Graphics

Our interest here is exploratory graphics.

That means not [that] pretty.

To display data, you encode the values to a visual property.

Example	Encoding	Ordered	Useful values	Quantitative	Ordinal	Categorical	Relational
	position, placement	yes	infinite	Good	Good	Good	Good
1, 2, 3; A, B, C	text labels	optional alpha or num	infinite	Good	Good	Good	Good
	length	yes	many	Good	Good		
	size, area	yes	many	Good	Good		
	angle	yes	medium	Good	Good		
	pattern density	yes	few	Good	Good		
	weight, boldness	yes	few		Good		
	saturation, brightness	yes	few		Good		
	color	no	few (<20)			Good	
	shape, icon	no	medium			Good	

Ware, Information Visualization: Perception for Design (Morgan Kaufmann), p. 179.

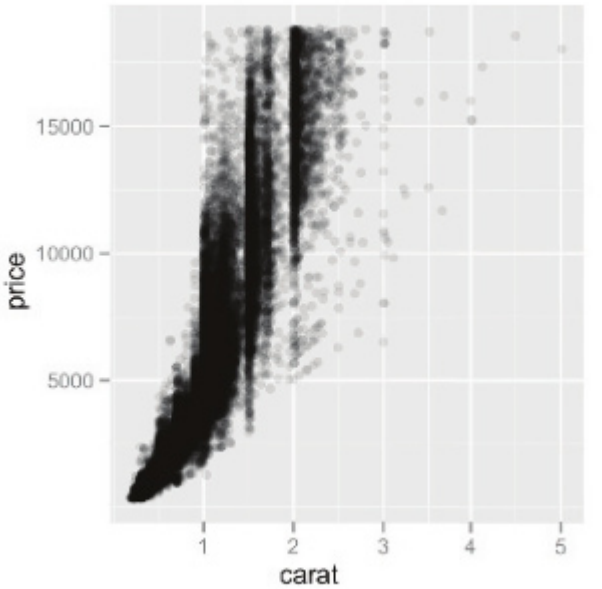
Not all visual properties are born equal.

Example	Encoding	Ordered	Useful values	Quantitative	Ordinal	Categorical	Relational
	position, placement	yes	infinite	Good	Good	Good	Good
1, 2, 3; A, B, C	text labels	optional alpha or num	infinite	Good	Good	Good	Good
	length	yes	many	Good	Good		
	size, area	yes	many	Good	Good		
	angle	yes	medium	Good	Good		
	pattern density	yes	few	Good	Good		
	weight, boldness	yes	few		Good		
	saturation, brightness	yes	few		Good		
	color	no	few (<20)			Good	
	shape, icon	no	medium			Good	

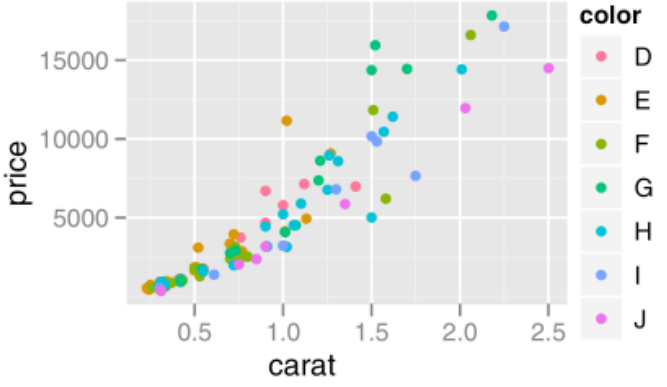
Ware, Information Visualization: Perception for Design (Morgan Kaufmann), p. 179.

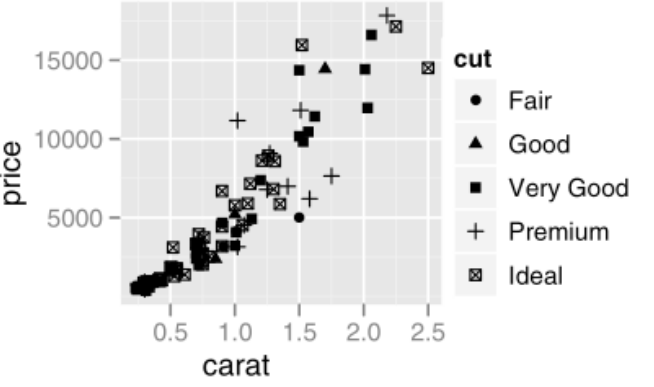
The "**Grammar of graphics**"
is a realization of the fact that many of the common plots are just combinations of those encodings.

Scatter plot is a combination of two positional encodings.

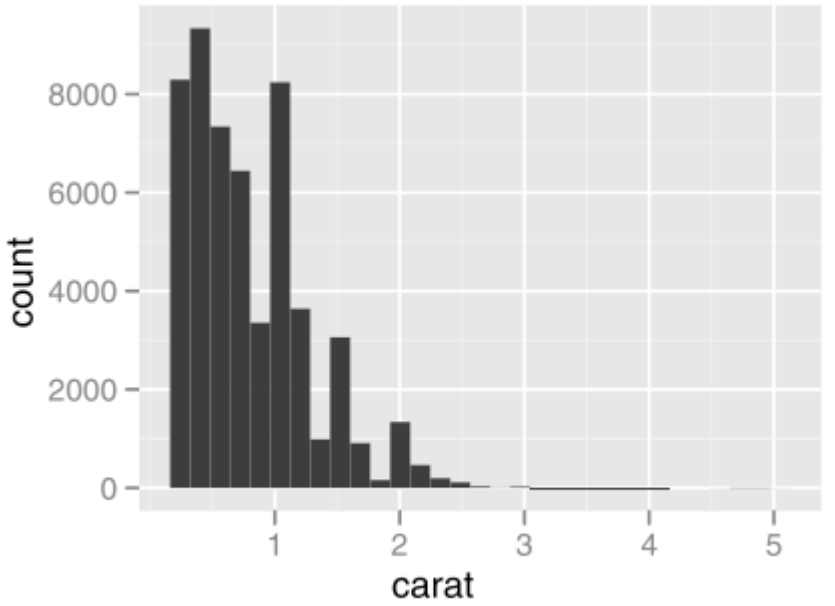


Scatter plot is a combination of two positional encodings. Additional encodings are possible.





Bar chart is a combination of positional and length encodings.



ggplot2 is a concise language for expressing of the mapping from data to geometry.

```
> ggplot(d, aes(carat, price)) + geom_point()
```

- maps d$carat to x, d$price to y
- adds a layer with points at x, y

ggplot2 is a concise language for expressing of the mapping from data to geometry.

```
> ggplot(d, aes(carat, price, colour=color)) +
geom_point()
```

- maps d$carat to x, d$price to y, d$color to colour
- adds a layer with points at x, y

ggplot2 is a concise language for expressing of the mapping from data to geometry.

```
> ggplot(d, aes(carat)) + geom_histogram()
```

- maps d$carat to x
- summarizes x as a histogram
- adds a layer with bars at histogram bins

ggplot works best with one particular
way of organizing data in the tables: "**tidy data**"

- variables in columns
- observations in rows

	Pregnant	Not pregnant
Male	0	5
Female	1	4

There are three variables in this data set. What are they?

from Hadley Wickham's talk on Tidy data, June 2012

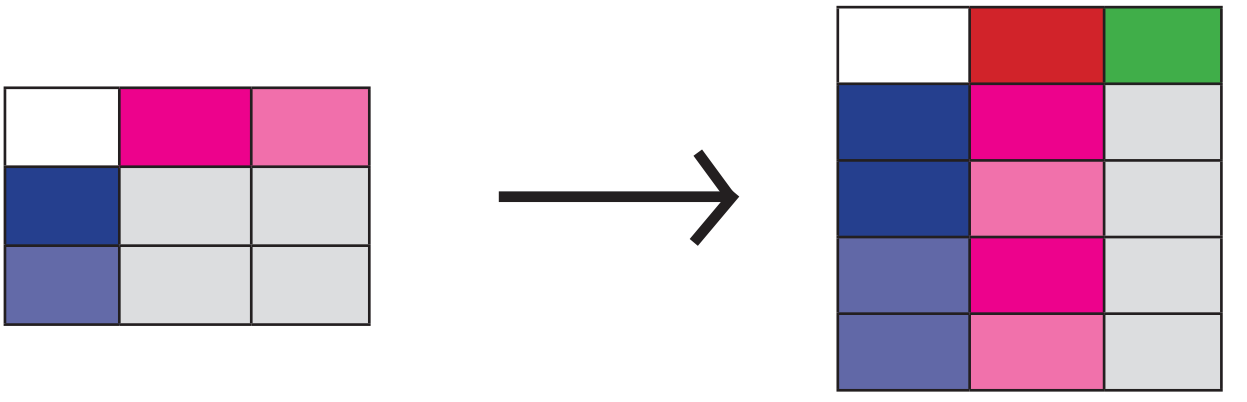
pregnant	sex	n
no	female	4
no	male	5
yes	female	1
yes	male	0

from Hadley Wickham's talk on Tidy data, June 2012

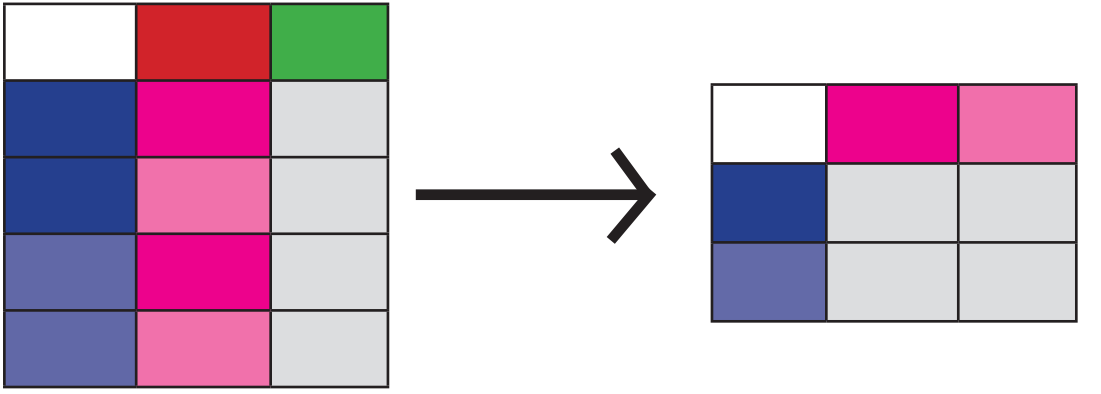
The **tidyr** package

- gather() gathers more columns into one
- extract() splits one column into more (keeping the number of rows)
- spread() complements gather

gather(key, value, columns)
gathers more columns into one

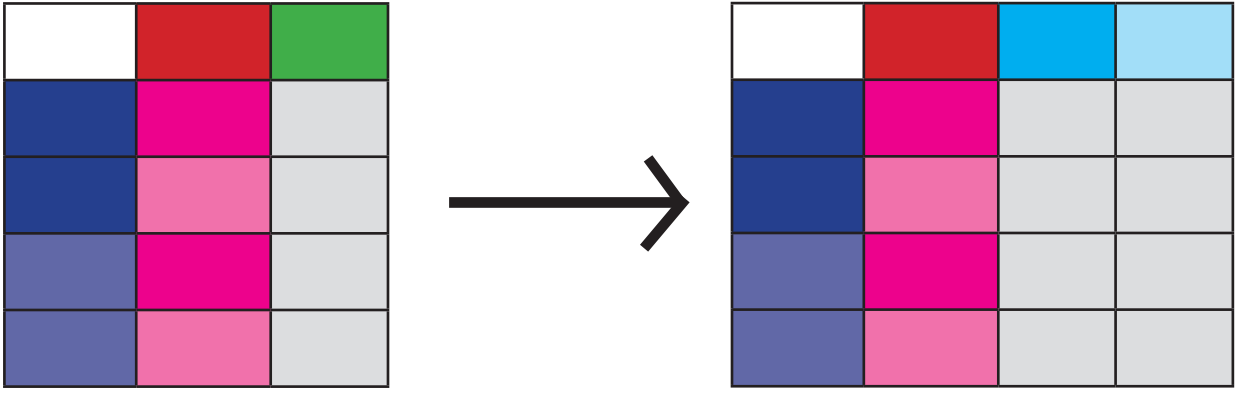


spread(key, value)
is the counterpart of gather



extract(col, into, regex)

splits a column using a regular expression



Tidying up the 'pregnancy' example

```
dd <- data.frame(
     row.names=c('Male', 'Female'),
     Pregnant=c(0, 1),
     `Non Pregnant`=c(5, 4))

library(dplyr)
library(tidyr)
dd %>%
  mutate(sex=row.names(.)) %>%
  gather(status, count, 1:2)
```