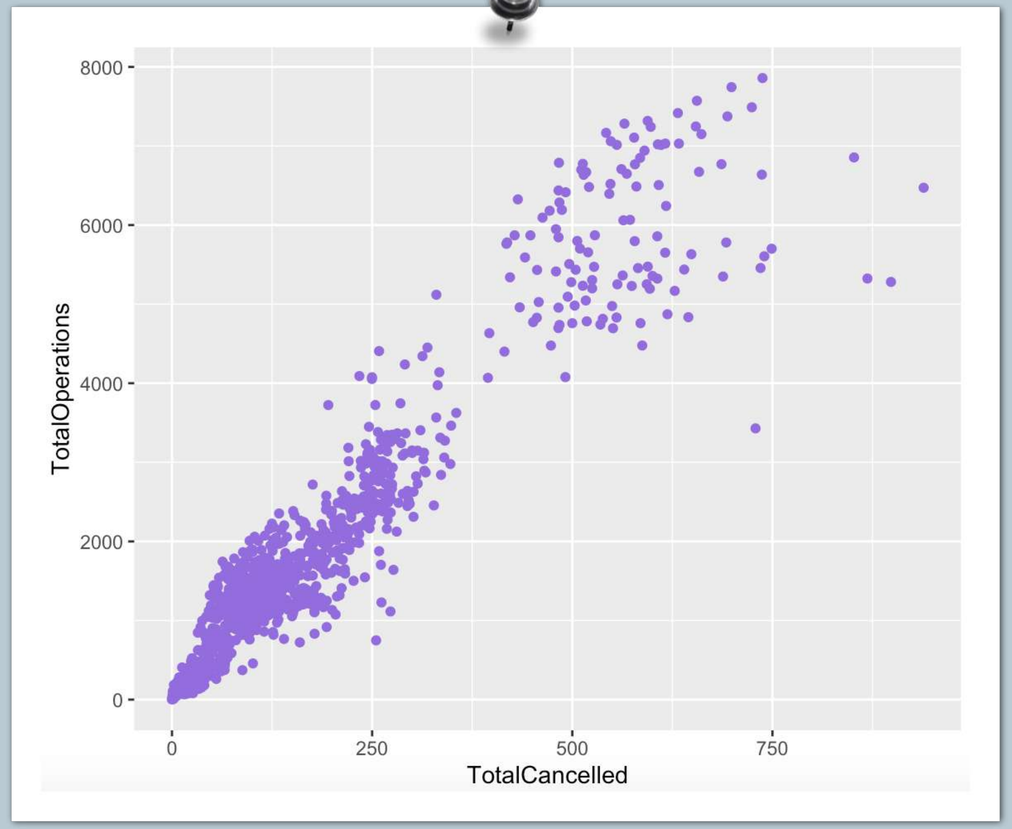


Aesthetics: Inside, Outside, or both

- **Within `aes()`:** the argument changes based on the values of the variable

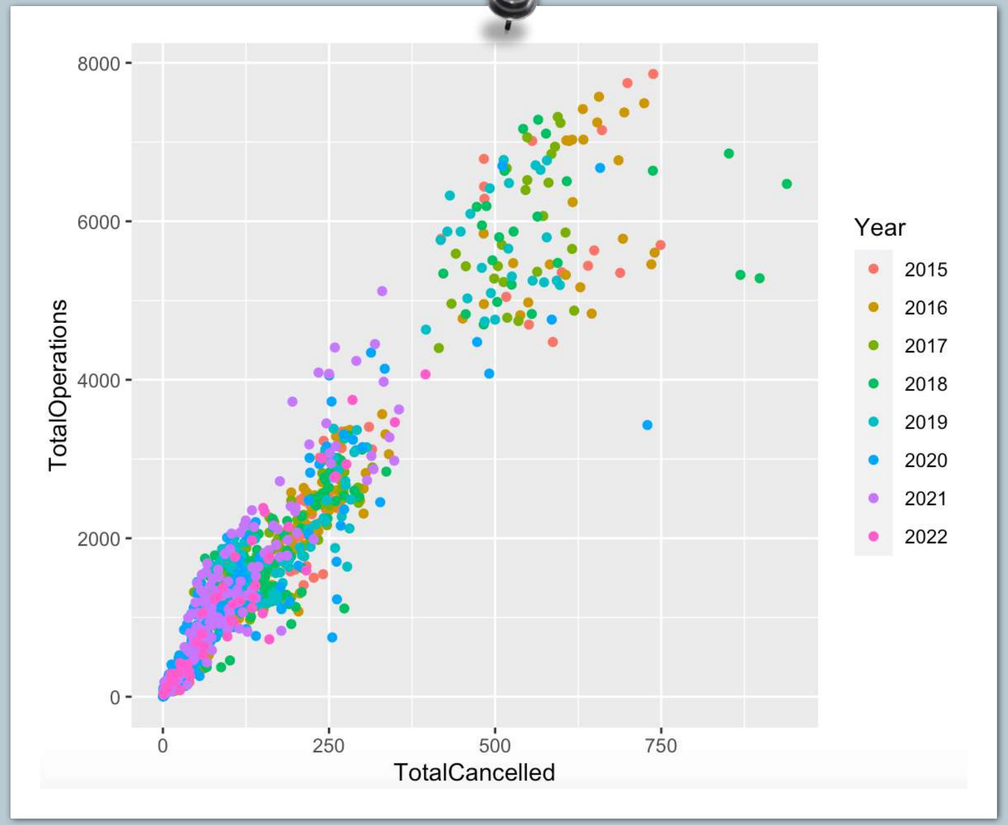


- **Outside `aes()`:** the argument is given a single value and *doesn't* change based on the values of the variable

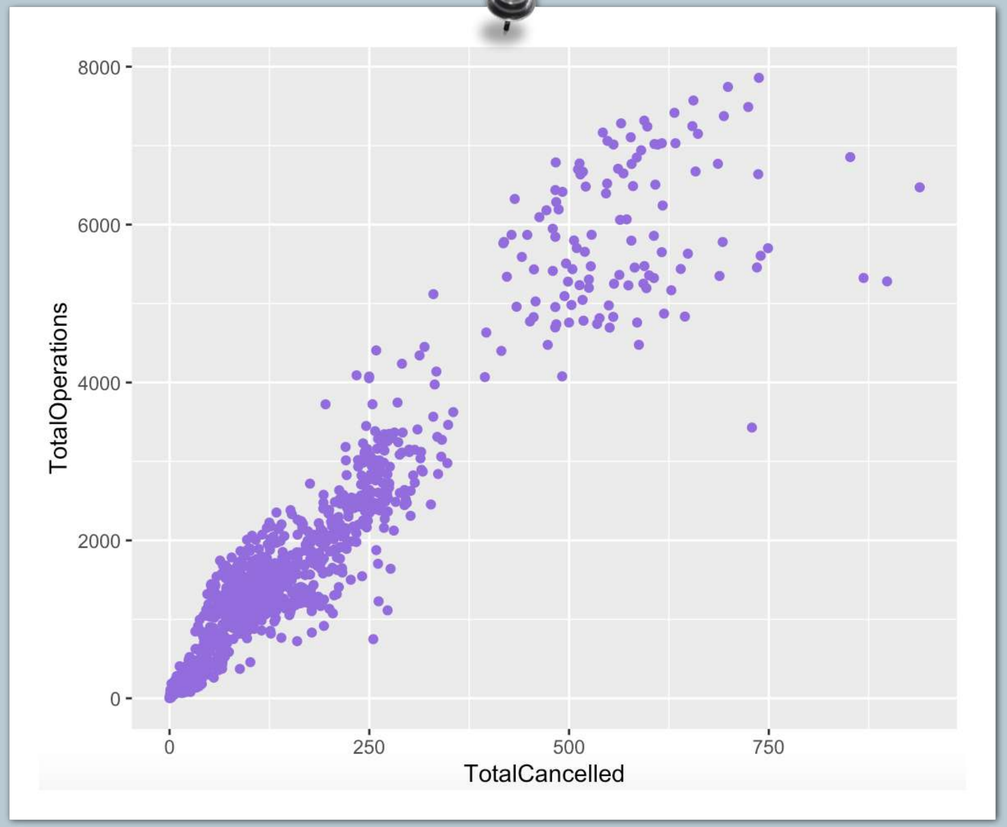


Aesthetics: Inside vs Outside

```
ggplot(cancelled, aes(x = TotalCancelled, y = TotalOperations,
                      color = Year)) +
  geom_point()
```

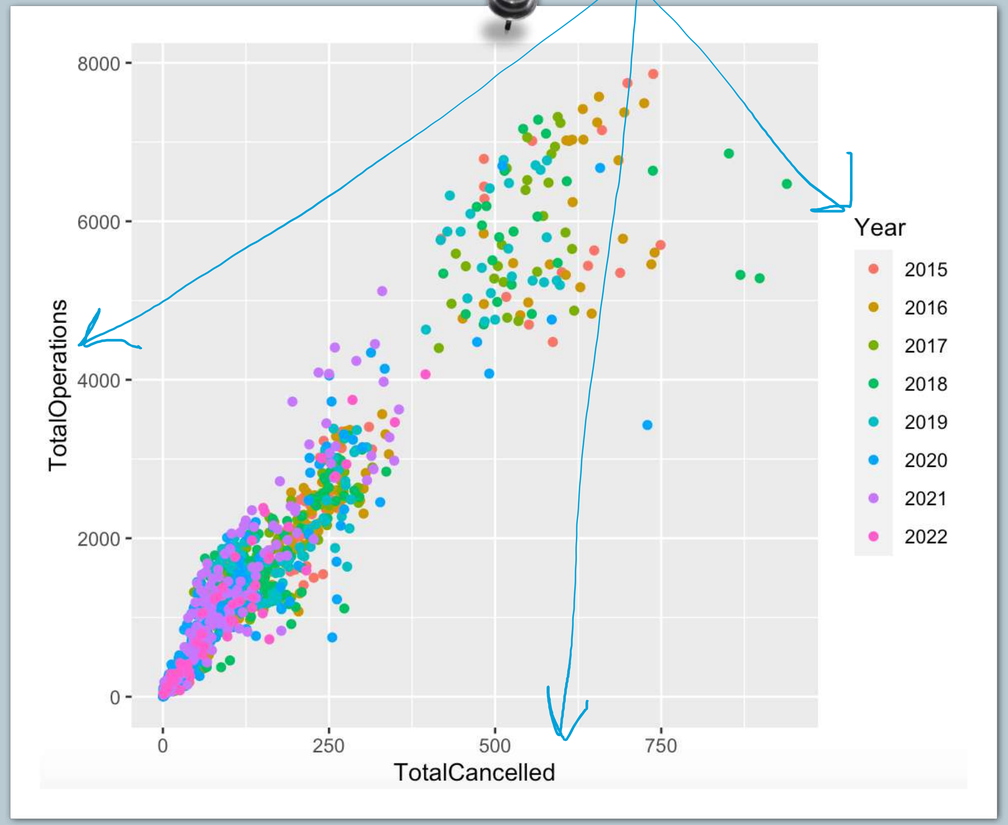



```
ggplot(cancelled, aes(x = TotalCancelled,
                      y = TotalOperations) +
  geom_point(color = "mediumpurple")
```

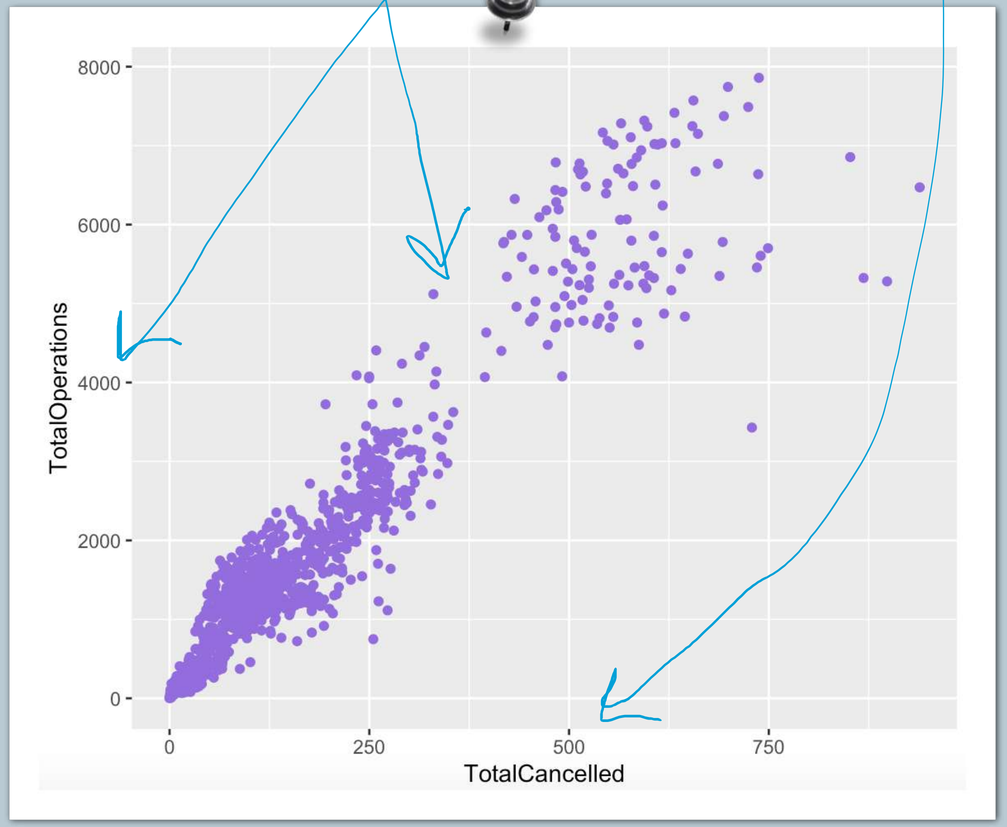


Aesthetics: Inside vs Outside

```
ggplot(cancelled, aes(x = TotalCancelled, y = TotalOperations,
                      color = Year)) +
  geom_point()
```



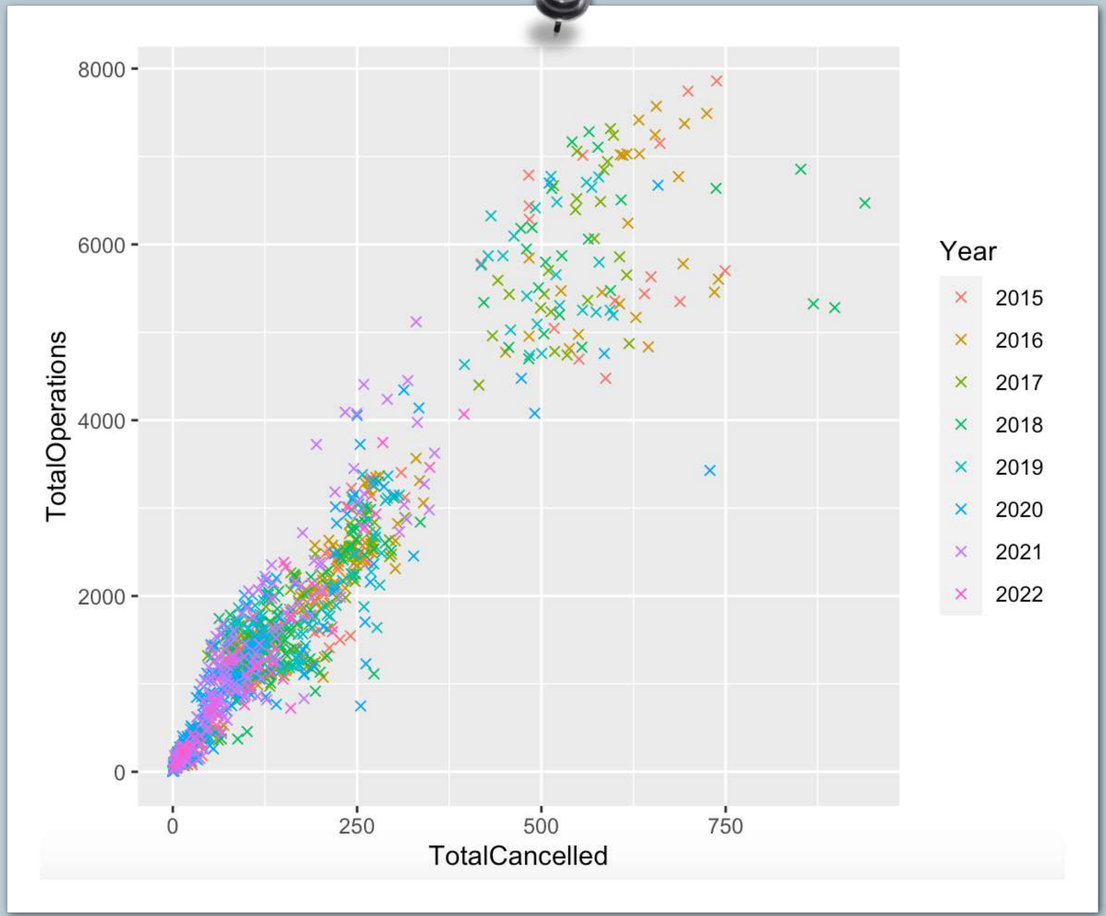

```
ggplot(cancelled, aes(x = TotalCancelled,
                      y = TotalOperations) +
  geom_point(color = "mediumpurple")
```



Aesthetics: Inside, Outside, or both

- [To learn more about the different aesthetic specifications]
- [For a list of colors in R]

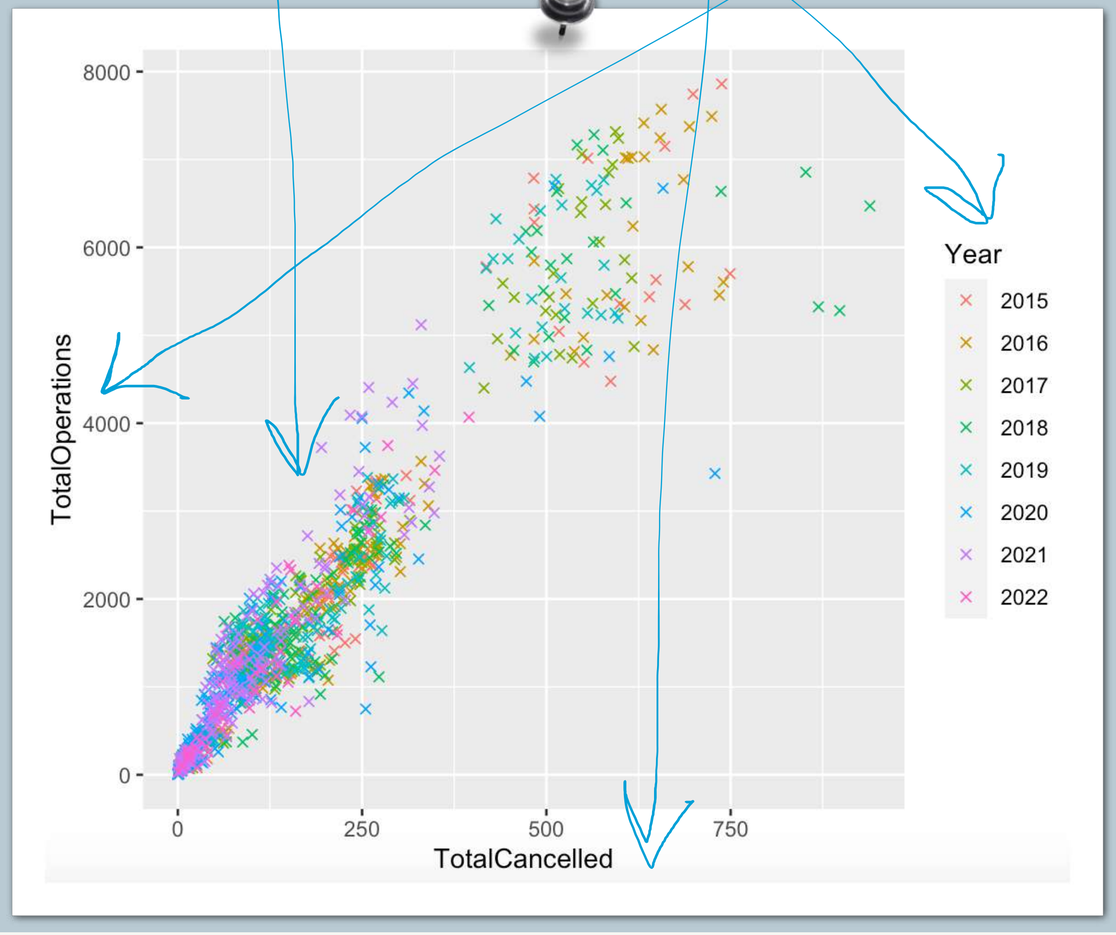
```
ggplot(cancelled, aes(x = TotalCancelled, y = TotalOperations,
                      color = Year)) +
  geom_point(shape = 4)
```



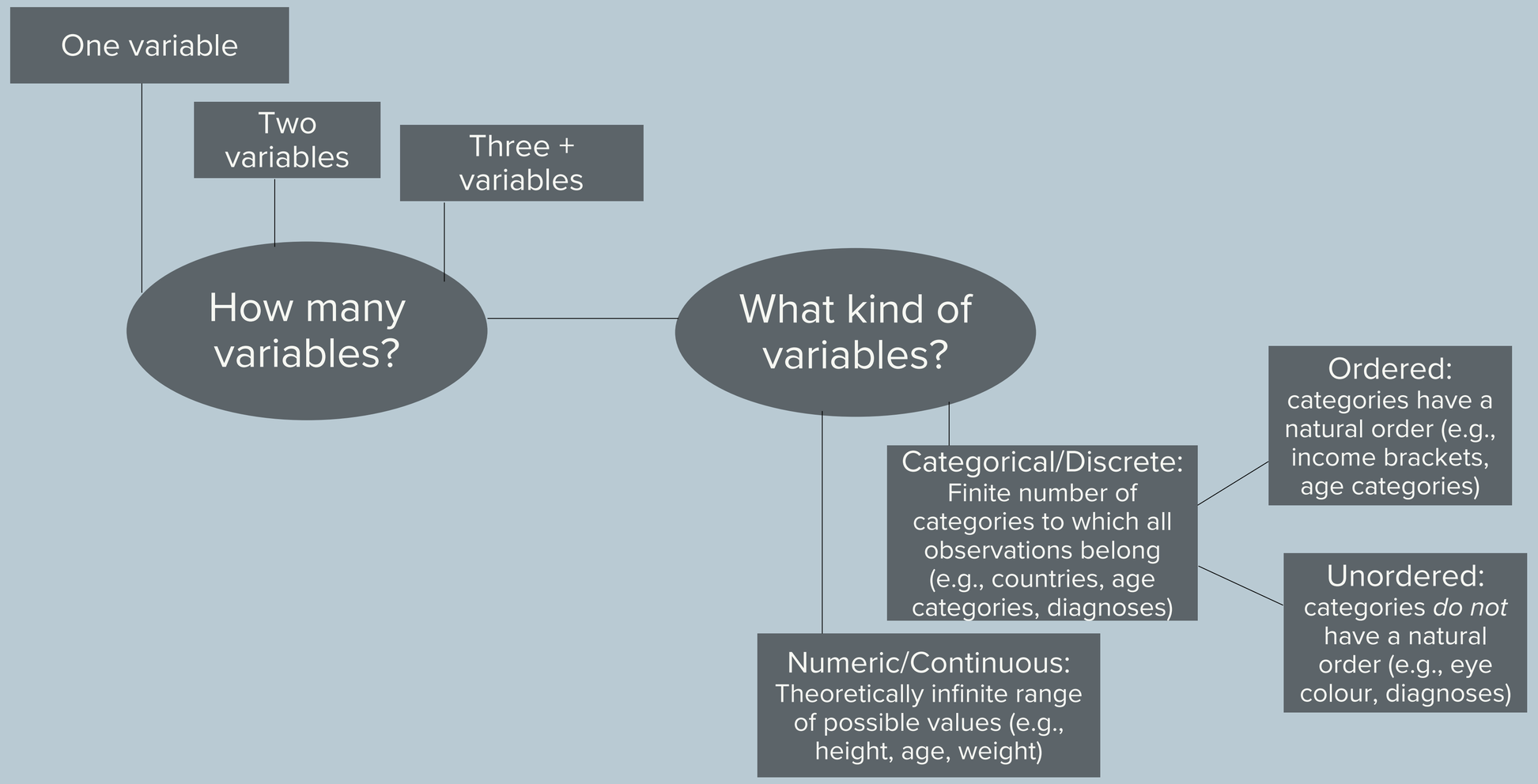
Aesthetics: Inside, Outside, or both

- [To learn more about the different aesthetic specifications]
- [For a list of colors in R]

```
ggplot(cancelled, aes(x = TotalCancelled, y = TotalOperations,
                      color = Year)) +
  geom_point(shape = 4)
```



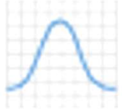
Geoms: decisions, decisions



ONE VARIABLE continuous

c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)

c + geom_area(stat = "bin"**)**
x, y, alpha, color, fill, linetype, size



c + geom_density(kernel = "gaussian"**)**
x, y, alpha, color, fill, group, linetype, size, weight

c + geom_dotplot()
x, y, alpha, color, fill

c + geom_freqpoly()
x, y, alpha, color, group, linetype, size

c + geom_histogram(binwidth = 5**)**
x, y, alpha, color, fill, linetype, size, weight

c2 + geom_qq(aes(sample = hwy)**)**
x, y, alpha, color, fill, linetype, size, weight

discrete

d <- ggplot(mpg, aes(fl))

d + geom_bar()
x, alpha, color, fill, linetype, size, weight

One variable geoms

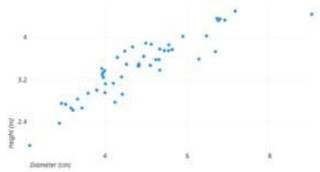
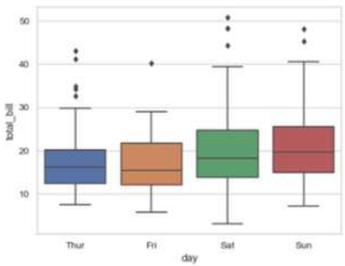
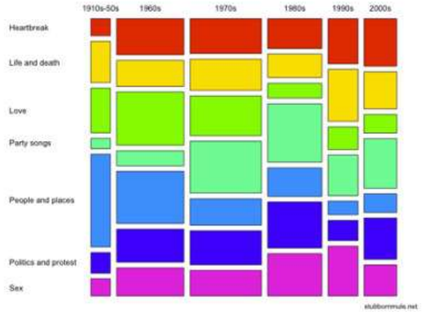
Continuous/numeric X

- **Histogram**
- **Density plot**
- **Dot plot**
- **Etc.**

Discrete/Categorical X

- **Bar charts**

When visualising 2 variables

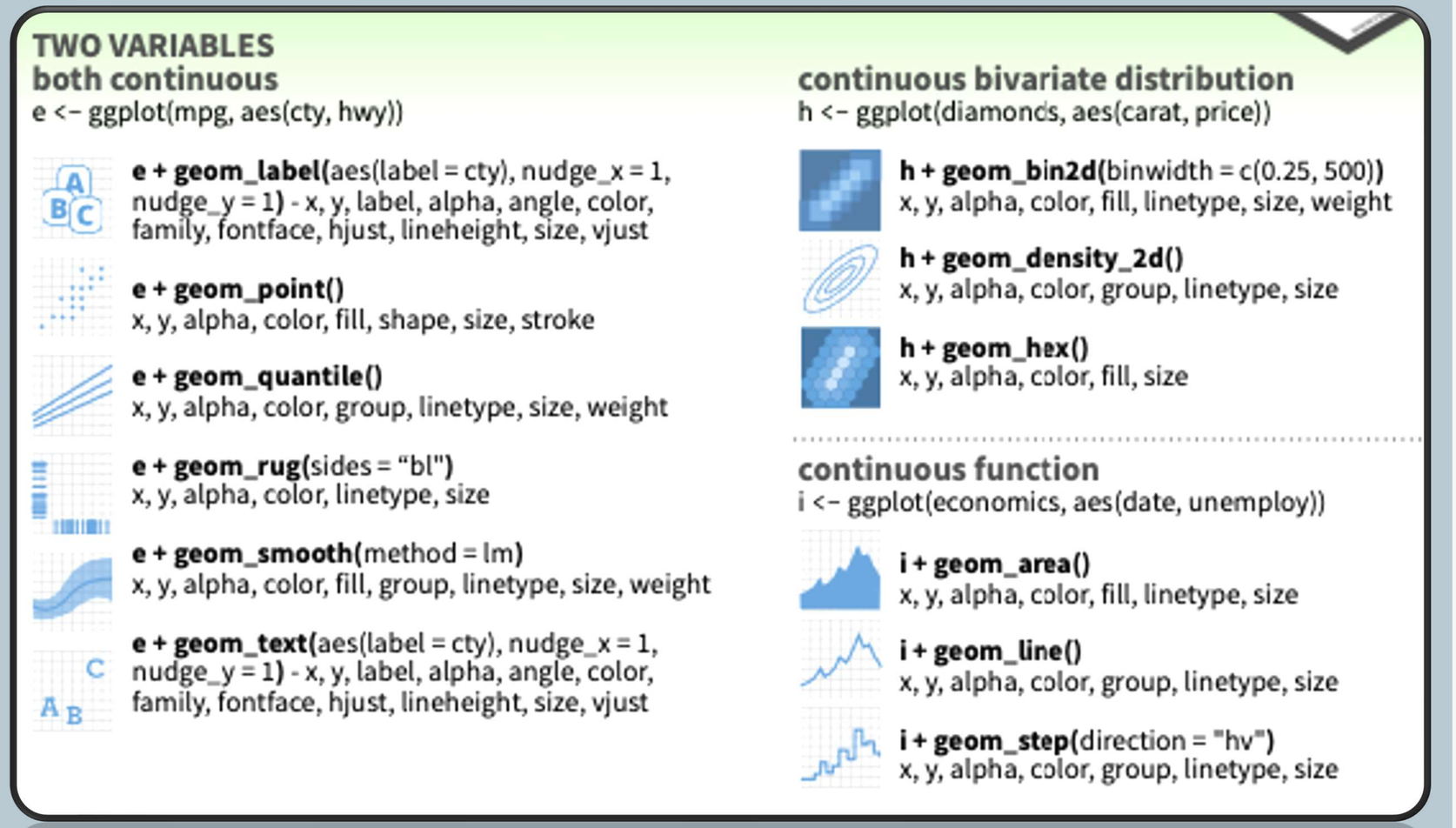
	Numerical	Categorical
Numerical	 • Scatter plot (point) • 2D binning (bin2d, hex) • Contour plot (density2d) • Quantiles (quantile, qq) • Lines (line, smooth) • Ribbons (ribbon, area)	 • Boxplot (boxplot, violin) • Counts (count, tile) • Error bars (errorbar) • Columns (col)
Categorical	***Which ggplot2 data viz is right for your data?*** (geoms in parentheses)	 • Mosaic (ggmosaic::geom_mosaic) • Counts (count, tile)

From Dr Sam Tyner's guest lecture video in Topic 4

2 variable geoms: numeric

Continuous/numeric X & Y

- **Line chart**
- **Scatter plot**
- **Etc.**



TWO VARIABLES
both continuous
e <- ggplot(mpg, aes(cty, hwy))

e + geom_label(aes(label = cty), nudge_x = 1, nudge_y = 1**)** - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

e + geom_point()
x, y, alpha, color, fill, shape, size, stroke

e + geom_quantile()
x, y, alpha, color, group, linetype, size, weight

e + geom_rug(sides = "bl"**)**
x, y, alpha, color, linetype, size

e + geom_smooth(method = lm**)**
x, y, alpha, color, fill, group, linetype, size, weight

e + geom_text(aes(label = cty), nudge_x = 1, nudge_y = 1**)** - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

continuous bivariate distribution
h <- ggplot(diamonds, aes(carat, price))

h + geom_bin2d(binwidth = c(0.25, 500)**)**
x, y, alpha, color, fill, linetype, size, weight

h + geom_density_2d()
x, y, alpha, color, group, linetype, size

h + geom_hex()
x, y, alpha, color, fill, size

continuous function
i <- ggplot(economics, aes(date, unemploy))

i + geom_area()
x, y, alpha, color, fill, linetype, size

i + geom_line()
x, y, alpha, color, group, linetype, size

i + geom_step(direction = "hv"**)**
x, y, alpha, color, group, linetype, size

2 variable geoms: numeric & categorical

Continuous/numeric X & Discrete/Categorical Y

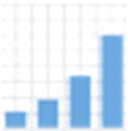
- **Bar chart**

Discrete/Categorical X & Continuous/numeric Y

- **Box plot**
- **Dot plot**
- **Violin chart**

one discrete, one continuous

f <– ggplot(mpg, aes(class, hwy))



f + geom_col()
x, y, alpha, color, fill, group, linetype, size

f + geom_boxplot()
x, y, lower, middle, upper, ymax, ymin, alpha, color, fill, group, linetype, shape, size, weight

f + geom_dotplot(binaxis = "y", stackdir = “center")
x, y, alpha, color, fill, group

f + geom_violin(scale = “area")
x, y, alpha, color, fill, group, linetype, size, weight

2 variable geoms: categorical

Discrete/categorical X & Y

- Mosaic charts
- Counts

both discrete

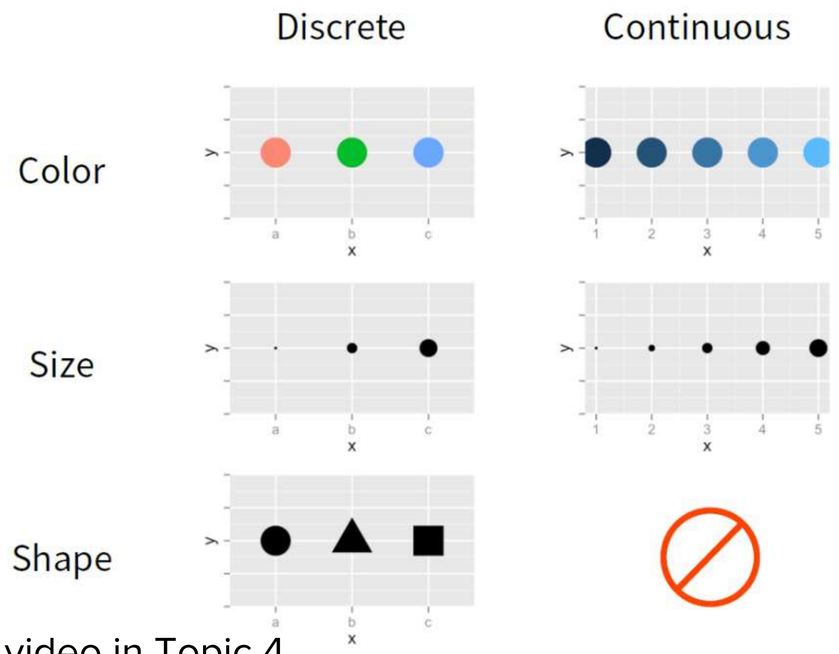
g <- ggplot(diamonds, aes(cut, color))

g + geom_count()
x, y, alpha, color, fill, shape, size, stroke

e + geom_jitter(height = 2, width = 2**)**
x, y, alpha, color, fill, shape, size

When visualising 3 variables

Start with a 2-variable visualization, then add color, shape, or size:

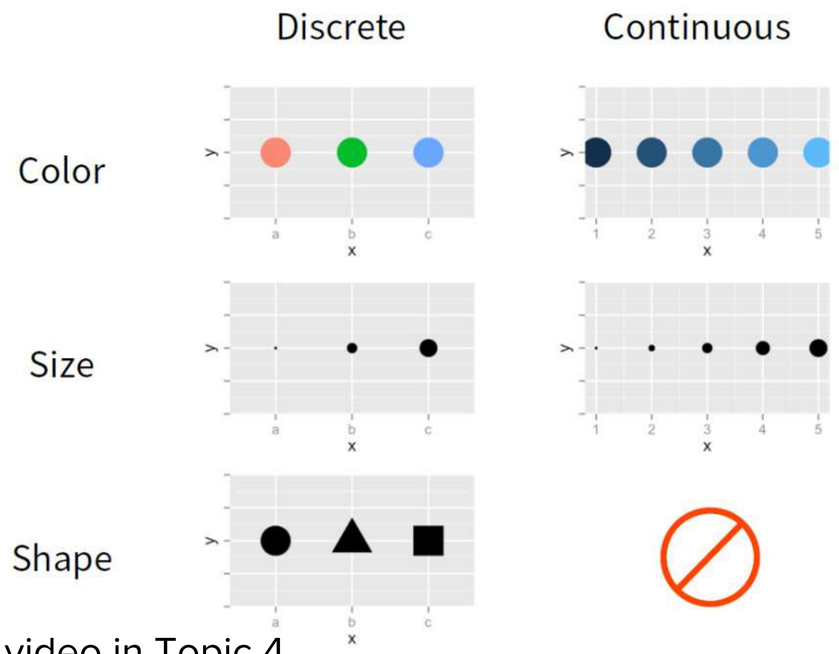


From Dr Sam Tyner's guest lecture video in Topic 4

When visualising 3+variables

Start with a 2-variable visualization, then add color, shape, or size:

Add another variable with Facets



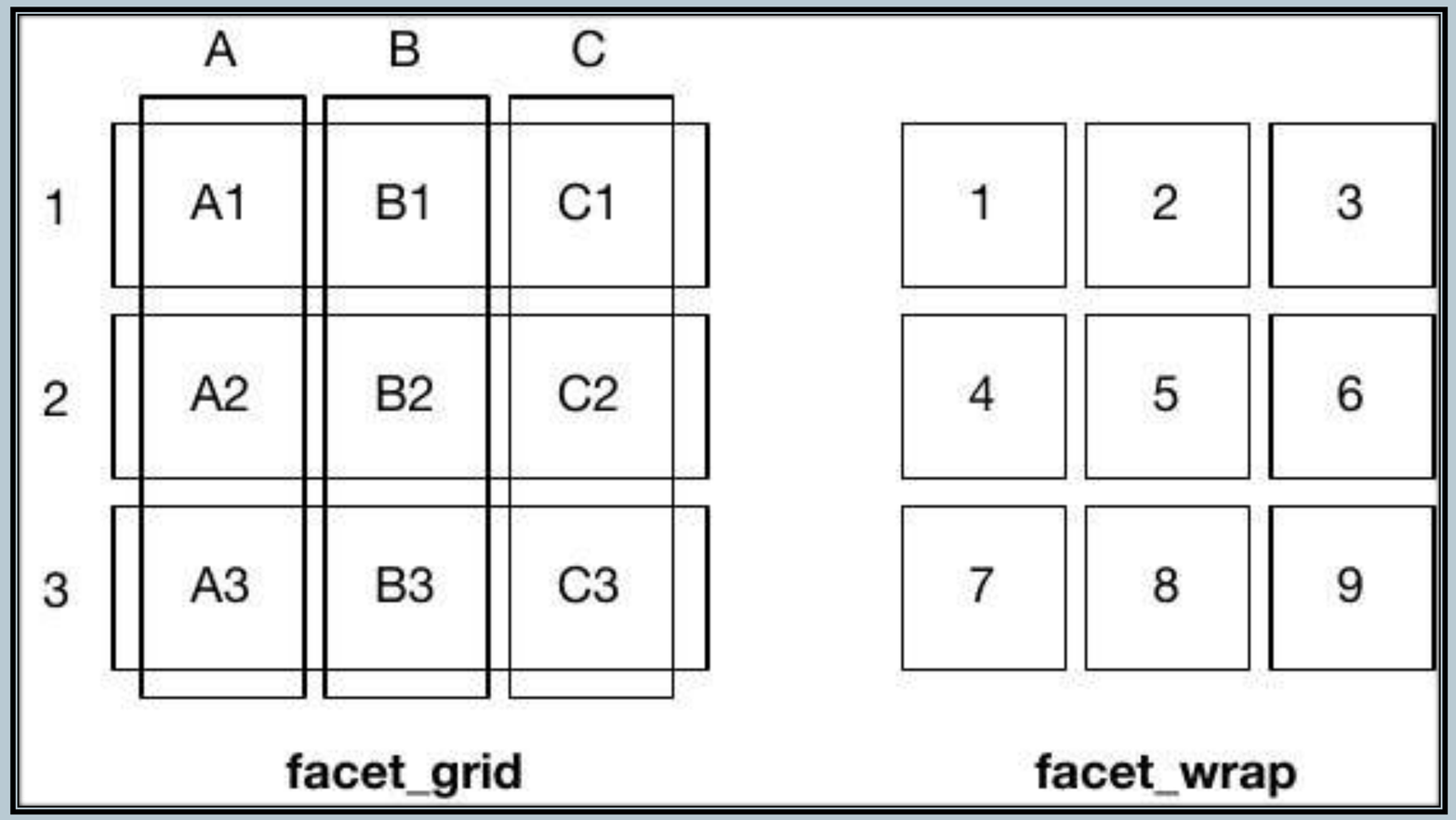
From Dr Sam Tyner's guest lecture video in Topic 4

Visualising 3+ variables: Facets

Fundamentally 2 dimensional

`+facet_grid(row ~ column)`

Makes a matrix of panels by row & column faceting variables

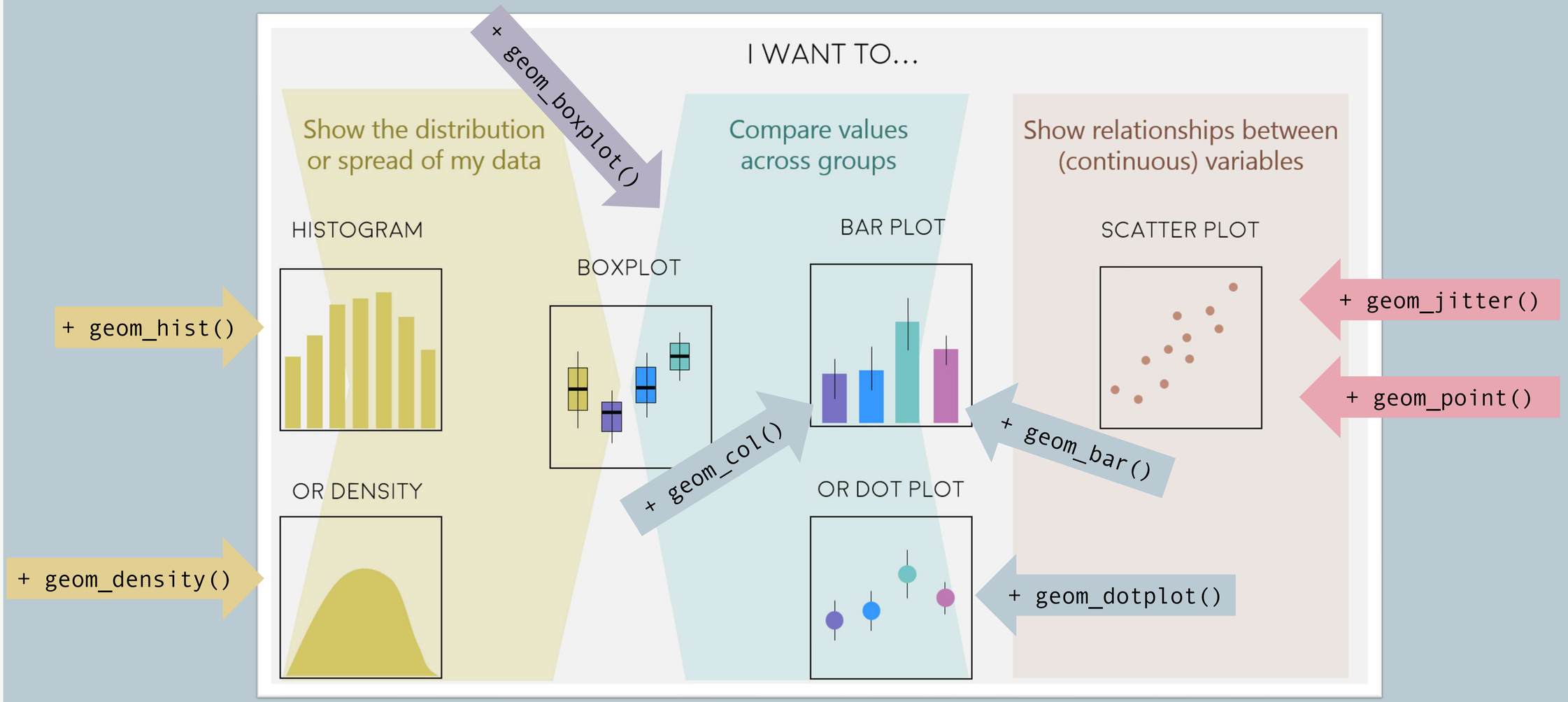


Fundamentally 1 dimensional

`+facet_wrap(variable)`

Makes a long ribbon of panels

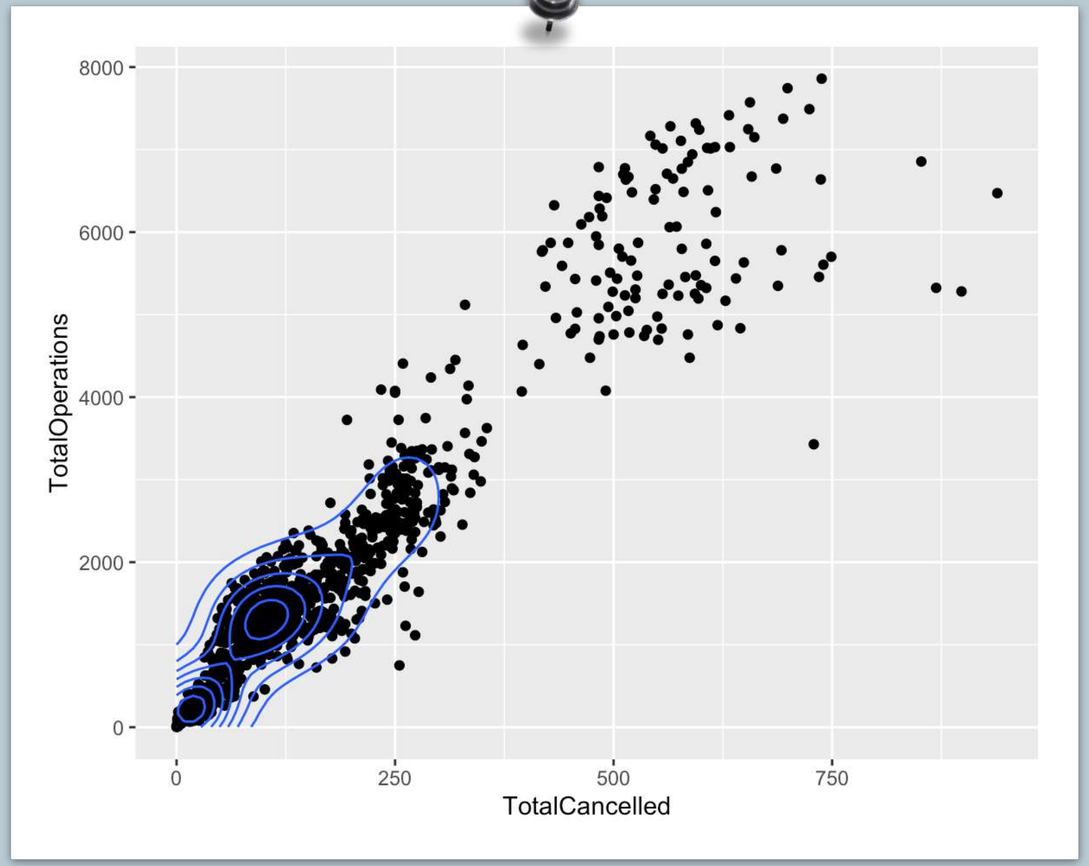
Geoms: decisions, decisions



ggplot2 layers

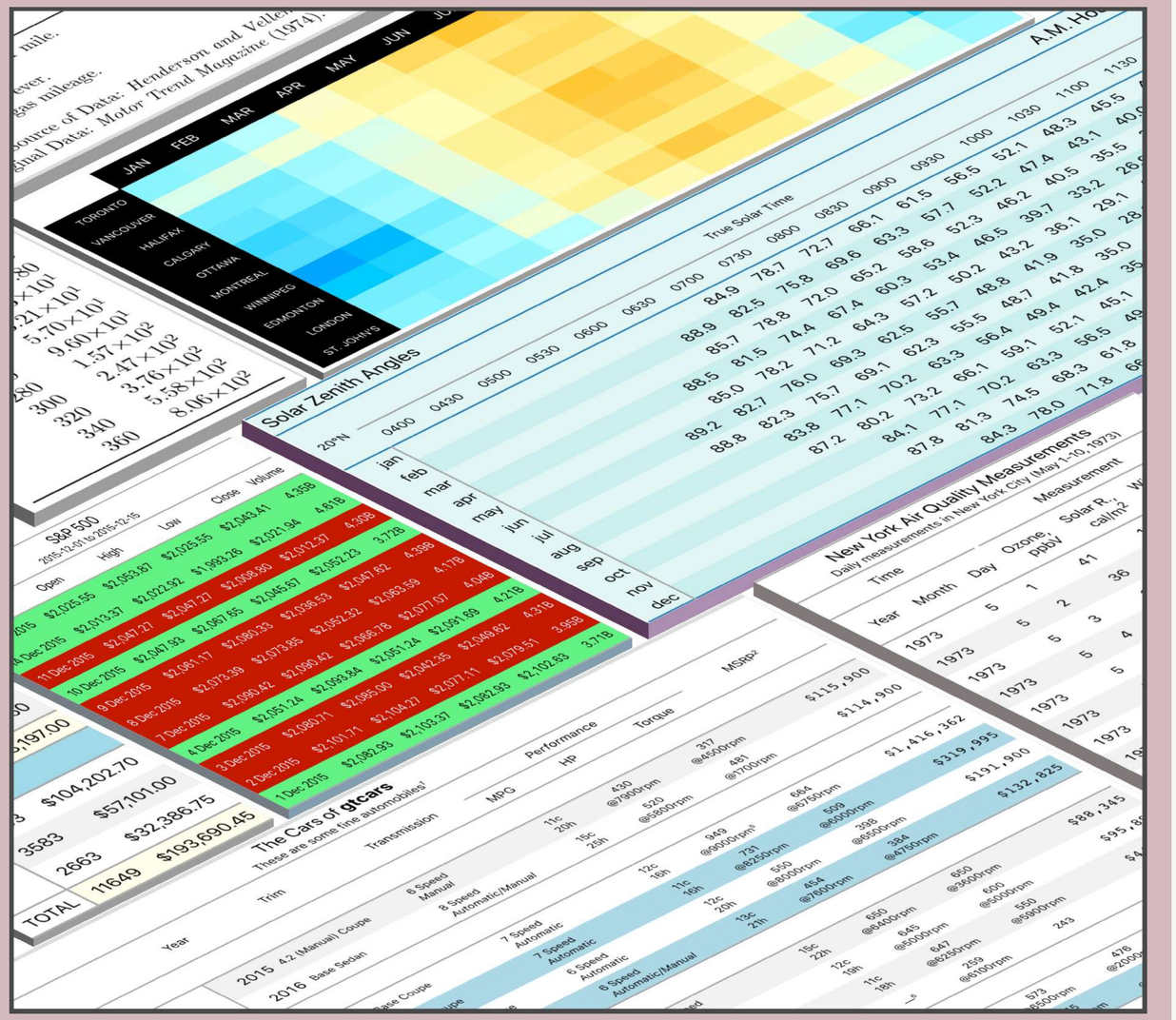
- Stacked in order of code appearance
- Important to keep in mind as elements written later in your code may hide or overwrite previous elements

```
ggplot(cancelled, aes(TotalCancelled, TotalOperations)) +
  geom_point() +
  geom_density2d()
```

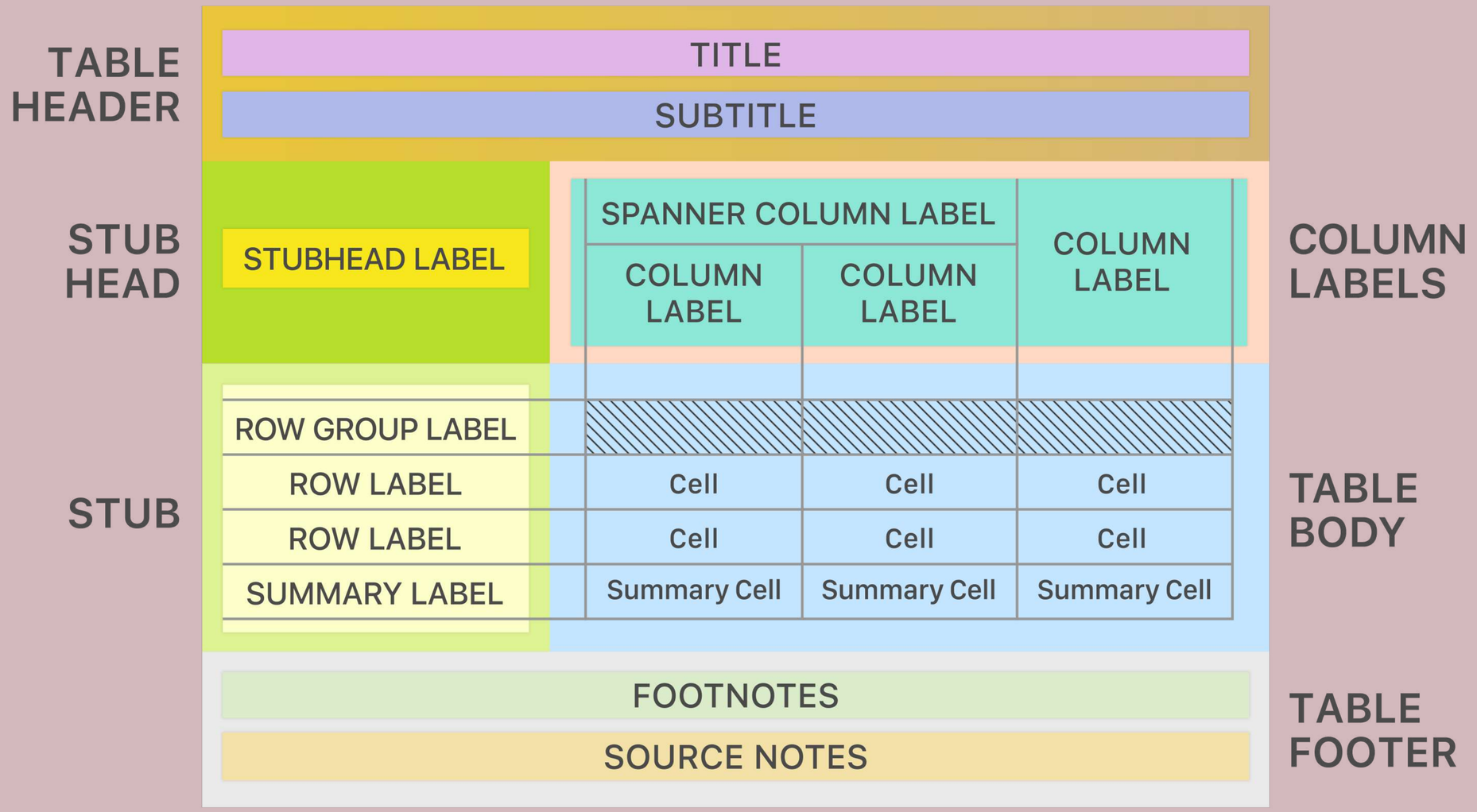


gt tables

- **Grammar of tables**
- **Pipe wrangled data to %>% `gt()`**



The Parts of a gt Table



`gt` covered in more detail in Week 6!

Create Table

`gt()`	Create a gt table object
`gt_preview()`	Preview a gt table object

Create/Modify Parts

`tab_header()`	Add a table header
`tab_spanner()`	Add a spanner column label
`tab_row_group()`	Add a row group
`tab_stubhead_label()`	Add label text to the stubhead
`tab_footnote()`	Add a footnote
`tab_source_note()`	Add a source note citation
`tab_options()`	Modify the table output options
`tab_style()`	Add custom styles to one or more cells

Format Data

`fmt()`	Set a column format with a formatter function
`fmt_number()`	Format numeric values
`fmt_scientific()`	Format values to scientific notation
`fmt_percent()`	Format values as a percentage
`fmt_currency()`	Format values as currencies
`fmt_date()`	Format values as dates
`fmt_time()`	Format values as times
`fmt_datetime()`	Format values as date-times
`fmt_markdown()`	Format Markdown text
`fmt_missing()`	Format missing values
`fmt_passthrough()`	Format by simply passing data through
`text_transform()`	Perform targeted text transformation with a function
`data_color()`	Set data cell colors using a palette or a color function

Modify Columns

`cols_align()`	Set the alignment of columns
`cols_hide()`	Hide one or more columns
`cols_label()`	Relabel one or more columns
`cols_merge()`	Merge two columns to a single column
`cols_merge_range()`	Merge two columns to a value range column
`cols_merge_uncert()`	Merge two columns to a value & uncertainty column
`cols_move()`	Move one or more columns
`cols_move_to_end()`	Move one or more columns to the end
`cols_move_to_start()`	Move one or more columns to the start
`cols_split_delim()`	Create group names and column labels via delimited names

Modify Rows

`row_group_order()`	Modify the ordering of any row groups

Add Rows

`summary_rows()`	Add summary rows using aggregation functions

Export Table

`gtsave()`	Save a gt table as a file
`as_raw_html()`	Get the HTML content of a gt table
`as_latex()`	Output a gt object as LaTeX
`as_rtf()`	Save a gt object as an RTF file
`extract_summary()`	Extract a summary list from a gt object

Shiny

`render_gt()`	A gt table render function for use in Shiny
`gt_output()`	Create a gt table output element for Shiny

Information

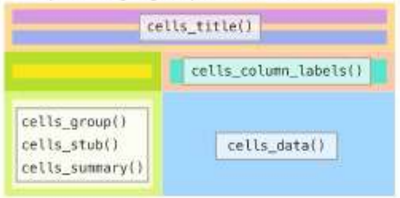
`info_date_style()`	View a table with info on date styles
`info_time_style()`	View a table with info on time styles
`info_paletteer()`	View a table with info on color palettes
`info_currencies()`	View a table with info on supported currencies
`info_locales()`	View a table with info on supported locales

Datasets

`countrypops`	Yearly populations of countries from 1960 to 2017
`sza`	Twice hourly solar zenith angles by month & latitude
`gtcars`	Deluxe automobiles from the 2014–2017 period
`sp500`	Daily S&P 500 Index data from 1950 to 2015
`pizzaplace`	A year of pizza sales from a pizza place
`exibble`	A toy example tibble for testing with gt: exibble

Location Helpers

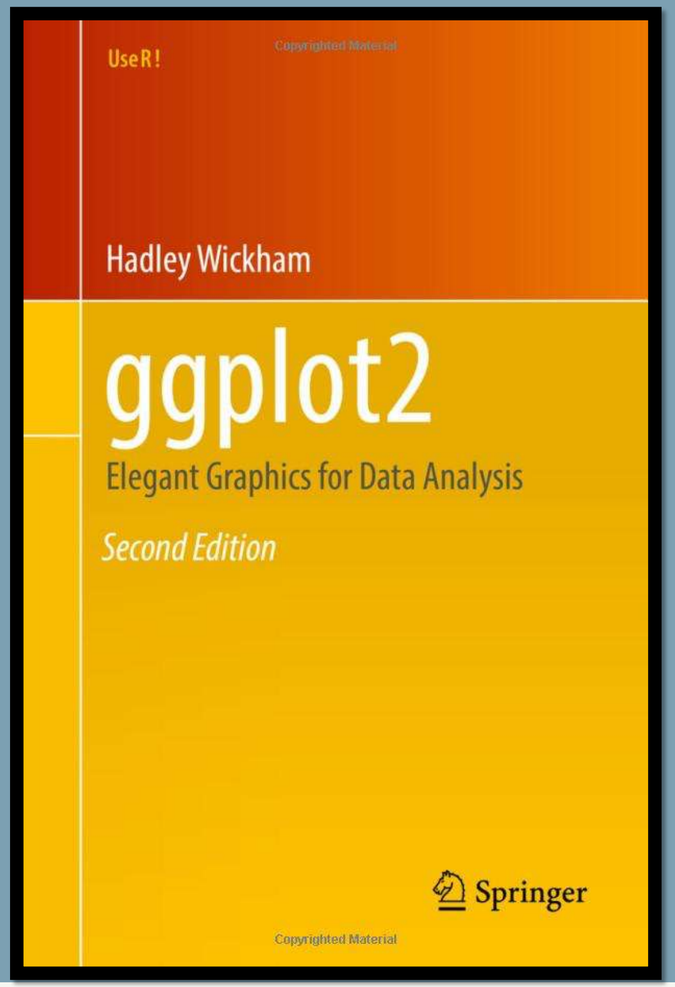
Helpers for targeting multiple cells in different locations



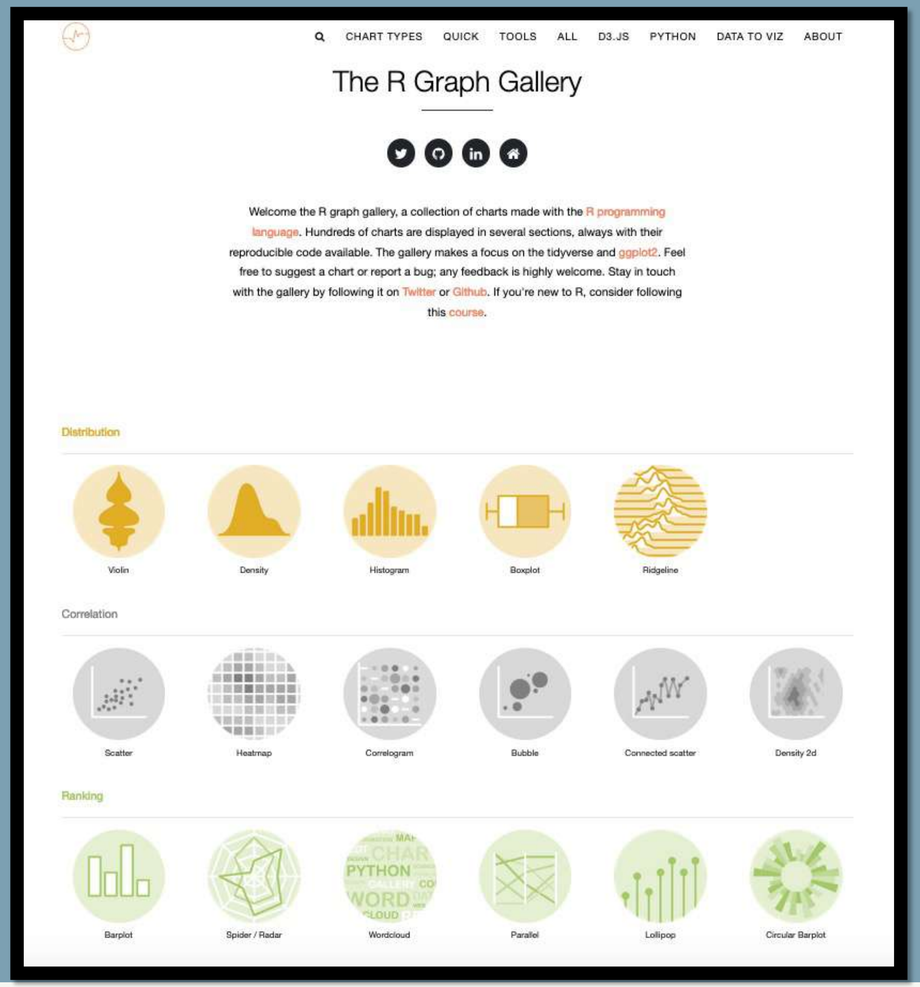
Credit: [Daniel J Sjoberg]

Further Resources

[(free!) ggplot2 book]



[R Graph Gallery]



Questions?