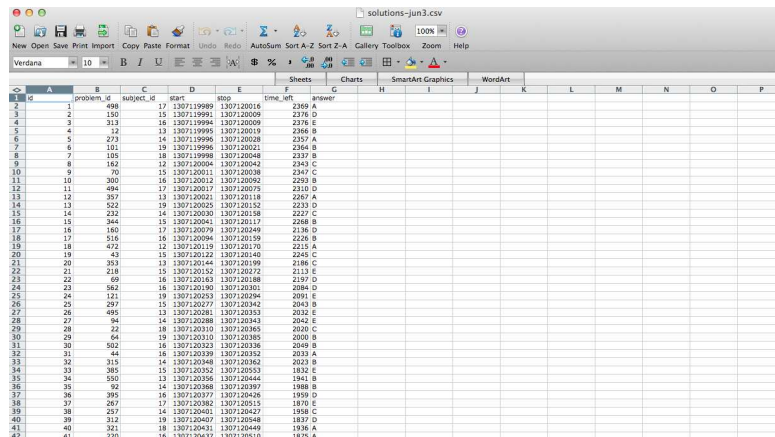




# Reshaping data

Jeffrey Leek  
Johns Hopkins Bloomberg School of Public Health

# The goal is tidy data



The screenshot shows a spreadsheet application window titled 'solutions-jun3.csv'. The spreadsheet has columns labeled A through P. The data is organized into rows, with the first row (row 1) containing headers: 'id', 'subject\_id', 'visit', 'date', and 'answer'. The subsequent rows (rows 2 through 42) contain numerical data for each of these variables. For example, row 2 has values 1, 100, 1, 130719999, and 2396 A. The data continues in this pattern for all 42 rows.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	id	subject_id	visit	date	answer											
2	1	100	1	130719999	2396 A											
3	2	150	15	130719991	2376 D											
4	3	112	16	130719984	2376 E											
5	4	12	13	130719995	2366 B											
6	5	273	14	130719996	2357 A											
7	6	101	19	130719998	2364 B											
8	7	105	18	130719998	2357 B											
9	8	162	12	130720004	2343 C											
10	9	70	15	130720011	2347 C											
11	10	300	16	130720012	2393 B											
12	11	494	17	130720017	2310 D											
13	12	397	13	130720021	2287 A											
14	13	522	19	130720025	2273 D											
15	14	232	14	130720030	2227 C											
16	15	344	15	130720041	2288 B											
17	16	160	17	130720079	2126 D											
18	17	516	16	130720094	2228 B											
19	18	472	12	130720119	2215 A											
20	19	43	15	130720122	2245 C											
21	20	353	13	130720144	2186 C											
22	21	218	15	130720152	2113 E											
23	22	69	16	130720163	2197 D											
24	23	362	16	130720180	2084 D											
25	24	121	19	130720253	2011 E											
26	25	297	15	130720277	2043 B											
27	26	495	13	130720281	2012 E											
28	27	94	14	130720288	2042 E											
29	28	22	18	130720310	2020 C											
30	29	64	19	130720310	2000 B											
31	30	562	16	130720323	2049 B											
32	31	44	16	130720339	2033 A											
33	32	315	14	130720348	2023 B											
34	33	385	15	130720352	1932 E											
35	34	590	13	130720356	1941 B											
36	35	92	14	130720368	1988 B											
37	36	395	16	130720377	1959 D											
38	37	267	17	130720382	1970 E											
39	38	157	14	130720401	1984 C											
40	39	312	19	130720407	1937 D											
41	40	321	18	130720431	1936 A											
42	41	220	16	130720437	1874 A											

1. Each variable forms a column
2. Each observation forms a row
3. Each table/file stores data about one kind of observation (e.g. people/hospitals).

<http://vita.had.co.nz/papers/tidy-data.pdf>

[Leek, Taub, and Pineda 2011 PLoS One](#)

# Start with reshaping

```
library(reshape2)  
head(mtcars)
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

# Melting data frames

```
mtcars$carname <- rownames(mtcars)
carMelt <- melt(mtcars,id=c("carname","gear","cyl"),measure.vars=c("mpg","hp"))
head(carMelt,n=3)
```

	carname	gear	cyl	variable	value
1	Mazda RX4	4	6	mpg	21.0
2	Mazda RX4 Wag	4	6	mpg	21.0
3	Datsun 710	4	4	mpg	22.8

```
tail(carMelt,n=3)
```

	carname	gear	cyl	variable	value
62	Ferrari Dino	5	6	hp	175
63	Maserati Bora	5	8	hp	335
64	Volvo 142E	4	4	hp	109

# Casting data frames

```
cylData <- dcast(carMelt, cyl ~ variable)
cylData
```

```
   cyl mpg hp
1    4  11 11
2    6   7  7
3    8  14 14
```

```
cylData <- dcast(carMelt, cyl ~ variable,mean)
cylData
```

```
   cyl   mpg   hp
1    4 26.66 82.64
2    6 19.74 122.29
3    8 15.10 209.21
```

# Averaging values

```
head(InsectSprays)
```

	count	spray
1	10	A
2	7	A
3	20	A
4	14	A
5	14	A
6	12	A

```
tapply(InsectSprays$count, InsectSprays$spray, sum)
```

A	B	C	D	E	F
174	184	25	59	42	200

# Another way - split

```
spIns = split(InsectSprays$count, InsectSprays$spray)
spIns
```

\$A

```
[1] 10 7 20 14 14 12 10 23 17 20 14 13
```

\$B

```
[1] 11 17 21 11 16 14 17 17 19 21 7 13
```

\$C

```
[1] 0 1 7 2 3 1 2 1 3 0 1 4
```

\$D

```
[1] 3 5 12 6 4 3 5 5 5 5 2 4
```

\$E

```
[1] 3 5 3 5 3 6 1 1 3 2 6 4
```

\$F

```
[1] 11 9 15 22 15 16 13 10 26 26 24 13
```

# Another way - apply

```
sprCount = lapply(spIns,sum)
sprCount
```

```
$A
[1] 174
```

```
$B
[1] 184
```

```
$C
[1] 25
```

```
$D
[1] 59
```

```
$E
[1] 42
```

```
$F
[1] 200
```



# Another way - combine

```
unlist(sprCount)
```

A	B	C	D	E	F
174	184	25	59	42	200

```
sapply(spIns,sum)
```

A	B	C	D	E	F
174	184	25	59	42	200

# Another way - plyr package

```
ddply(InsectSprays,.(spray),summarize,sum=sum(count))
```

	spray	sum
1	A	174
2	B	184
3	C	25
4	D	59
5	E	42
6	F	200

# Creating a new variable

```
spraySums <- ddply(InsectSprays,.(spray),summarize,sum=ave(count,FUN=sum))  
dim(spraySums)
```

```
[1] 72 2
```

```
head(spraySums)
```

```
  spray sum  
1     A 174  
2     A 174  
3     A 174  
4     A 174  
5     A 174  
6     A 174
```

# More information

- A tutorial from the developer of plyr - <http://plyr.had.co.nz/09-user/>
- A nice reshape tutorial <http://www.slideshare.net/jeffreybreen/reshaping-data-in-r>
- A good plyr primer - <http://www.r-bloggers.com/a-quick-primer-on-split-apply-combine-problems/>
- See also the functions
  - `acast` - for casting as multi-dimensional arrays
  - `arrange` - for faster reordering without using `order()` commands
  - `mutate` - adding new variables