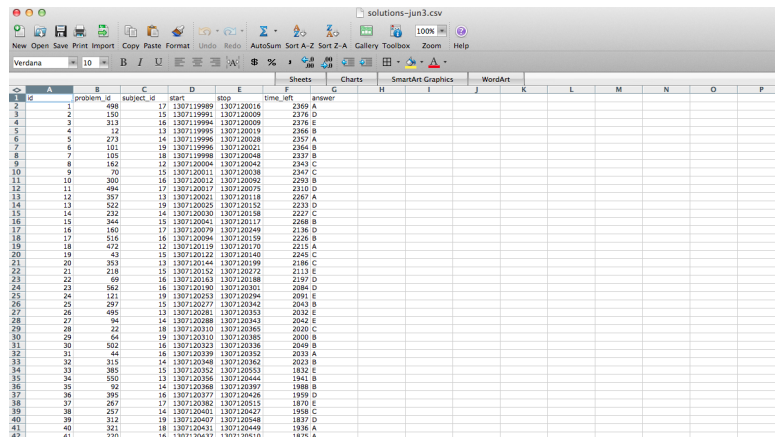




Reshaping data

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 contains a table with 7 columns: 'id', 'enrolment_id', 'subject_id', 'start', 'stop', 'time_id', and 'answer'. The data is organized in rows, with each row representing an observation. The 'id' column is the first column, followed by 'enrolment_id', 'subject_id', 'start', 'stop', 'time_id', and 'answer'. The 'start' and 'stop' columns contain dates in YYYYMMDD format. The 'time_id' column contains numerical values. The 'answer' column contains categorical values. The spreadsheet is displayed in a standard grid format with column letters (A-P) and row numbers (1-42) visible on the left side.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
	id	enrolment_id	subject_id	start	stop	time_id	answer									
1	1	68	17	130719999	130720016	2369	D									
2	2	150	15	130719991	130720009	2376	D									
3	3	113	16	130719994	130720009	2376	E									
4	4	12	13	130719995	130720019	2366	B									
5	5	273	14	130719996	130720028	2357	A									
6	6	101	19	130719998	130720021	2364	B									
7	7	105	18	130719998	130720048	2357	B									
8	8	162	12	130720004	130720042	2343	C									
9	9	70	15	130720011	130720038	2347	C									
10	10	300	16	130720012	130720092	2393	B									
11	11	494	17	130720017	130720075	2310	D									
12	12	397	13	130720021	130720118	2287	A									
13	13	522	19	130720025	130720152	2233	D									
14	14	232	14	130720030	130720158	2227	C									
15	15	444	15	130720041	130720117	2288	B									
16	16	160	17	130720079	130720249	2136	D									
17	17	516	16	130720094	130720129	2228	B									
18	18	472	12	130720119	130720170	2215	A									
19	19	43	15	130720122	130720140	2245	C									
20	20	393	13	130720144	130720199	2186	C									
21	21	218	15	130720152	130720272	2113	E									
22	22	69	16	130720163	130720188	2197	D									
23	23	84	16	130720180	130720201	2084	D									
24	24	121	19	130720253	130720294	2001	E									
25	25	297	15	130720277	130720342	2043	B									
26	26	495	13	130720281	130720353	2032	E									
27	27	94	14	130720288	130720343	2042	E									
28	28	22	18	130720310	130720385	2020	C									
29	29	64	19	130720310	130720385	2000	B									
30	30	562	16	130720323	130720336	2049	B									
31	31	44	16	130720339	130720352	2033	A									
32	32	315	14	130720348	130720362	2023	B									
33	33	385	15	130720352	130720553	1932	E									
34	34	590	13	130720356	130720444	1941	B									
35	35	92	14	130720368	130720397	1988	B									
36	36	395	16	130720377	130720436	1959	D									
37	37	297	17	130720382	130720515	1870	E									
38	38	257	14	130720401	130720427	1948	C									
39	39	312	19	130720407	130720548	1837	D									
40	40	321	18	130720431	130720449	1938	A									
41	41	220	16	130720437	130720510											

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