



Subsetting and sorting

Jeffrey Leek
Johns Hopkins Bloomberg School of Public Health

Subsetting - quick review

```
set.seed(13435)
X <- data.frame("var1"=sample(1:5), "var2"=sample(6:10), "var3"=sample(11:15))
X <- X[sample(1:5),]; X$var2[c(1,3)] = NA
X
```

	var1	var2	var3
1	2	NA	15
4	1	10	11
2	3	NA	12
3	5	6	14
5	4	9	13

Subsetting - quick review

```
X[,1]
```

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

```
X[, "var1"]
```

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

```
X[1:2, "var2"]
```

```
[1] NA 10
```

Logicals ands and ors

```
X[ (X$var1 <= 3 & X$var3 > 11), ]
```

	var1	var2	var3
1	2	NA	15
2	3	NA	12

```
X[ (X$var1 <= 3 | x$var3 > 15), ]
```

	var1	var2	var3
1	2	NA	15
4	1	10	11
2	3	NA	12

Dealing with missing values

```
X[which(X$var2 > 8), ]
```

	var1	var2	var3
4	1	10	11
5	4	9	13

Sorting

```
sort(X$var1)
```

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

```
sort(X$var1,decreasing=TRUE)
```

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

```
sort(X$var2,na.last=TRUE)
```

```
[1] 6 9 10 NA NA
```

Ordering

```
X[order(X$var1), ]
```

	var1	var2	var3
4	1	10	11
1	2	NA	15
2	3	NA	12
5	4	9	13
3	5	6	14

Ordering

```
X[order(X$var1,X$var3),]
```

	var1	var2	var3
4	1	10	11
1	2	NA	15
2	3	NA	12
5	4	9	13
3	5	6	14

Ordering with plyr

```
library(plyr)  
arrange(X,var1)
```

	var1	var2	var3
1	1	10	11
2	2	NA	15
3	3	NA	12
4	4	9	13
5	5	6	14

```
arrange(X,desc(var1))
```

	var1	var2	var3
1	5	6	14
2	4	9	13
3	3	NA	12
4	2	NA	15

Adding rows and columns

```
X$var4 <- rnorm(5)  
X
```

	var1	var2	var3	var4
1	2	NA	15	0.18760
4	1	10	11	1.78698
2	3	NA	12	0.49669
3	5	6	14	0.06318
5	4	9	13	-0.53613

Adding rows and columns

```
Y <- cbind(X,rnorm(5))  
Y
```

```
var1 var2 var3      var4 rnorm(5)  
1    2   NA   15  0.18760  0.62578  
4    1   10   11  1.78698 -2.45084  
2    3   NA   12  0.49669  0.08909  
3    5     6   14  0.06318  0.47839  
5    4     9   13 -0.53613  1.00053
```

Notes and further resources

- R programming in the Data Science Track
- Andrew Jaffe's lecture notes http://www.biostat.jhsph.edu/~ajaffe/lec_winterR/Lecture%202.pdf



Summarizing data

Jeffrey Leek
Johns Hopkins Bloomberg School of Public Health

Example data set

The screenshot shows a web browser window for the "Restaurants" dataset on Open Baltimore. The URL is <https://data.baltimorecity.gov/Community/Restaurants/k5ry-ef3g>. The page title is "OPEN BALTIMORE". The main content is a table with 19 rows of restaurant data, each with columns for name, zipCode, neighborhood, councilDistrict, policeDistrict, and Location 1.

	name	zipCode	neighborhood	councilDistrict	policeDistrict	Location 1
1	410	21206	Frankford		2	NORTHEASTERN 4509 BELAIR ROAD
2	1919	21231	Fells Point		1	SOUTHEASTERN 1919 FLEET ST
3	SAUTE	21224	Canton		1	SOUTHEASTERN 2844 HUDSON ST
4	#1 CHINESE KITCHEN	21211	Hampden		14	NORTHERN 3998 ROLAND AVE
5	#1 chinese restaurant	21223	Millhill		9	SOUTHWESTERN 2481 frederick ave
6	19TH HOLE	21218	Clifton Park		14	NORTHEASTERN 2722 HARFORD RD
7	3 KINGS	21205	McElderry Park		13	SOUTHEASTERN 2510 MCLEDDERRY ST
8	3 MILES HOUSE, INC.	21211	Remington		7	NORTHERN 2701 MILES AVE
9	3 WS TAVERN	21205	McElderry Park		13	SOUTHEASTERN 2518 MONUMENT ST
10	300 SOUTH ANN STREET	21231	Upper Fells Point		1	SOUTHEASTERN 300 ANN ST
11	438 CLUB	21226	Curtis Bay		10	SOUTHERN 1600 HAZEL ST
12	5-MILE HOUSE	21215	Woodmere		5	NORTHWESTERN 5302 REISTERSTOWN RD
13	743 S. MONTFORD,INC.	21224	Canton		1	SOUTHEASTERN 743 MONTFORD ST
14	A & W RESTAURANT	21224	Pulaski Industrial Area		1	SOUTHEASTERN 5625 O DONNELL ST
15	A TASTE OF CHINA	21202	Downtown		11	CENTRAL 219 BALTIMORE ST
16	ABACROMBIE FINE FOODS	21201	Mid-Town Belvedere		11	CENTRAL 58 BIDDLE STREET
17	ABC SUSHI	21205	Middle East		13	EASTERN 2003 MONUMENT ST
18	ACROPOLIS RESTAURANT	21224	Greektown		2	SOUTHEASTERN 4718 EASTERN AVE
19	ADMIRAL FELL INN	21231	Fells Point		1	SOUTHEASTERN 818 BROADWAY

<https://data.baltimorecity.gov/Community/Restaurants/k5ry-ef3g>

Getting the data from the web

```
if(!file.exists("./data")){dir.create("./data")}  
fileUrl <- "https://data.baltimorecity.gov/api/views/k5ry-ef3g/rows.csv?accessType=DOWNLOAD"  
download.file(fileUrl, destfile="./data/restaurants.csv", method="curl")  
restData <- read.csv("./data/restaurants.csv")
```

Look at a bit of the data

```
head(restData,n=3)
```

		name	zipCode	neighborhood	councilDistrict	policeDistrict	Location.1
1	410	21206	Frankford		2	NORTHEASTERN	4509 BELAIR ROAD\nBaltimore, MD\n
2	1919	21231	Fells Point		1	SOUTHEASTERN	1919 FLEET ST\nBaltimore, MD\n
3	SAUTE	21224	Canton		1	SOUTHEASTERN	2844 HUDSON ST\nBaltimore, MD\n

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

		name	zipCode	neighborhood	councilDistrict	policeDistrict	Location.1
1325	ZINK'S CAF\u00090	21213	Belair-Edison		13	NORTHEASTERN	
1326	ZISSLIMOS BAR	21211	Hampden		7	NORTHERN	
1327	ZORBAS	21224	Greektown		2	SOUTHEASTERN	

		name	zipCode	neighborhood	councilDistrict	policeDistrict	Location.1
1325	3300 LAWNVIEW AVE	21213	Belair-Edison		13	NORTHEASTERN	3300 LAWNVIEW AVE\nBaltimore, MD\n
1326	1023 36TH ST	21211	Hampden		7	NORTHERN	1023 36TH ST\nBaltimore, MD\n
1327	4710 EASTERN Ave	21224	Greektown		2	SOUTHEASTERN	4710 EASTERN Ave\nBaltimore, MD\n

Make summary

```
summary(restData)
```

	name	zipCode	neighborhood	councilDistrict
MCDONALD'S	: 8	Min. : -21226	Downtown : 128	Min. : 1.00
POPEYES FAMOUS FRIED CHICKEN:	: 7	1st Qu.: 21202	Fells Point : 91	1st Qu.: 2.00
SUBWAY	: 6	Median : 21218	Inner Harbor: 89	Median : 9.00
KENTUCKY FRIED CHICKEN	: 5	Mean : 21185	Canton : 81	Mean : 7.19
BURGER KING	: 4	3rd Qu.: 21226	Federal Hill: 42	3rd Qu.: 11.00
DUNKIN DONUTS	: 4	Max. : 21287	Mount Vernon: 33	Max. : 14.00
(Other)	: 1293		(Other) : 863	
	policeDistrict	Location.1		
SOUTHEASTERN: 385	1101 RUSSELL ST\nBaltimore, MD\n: 9			
CENTRAL : 288	201 PRATT ST\nBaltimore, MD\n: 8			
SOUTHERN : 213	2400 BOSTON ST\nBaltimore, MD\n: 8			
NORTHERN : 157	300 LIGHT ST\nBaltimore, MD\n: 5			
NORTHEASTERN: 72	300 CHARLES ST\nBaltimore, MD\n: 4			
EASTERN : 67	301 LIGHT ST\nBaltimore, MD\n: 4			
(Other) : 145	(Other) : 1289			

Mpre in depth information

```
str(restData)
```

```
'data.frame': 1327 obs. of 6 variables:  
 $ name          : Factor w/ 1277 levels "#1 CHINESE KITCHEN",...: 9 3 992 1 2 4 5 6 7 8 ...  
 $ zipCode       : int 21206 21231 21224 21211 21223 21218 21205 21211 21205 21231 ...  
 $ neighborhood  : Factor w/ 173 levels "Abell","Arlington",...: 53 52 18 66 104 33 98 133 98 157 ...  
 $ councilDistrict: int 2 1 1 14 9 14 13 7 13 1 ...  
 $ policeDistrict : Factor w/ 9 levels "CENTRAL","EASTERN",...: 3 6 6 4 8 3 6 4 6 6 ...  
 $ Location.1     : Factor w/ 1210 levels "1 BIDDLE ST\nBaltimore, MD\n",...: 835 334 554 755 492 537 5
```

Quantiles of quantitative variables

```
quantile(restData$councilDistrict,na.rm=TRUE)
```

```
0% 25% 50% 75% 100%
1    2    9   11   14
```

```
quantile(restData$councilDistrict,probs=c(0.5,0.75,0.9))
```

```
50% 75% 90%
9   11   12
```

Make table

```
table(restData$zipCode,useNA="ifany")
```

-21226	21201	21202	21205	21206	21207	21208	21209	21210	21211	21212	21213	21214	21215
1	136	201	27	30	4	1	8	23	41	28	31	17	54
21216	21217	21218	21220	21222	21223	21224	21225	21226	21227	21229	21230	21231	21234
10	32	69	1	7	56	199	19	18	4	13	156	127	7
21237	21239	21251	21287										
1	3	2	1										

Make table

```
table(restData$councilDistrict,restData$zipCode)
```

	-21226	21201	21202	21205	21206	21207	21208	21209	21210	21211	21212	21213	21214	21215	21216
1	0	0	37	0	0	0	0	0	0	0	0	2	0	0	0
2	0	0	0	3	27	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	2	17	0	0
4	0	0	0	0	0	0	0	0	0	0	27	0	0	0	0
5	0	0	0	0	0	3	0	6	0	0	0	0	0	31	0
6	0	0	0	0	0	0	0	1	19	0	0	0	0	15	1
7	0	0	0	0	0	0	0	1	0	27	0	0	0	6	7
8	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
9	0	1	0	0	0	0	0	0	0	0	0	0	0	0	2
10	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
11	0	115	139	0	0	0	1	0	0	0	1	0	0	0	0
12	0	20	24	4	0	0	0	0	0	0	0	13	0	0	0
13	0	0	0	20	3	0	0	0	0	0	0	13	0	1	0
14	0	0	0	0	0	0	0	0	4	14	0	1	0	1	0

Check for missing values

```
sum(is.na(restData$councilDistrict))
```

```
[1] 0
```

```
any(is.na(restData$councilDistrict))
```

```
[1] FALSE
```

```
all(restData$zipCode > 0)
```

```
[1] FALSE
```

Row and column sums

```
colSums(is.na(restData))
```

name	zipCode	neighborhood	councilDistrict	policeDistrict	Location.1
0	0	0	0	0	0

```
all(colSums(is.na(restData))==0)
```

```
[1] TRUE
```

Values with specific characteristics

```
table(restData$zipCode %in% c("21212"))
```

```
FALSE  TRUE  
1299    28
```

```
table(restData$zipCode %in% c("21212", "21213"))
```

```
FALSE  TRUE  
1268    59
```

Values with specific characteristics

```
restData[restData$zipCode %in% c("21212","21213"),]
```

		name	zipCode	neighborhood	councilDistrict
29		BAY ATLANTIC CLUB	21212	Downtown	11
39		BERMUDA BAR	21213	Broadway East	12
92		ATWATER'S	21212	Chinquapin Park-Belvedere	4
111		BALTIMORE ESTONIAN SOCIETY	21213	South Clifton Park	12
187		CAFE ZEN	21212	Rosebank	4
220		CERIELLO FINE FOODS	21212	Chinquapin Park-Belvedere	4
266		CLIFTON PARK GOLF COURSE SNACK BAR	21213	Darley Park	14
276		CLUB HOUSE BAR & GRILL	21213	Orangeville Industrial Area	13
289		CLUBHOUSE BAR & GRILL	21213	Orangeville Industrial Area	13
291		COCKY LOU'S	21213	Broadway East	12
362		DREAM TAVERN, CARRIBEAN U.S.A.	21213	Broadway East	13
373		DUNKIN DONUTS	21212	Homeland	4
383		EASTSIDE SPORTS SOCIAL CLUB	21213	Broadway East	13
417		FIELDS OLD TRAIL	21212	Mid-Govans	4
475		GRAND CRU	21212	Chinquapin Park-Belvedere	4
545		RANDY'S BAR	21213	Broadway East	12
604		MURPHY'S NEIGHBORHOOD BAR & GRILL	21212	Mid-Govans	4

Cross tabs

```
data(UCBAdmissions)
DF = as.data.frame(UCBAdmissions)
summary(DF)
```

Admit	Gender	Dept	Freq
Admitted:12	Male :12	A:4	Min. : 8
Rejected:12	Female:12	B:4	1st Qu.: 80
		C:4	Median :170
		D:4	Mean :189
		E:4	3rd Qu.:302
		F:4	Max. :512

Cross tabs

```
xt <- xtabs(Freq ~ Gender + Admit,data=DF)
xt
```

		Admit	
		Admitted	Rejected
Gender	Male	1198	1493
	Female	557	1278

Flat tables

```
warpbreaks$replicate <- rep(1:9, len = 54)
xt = xtabs(breaks ~ ., data=warpbreaks)
xt
```

```
, , replicate = 1
```

```
      tension
wool  L  M  H
  A 26 18 36
  B 27 42 20
```

```
, , replicate = 2
```

```
      tension
wool  L  M  H
  A 30 21 21
  B 14 26 21
```

```
, , replicate = 3
```

Flat tables

```
ftable(xt)
```

		replicate	1	2	3	4	5	6	7	8	9
		wool	tension								
A	L		26	30	54	25	70	52	51	26	67
	M		18	21	29	17	12	18	35	30	36
	H		36	21	24	18	10	43	28	15	26
B	L		27	14	29	19	29	31	41	20	44
	M		42	26	19	16	39	28	21	39	29
	H		20	21	24	17	13	15	15	16	28

Size of a data set

```
fakeData = rnorm(1e5)  
object.size(fakeData)
```

800040 bytes

```
print(object.size(fakeData),units="Mb")
```

0.8 Mb



Creating new variables

Jeffrey Leek
Johns Hopkins Bloomberg School of Public Health

Why create new variables?

- Often the raw data won't have a value you are looking for
- You will need to transform the data to get the values you would like
- Usually you will add those values to the data frames you are working with
- Common variables to create
 - Missingness indicators
 - "Cutting up" quantitative variables
 - Applying transforms

Example data set

The screenshot shows a web browser window for the "Restaurants" dataset on Open Baltimore. The URL is <https://data.baltimorecity.gov/Community/Restaurants/k5ry-ef3g>. The page title is "OPEN BALTIMORE". The main content is a table with 19 rows of restaurant data, each with a unique ID (1-19) and columns for name, zipCode, neighborhood, councilDistrict, policeDistrict, and Location 1.

	name	zipCode	neighborhood	councilDistrict	policeDistrict	Location 1
1	410	21206	Frankford		2	NORTHEASTERN 4509 BELAIR ROAD
2	1919	21231	Fells Point		1	SOUTHEASTERN 1919 FLEET ST
3	SAUTE	21224	Canton		1	SOUTHEASTERN 2844 HUDSON ST
4	#1 CHINESE KITCHEN	21211	Hampden		14	NORTHERN 3998 ROLAND AVE
5	#1 chinese restaurant	21223	Millhill		9	SOUTHWESTERN 2481 frederick ave
6	19TH HOLE	21218	Clifton Park		14	NORTHEASTERN 2722 HARFORD RD
7	3 KINGS	21205	McElderry Park		13	SOUTHEASTERN 2510 MCLEDDERRY ST
8	3 MILES HOUSE, INC.	21211	Remington		7	NORTHERN 2701 MILES AVE
9	3 WS TAVERN	21205	McElderry Park		13	SOUTHEASTERN 2518 MONUMENT ST
10	300 SOUTH ANN STREET	21231	Upper Fells Point		1	SOUTHEASTERN 300 ANN ST
11	438 CLUB	21226	Curtis Bay		10	SOUTHERN 1600 HAZEL ST
12	5-MILE HOUSE	21215	Woodmere		5	NORTHWESTERN 5302 REISTERSTOWN RD
13	743 S. MONTFORD,INC.	21224	Canton		1	SOUTHEASTERN 743 MONTFORD ST
14	A & W RESTAURANT	21224	Pulaski Industrial Area		1	SOUTHEASTERN 5625 O DONNELL ST
15	A TASTE OF CHINA	21202	Downtown		11	CENTRAL 219 BALTIMORE ST
16	ABACROMBIE FINE FOODS	21201	Mid-Town Belvedere		11	CENTRAL 58 BIDDLE STREET
17	ABC SUSHI	21205	Middle East		13	EASTERN 2003 MONUMENT ST
18	ACROPOLIS RESTAURANT	21224	Greektown		2	SOUTHEASTERN 4718 EASTERN AVE
19	ADMIRAL FELL INN	21231	Fells Point		1	SOUTHEASTERN 818 BROADWAY

<https://data.baltimorecity.gov/Community/Restaurants/k5ry-ef3g>

Getting the data from the web

```
if(!file.exists("./data")){dir.create("./data")}  
fileUrl <- "https://data.baltimorecity.gov/api/views/k5ry-ef3g/rows.csv?accessType=DOWNLOAD"  
download.file(fileUrl,destfile="./data/restaurants.csv",method="curl")  
restData <- read.csv("./data/restaurants.csv")
```

Creating sequences

Sometimes you need an index for your data set

```
s1 <- seq(1,10,by=2) ; s1
```

```
[1] 1 3 5 7 9
```

```
s2 <- seq(1,10,length=3); s2
```

```
[1] 1.0 5.5 10.0
```

```
x <- c(1,3,8,25,100); seq(along = x)
```

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

Subsetting variables

```
restData$nearMe = restData$neighborhood %in% c("Roland Park", "Homeland")
table(restData$nearMe)
```

```
FALSE  TRUE
1314    13
```

Creating binary variables

```
restData$zipWrong = ifelse(restData$zipCode < 0, TRUE, FALSE)  
table(restData$zipWrong,restData$zipCode < 0)
```

	FALSE	TRUE
FALSE	1326	0
TRUE	0	1

Creating categorical variables

```
restData$zipGroups = cut(restData$zipCode, breaks=quantile(restData$zipCode))  
table(restData$zipGroups)
```

(-2.123e+04, 2.12e+04]	(2.12e+04, 2.122e+04]	(2.122e+04, 2.123e+04]	(2.123e+04, 2.129e+04]
337	375	282	332

```
table(restData$zipGroups, restData$zipCode)
```

	-21226	21201	21202	21205	21206	21207	21208	21209	21210	21211	21212	21213
(-2.123e+04, 2.12e+04]	0	136	201	0	0	0	0	0	0	0	0	0
(2.12e+04, 2.122e+04]	0	0	0	27	30	4	1	8	23	41	28	31
(2.122e+04, 2.123e+04]	0	0	0	0	0	0	0	0	0	0	0	0
(2.123e+04, 2.129e+04]	0	0	0	0	0	0	0	0	0	0	0	0

Easier cutting

```
library(Hmisc)
restData$zipGroups = cut2(restData$zipCode, g=4)
table(restData$zipGroups)
```

```
[-21226,21205) [ 21205,21220) [ 21220,21227) [ 21227,21287]
 338           375           300           314
```

Creating factor variables

```
restData$zcf <- factor(restData$zipCode)  
restData$zcf[1:10]
```

```
[1] 21206 21231 21224 21211 21223 21218 21205 21211 21205 21231  
32 Levels: -21226 21201 21202 21205 21206 21207 21208 21209 21210 21211 ... 21287
```

```
class(restData$zcf)
```

```
[1] "factor"
```

Levels of factor variables

```
yesno <- sample(c("yes", "no"), size=10, replace=TRUE)
yesnofac = factor(yesno, levels=c("yes", "no"))
relevel(yesnofac, ref="yes")
```

```
[1] yes yes yes yes no  yes yes yes no  no
Levels: yes no
```

```
as.numeric(yesnofac)
```

```
[1] 1 1 1 1 2 1 1 1 2 2
```

Cutting produces factor variables

```
library(Hmisc)
restData$zipGroups = cut2(restData$zipCode,g=4)
table(restData$zipGroups)
```

```
[-21226,21205) [ 21205,21220) [ 21220,21227) [ 21227,21287]
338           375           300           314
```

Using the mutate function

```
library(Hmisc); library(plyr)  
restData2 = mutate(restData,zipGroups=cut2(zipCode,g=4))  
table(restData2$zipGroups)
```

```
[-21226,21205) [ 21205,21220) [ 21220,21227) [ 21227,21287]  
338           375           300           314
```

Common transforms

- `abs(x)` absolute value
- `sqrt(x)` square root
- `ceiling(x)` ceiling(3.475) is 4
- `floor(x)` floor(3.475) is 3
- `round(x,digits=n)` round(3.475,digits=2) is 3.48
- `signif(x,digits=n)` signif(3.475,digits=2) is 3.5
- `cos(x), sin(x)` etc.
- `log(x)` natural logarithm
- `log2(x), log10(x)` other common logs
- `exp(x)` exponentiating x

http://www.biostat.jhsph.edu/~ajaffe/lec_winterR/Lecture%202.pdf

<http://statmethods.net/management/functions.html>

Notes and further reading

- A tutorial from the developer of plyr - <http://plyr.had.co.nz/09-user/>
- Andrew Jaffe's R notes http://www.biostat.jhsph.edu/~ajaffe/lec_winterR/Lecture%202.pdf
-



Reshaping data

Jeffrey Leek
Johns Hopkins Bloomberg School of Public Health

The goal is tidy data

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
i	problem_id	subject_id	start	stop	time_left	answr										
2	1	498	17	130719989	1307120010	2359										
3	1	150	16	130719991	1307120009	2376	O									
4	3	313	16	130719994	1307120009	2376	E									
5	4	12	16	130719995	1307120009	2376	E									
6	5	273	14	130719996	1307120020	2357	A									
7	6	101	19	130719996	1307120021	2364	B									
8	7	103	18	130719998	1307120028	2372	B									
9	8	162	12	1307120004	1307120042	2343	C									
10	9	70	13	1307120005	1307120049	2351	C									
11	10	300	16	1307120012	1307120094	2293	B									
12	11	494	17	1307120017	1307120094	2310	D									
13	12	297	13	1307120018	1307120118	2269	A									
14	13	522	19	1307120025	1307120152	2233	D									
15	14	233	15	1307120041	1307120157	2246	C									
16	15	344	15	1307120041	1307120117	2268	B									
17	16	160	17	1307120079	1307120249	2136	D									
18	17	253	16	1307120080	1307120249	2136	B									
19	18	472	12	1307120119	1307120170	2215	A									
20	19	45	15	1307120144	1307120144	2186	C									
21	20	353	13	1307120144	1307120199	2186	C									
22	21	210	15	1307120145	1307120199	2113	B									
23	22	69	18	1307120183	1307120188	2072	D									
24	23	562	16	1307120190	1307120301	2084	O									
25	24	111	19	1307120191	1307120301	2084	O									
26	25	297	15	1307120277	1307120342	2043	B									
27	26	493	14	1307120288	1307120343	2042	C									
28	27	94	14	1307120288	1307120343	2042	E									
29	28	22	18	1307120310	1307120363	2020	C									
30	29	64	15	1307120310	1307120363	2020	B									
31	30	502	16	1307120323	1307120338	2049	B									
32	31	44	16	1307120323	1307120341	2049	A									
33	32	315	14	1307120348	1307120362	2023	B									
34	33	385	15	1307120352	1307120553	1832	C									
35	34	555	13	1307120352	1307120561	1841	B									
36	35	92	14	1307120368	1307120397	1988	B									
37	36	295	15	1307120368	1307120397	1988	D									
38	37	267	17	1307120382	1307120513	1870	E									
39	38	297	14	1307120401	1307120427	1958	C									
40	39	312	19	1307120401	1307120427	1958	C									
41	40	321	18	1307120431	1307120449	1936	A									
42	41	770	16	1307120431	1307120456	1974	C									

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



Merging data

Jeffrey Leek
Johns Hopkins Bloomberg School of Public Health

Peer review experiment data

The screenshot shows a PLOS ONE article page. At the top, there is a banner with the text "Simplify your research with automatic and continuous dosing" and an image of medical syringes. Below the banner, the PLOS ONE logo is on the left, and navigation links for "Articles", "For Authors", "About Us", and "Search" are on the right. A sign-in button is also visible. The main article information includes "OPEN ACCESS" and "PEER-REVIEWED" status, the title "Cooperation between Referees and Authors Increases Peer Review Accuracy", and authors "Jeffrey T. Leek" and "Margaret A. Taub, Fernando J. Pineda". To the right, metrics are displayed: 6,497 views, 2 citations, 61 academic bookmarks, and 108 social shares. Below the title, there are tabs for "Article", "About the Authors", "Metrics", "Comments", and "Related Content". The "Article" tab is active. The "Metrics" section contains four small figures: a flowchart of the review process, three line graphs showing review scores over time, and a network graph of referee interactions. To the right of the article content are buttons for "Download", "Print", and "Share". A "Comments" section is present at the bottom right.

<http://www.plosone.org/article/info:doi/10.1371/journal.pone.0026895>

Peer review data

```
if(!file.exists("./data")){dir.create("./data")}

fileUrl1 = "https://dl.dropboxusercontent.com/u/7710864/data/reviews-apr29.csv"
fileUrl2 = "https://dl.dropboxusercontent.com/u/7710864/data/solutions-apr29.csv"
download.file(fileUrl1,destfile="./data/reviews.csv",method="curl")
download.file(fileUrl2,destfile="./data/solutions.csv",method="curl")
reviews = read.csv("./data/reviews.csv"); solutions <- read.csv("./data/solutions.csv")
head(reviews,2)
```

	<code>id</code>	<code>solution_id</code>	<code>reviewer_id</code>	<code>start</code>	<code>stop</code>	<code>time_left</code>	<code>accept</code>
1	1	3	27	1304095698	1304095758	1754	1
2	2	4	22	1304095188	1304095206	2306	1

```
head(solutions,2)
```

	<code>id</code>	<code>problem_id</code>	<code>subject_id</code>	<code>start</code>	<code>stop</code>	<code>time_left</code>	<code>answer</code>
1	1	156	29	1304095119	1304095169	2343	B
2	2	269	25	1304095119	1304095183	2329	C

Merging data - merge()

- Merges data frames
- Important parameters: *x,y,by,by.x,by.y,all*

```
names(reviews)
```

```
[1] "id"           "solution_id" "reviewer_id" "start"      "stop"       "time_left"  
[7] "accept"
```

```
names(solutions)
```

```
[1] "id"           "problem_id"  "subject_id" "start"      "stop"       "time_left"  "answer"
```

Merging data - merge()

```
mergedData = merge(reviews,solutions,by.x="solution_id",by.y="id",all=TRUE)  
head(mergedData)
```

	solution_id	id	reviewer_id	start.x	stop.x	time_left.x	accept	problem_id	subject_id
1	1	4		26	1304095267	1304095423	2089	1	156
2	2	6		29	1304095471	1304095513	1999	1	269
3	3	1		27	1304095698	1304095758	1754	1	34
4	4	2		22	1304095188	1304095206	2306	1	19
5	5	3		28	1304095276	1304095320	2192	1	605
6	6	16		22	1304095303	1304095471	2041	1	384
	start.y	stop.y	time_left.y	answer					
1	1304095119	1304095169		2343	B				
2	1304095119	1304095183		2329	C				
3	1304095127	1304095146		2366	C				
4	1304095127	1304095150		2362	D				
5	1304095127	1304095167		2345	A				
6	1304095131	1304095270		2242	C				

Default - merge all common column names

```
intersect(names(solutions),names(reviews))
```

```
[1] "id"       "start"     "stop"      "time_left"
```

```
mergedData2 = merge(reviews,solutions,all=TRUE)
head(mergedData2)
```

	id	start	stop	time_left	solution_id	reviewer_id	accept	problem_id	subject_id	answer
1	1	1304095119	1304095169	2343	NA	NA	NA	156	29	B
2	1	1304095698	1304095758	1754	3	27	1	NA	NA	<NA>
3	2	1304095119	1304095183	2329	NA	NA	NA	269	25	C
4	2	1304095188	1304095206	2306	4	22	1	NA	NA	<NA>
5	3	1304095127	1304095146	2366	NA	NA	NA	34	22	C
6	3	1304095276	1304095320	2192	5	28	1	NA	NA	<NA>

Using join in the plyr package

Faster, but less full featured - defaults to left join, see help file for more

```
df1 = data.frame(id=sample(1:10),x=rnorm(10))
df2 = data.frame(id=sample(1:10),y=rnorm(10))
arrange(join(df1,df2),id)
```

	<code>id</code>	<code>x</code>	<code>y</code>
1	1	0.2514	0.2286
2	2	0.1048	0.8395
3	3	-0.1230	-1.1165
4	4	1.5057	-0.1121
5	5	-0.2505	1.2124
6	6	0.4699	-1.6038
7	7	0.4627	-0.8060
8	8	-1.2629	-1.2848
9	9	-0.9258	-0.8276
10	10	2.8065	0.5794

If you have multiple data frames

```
df1 = data.frame(id=sample(1:10),x=rnorm(10))
df2 = data.frame(id=sample(1:10),y=rnorm(10))
df3 = data.frame(id=sample(1:10),z=rnorm(10))
dfList = list(df1,df2,df3)
join_all(dfList)
```

	id	x	y	z
1	6	0.39093	-0.16670	0.56523
2	1	-1.90467	0.43811	-0.37449
3	7	-1.48798	-0.85497	-0.69209
4	10	-2.59440	0.39591	-0.36134
5	3	-0.08539	0.08053	1.01247
6	4	-1.63165	-0.13158	0.21927
7	5	-0.50594	0.24256	-0.44003
8	9	-0.85062	-2.08066	-0.96950
9	2	-0.63767	-0.10069	0.09002
10	8	1.20439	1.29138	-0.88586

More on merging data

- The quick R data merging page - <http://www.statmethods.net/management/merging.html>
- plyr information - <http://plyr.had.co.nz/>
- Types of joins - [http://en.wikipedia.org/wiki/Join_\(SQL\)](http://en.wikipedia.org/wiki/Join_(SQL))