WEBINAR SERIES: ESSENTIAL TOOLS FOR DATA SCIENCE WITH R

The Grammar and Graphics of Data Science

#RStudio

The next webinar in the series:
“**Reproducible Reporting**” — Live!
Wednesday, August 13th, 11am Eastern Time US

Master R Developer Workshop,
Monday, September 8 – Tuesday, September 9, 2014.
New York City, NY.

R Day @StrataNYC + Hadoop World,
October 15th.
Javits Center, New York City, NY.

Grammars of data science

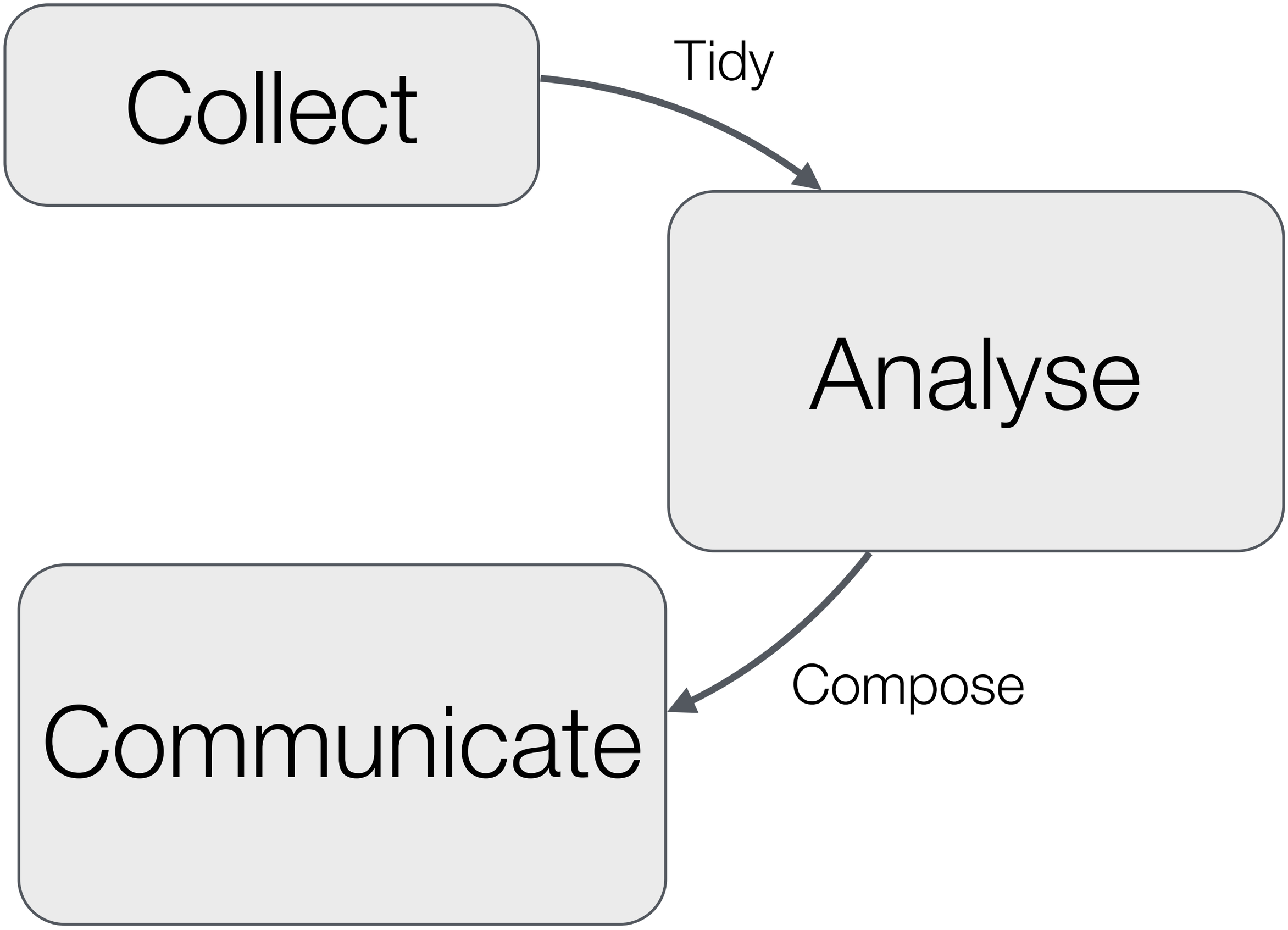
Hadley Wickham

@hadleywickham

Chief Scientist, RStudio

July 2014

What is data science?


Collect
Tidy
Analyse
Compose
Communicate

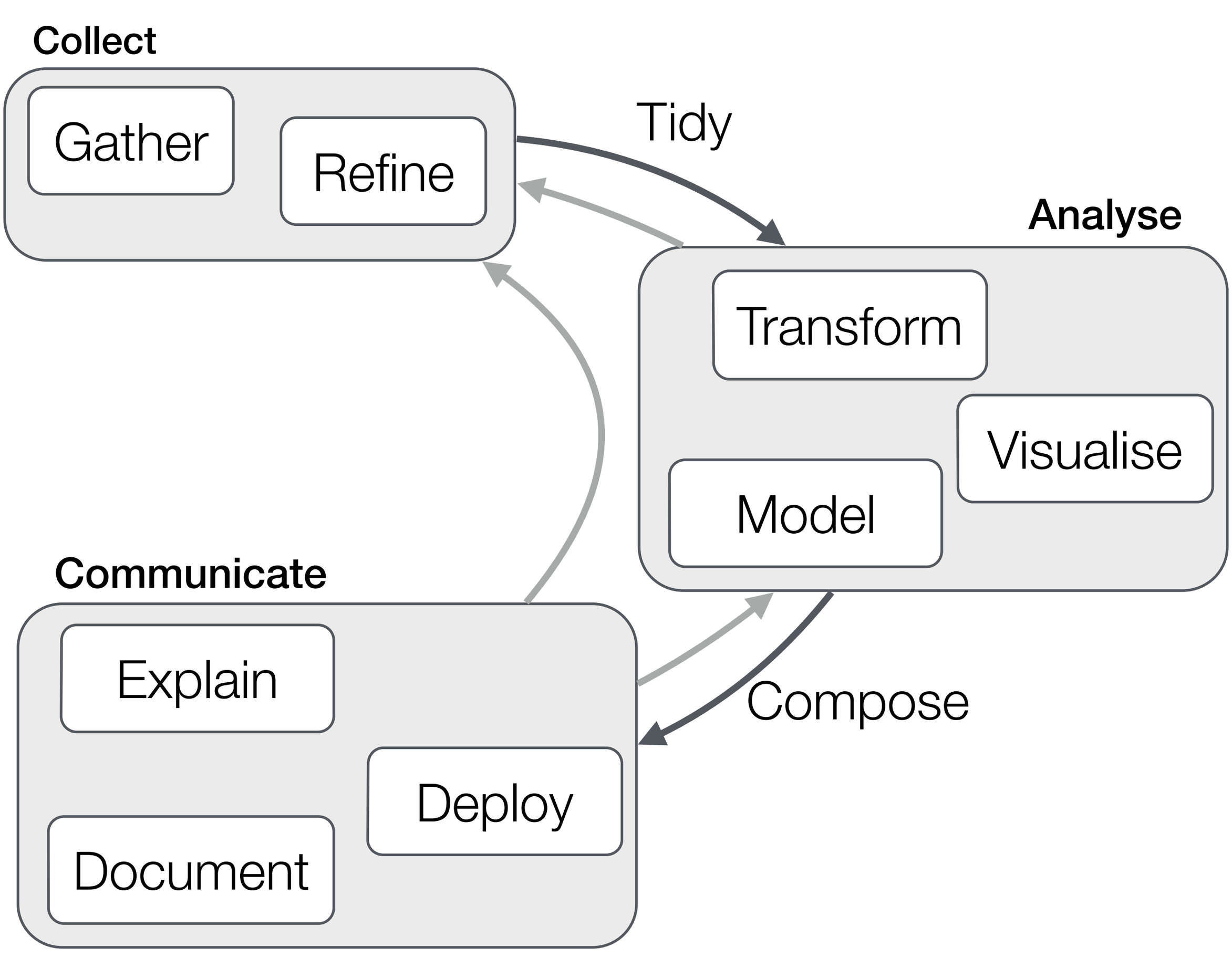
Collect

Tidy

Analyse

Compose

Communicate


Collect
Gather
Refine
Tidy
Analyse
Transform
Visualise
Model
Communicate
Explain
Deploy
Document
Compose

Tidy

Transform

Visualise

Model

Tidy

tidyr

Transform

dplyr

Visualise

ggvis

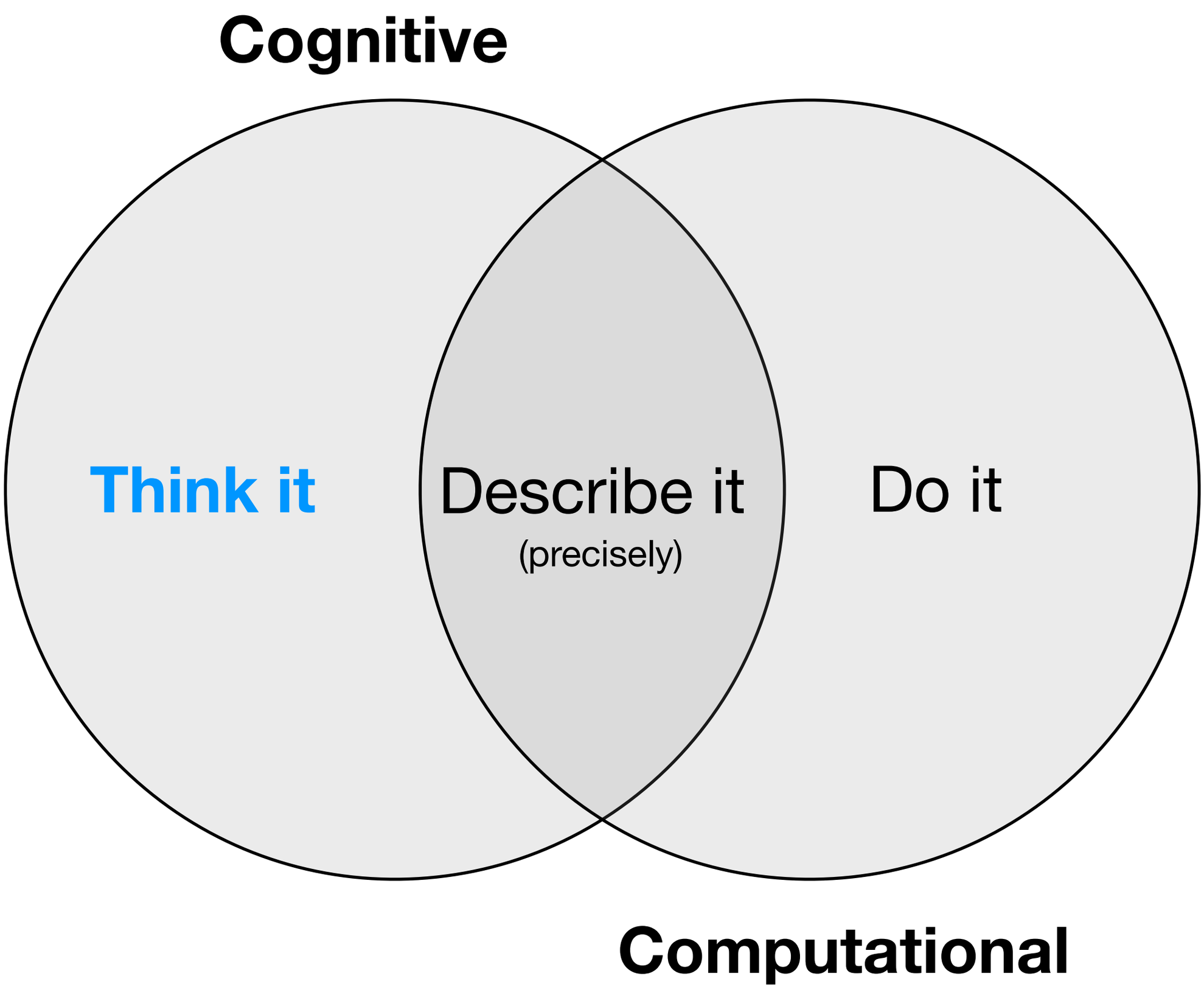
Model

tidyr

What is tidy data?

- Data that's easy to transform, visualise and model
- Key idea: store variables in a consistent way, always as columns
- tidyr provides useful tools to tidy messy data. Three most important are: **gather**, **spread** and **separate**.
- Google "tidy data" for more details.

dplyr


Cognitive
Think it
Describe it
(precisely)
Do it
Computational

+ group by

- **filter**: keep rows matching criteria
- **select**: pick columns by name
- **arrange**: reorder rows
- **mutate**: add new variables
- **summarise**: reduce variables to values

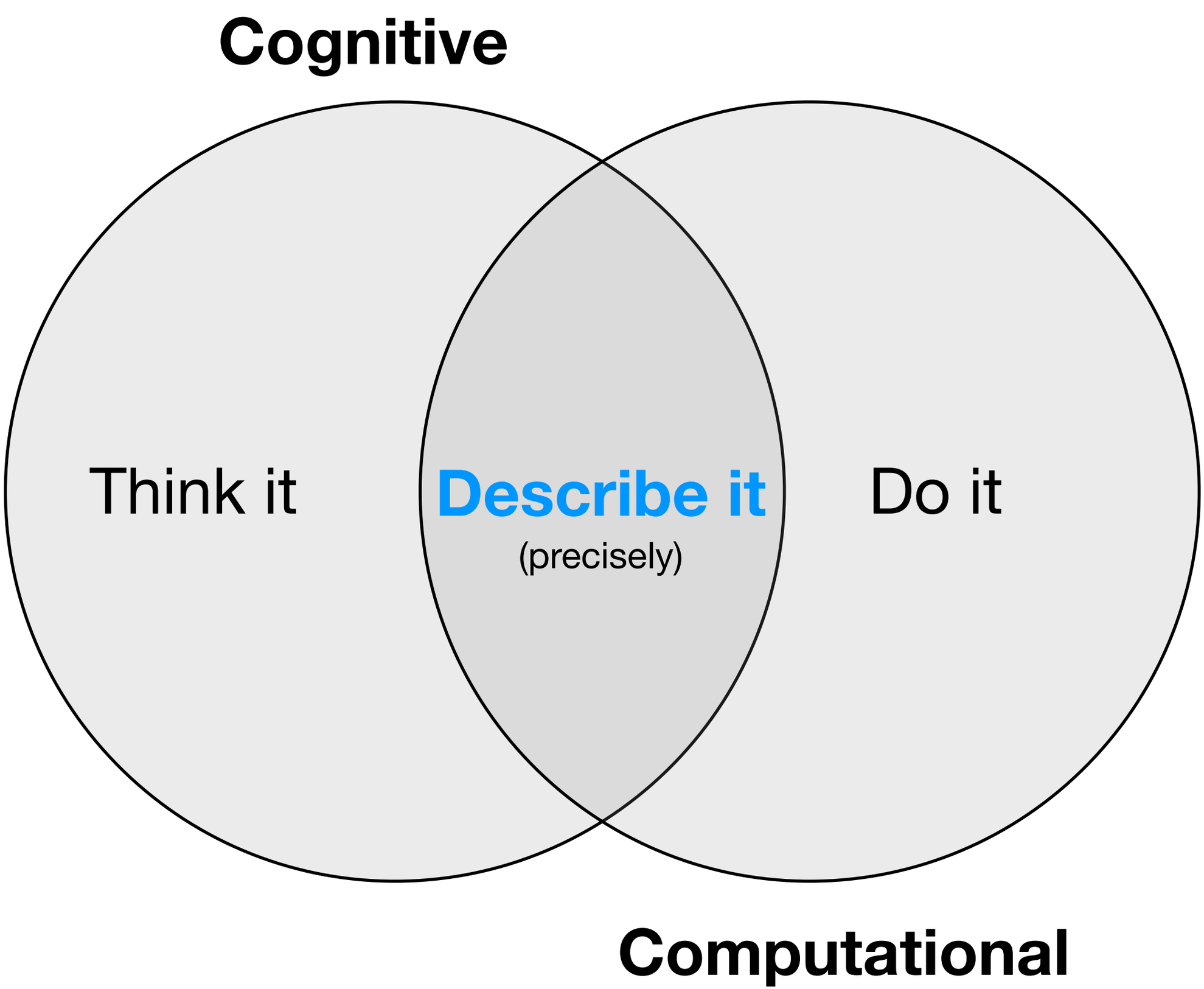
nycflights13

- `flights` [336,776 x 16]. Every flight departing NYC in 2013.
- `weather` [8,719 x 14]. Hourly weather data.
- `planes` [3,322 x 9]. Plane metadata.
- `airports` [1,397 x 7]. Airport metadata.

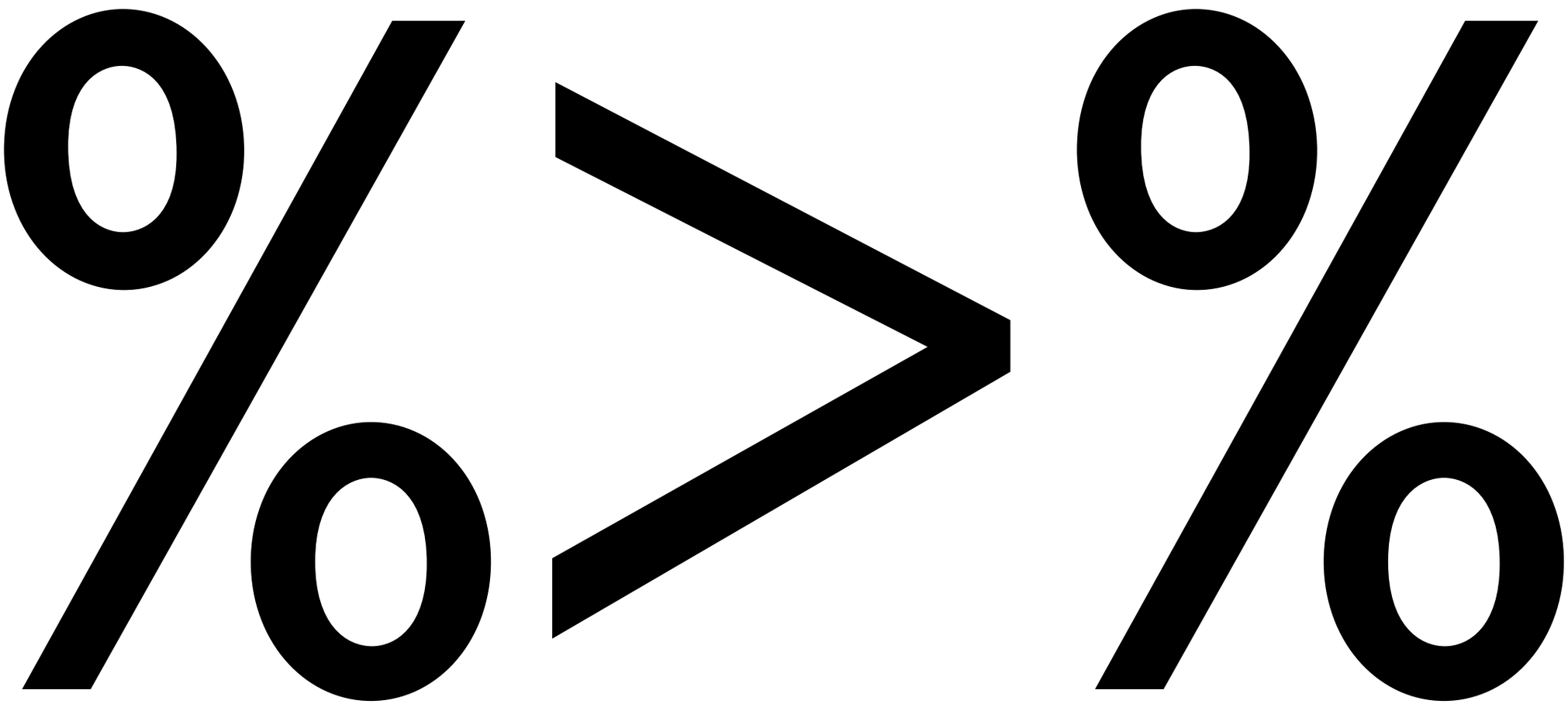
```
library(nycflights13)
library(dplyr)

flights
#> Source: local data frame [336,776 x 16]
#>
#>    year month day dep_time dep_delay arr_time arr_delay carrier tailnum
#> 1  2013     1   1      517         2      830        11      UA  N14228
#> 2  2013     1   1      533         4      850        20      UA  N24211
#> 3  2013     1   1      542         2      923        33      AA  N619AA
#> 4  2013     1   1      544        -1     1004       -18      B6  N804JB
#> 5  2013     1   1      554        -6      812       -25      DL  N668DN
#> 6  2013     1   1      554        -4      740        12      UA  N39463
#> 7  2013     1   1      555        -5      913        19      B6  N516JB
#> 8  2013     1   1      557        -3      709       -14      EV  N829AS
#> 9  2013     1   1      557        -3      838        -8      B6  N593JB
#> 10 2013     1   1      558        -2      753         8      AA  N3ALAA
#> ..  ...   ... ...      ...       ...      ...       ...     ...     ...
#> Variables not shown: flight (int), origin (chr), dest (chr),
#>   air_time (dbl), distance (dbl), hour (dbl), minute (dbl)
```

Demo

Pipelines


Cognitive
Think it
Describe it
(precisely)
Do it
Computational

```
hourly_delay <- filter(
  summarise(
    group_by(
      filter(
        flights,
        !is.na(dep_delay)
      ),
      date, hour
    ),
    delay = mean(dep_delay),
    n = n()
  ),
  n > 10
)
```



tidyr, dplyr, ggvis, ...

```r
# x %>% f(y) -> f(x, y)

hourly_delay <- flights %>%
  filter(!is.na(dep_delay)) %>%
  group_by(date, hour) %>%
  summarise(
    delay = mean(dep_delay),
    n = n()
  ) %>%
  filter(n > 10)
```

Remote sources

Cognitive

Think it

Describe it
(precisely)

Do it
(remote)

Computational

Other data sources

- PostgreSQL, redshift
- MySQL, MariaDB
- SQLite
- MonetDB, BigQuery
- *Oracle*, *SQL Server, Greenplum, ImpalaDB*

```r
flights %>%
  filter(!is.na(dep_delay)) %>%
  group_by(date, hour) %>%
  summarise(delay = mean(dep_delay), n = n()) %>%
  filter(n > 10)

# SELECT "date", "hour", "delay", "n"
# FROM (
#   SELECT "date", "hour",
#     AVG("dep_delay") AS "delay",
#     COUNT() AS "n"
#   FROM "flights"
#   WHERE NOT("dep_delay" IS NULL)
#   GROUP BY "date", "hour"
# ) AS "_W1"
# WHERE "n" > 10.0
```

```
translate_sql(month > 1, flights)
# <SQL> "Month" > 1.0
translate_sql(month > 1L, flights)
# <SQL> "Month" > 1

translate_sql(dest == "IAD" || dest == "DCA",
  hflights)
# <SQL> "dest" = 'IAD' OR "dest" = 'DCA'

dc <- c("IAD", "DCA")
translate_sql(dest %in% dc, flights)
# <SQL> "dest" IN ('IAD', 'DCA')
```

Learn more

```
# Built-in vignettes
browseVignettes(package = "dplyr")

# Translate plyr to dplyr
http://jimhester.github.io/plyrToDplyr/

# Common questions & answers
http://stackoverflow.com/questions/tagged/dplyr?sort=frequent
```