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
```

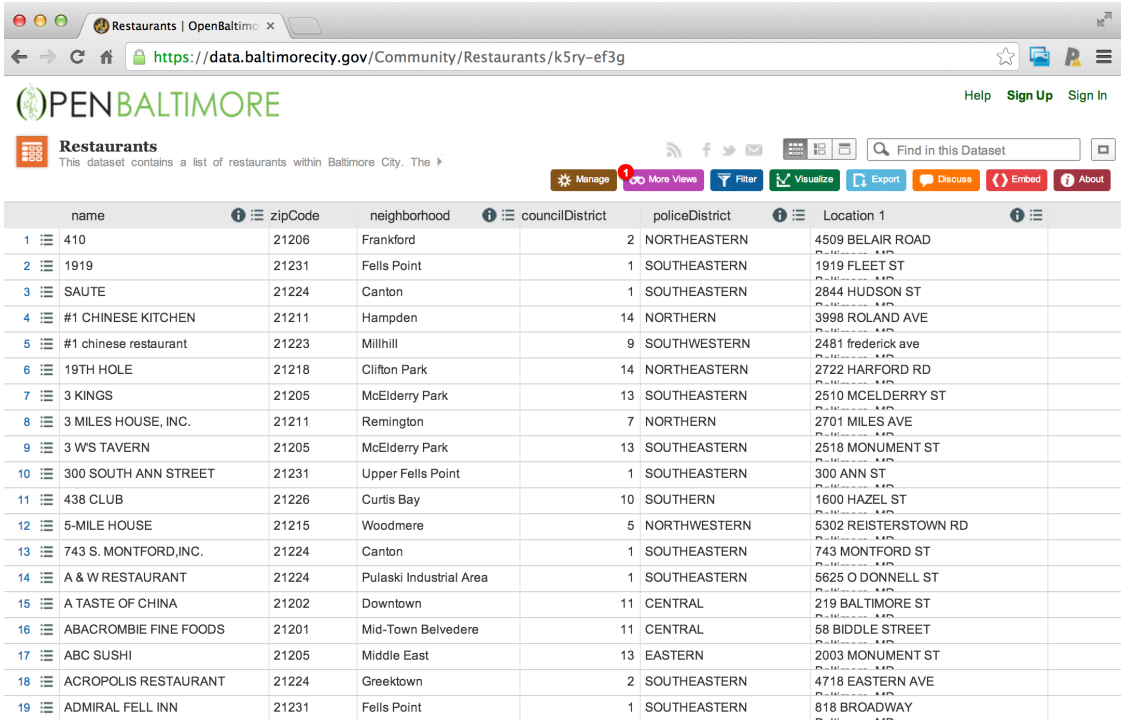
Notes and further resources

- 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



	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 MCELDERRY ST
8	3 MILES HOUSE, INC.	21211	Remington	7	NORTHERN	2701 MILES AVE
9	3 W'S 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
1325 ZINK'S CAF\u0090   21213 Belair-Edison              13   NORTHEASTERN
1326     ZISSIMOS BAR   21211       Hampden               7       NORTHERN
1327           ZORBAS   21224     Greektown               2   SOUTHEASTERN
                           Location.1
1325 3300 LAWNVIEW AVE\nBaltimore, MD\n
1326      1023 36TH ST\nBaltimore, MD\n
1327  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

   21217 21218 21220 21222 21223 21224 21225 21226 21227 21229 21230 21231 21234 21237 21239
```

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
Gender   Admitted Rejected
  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
```

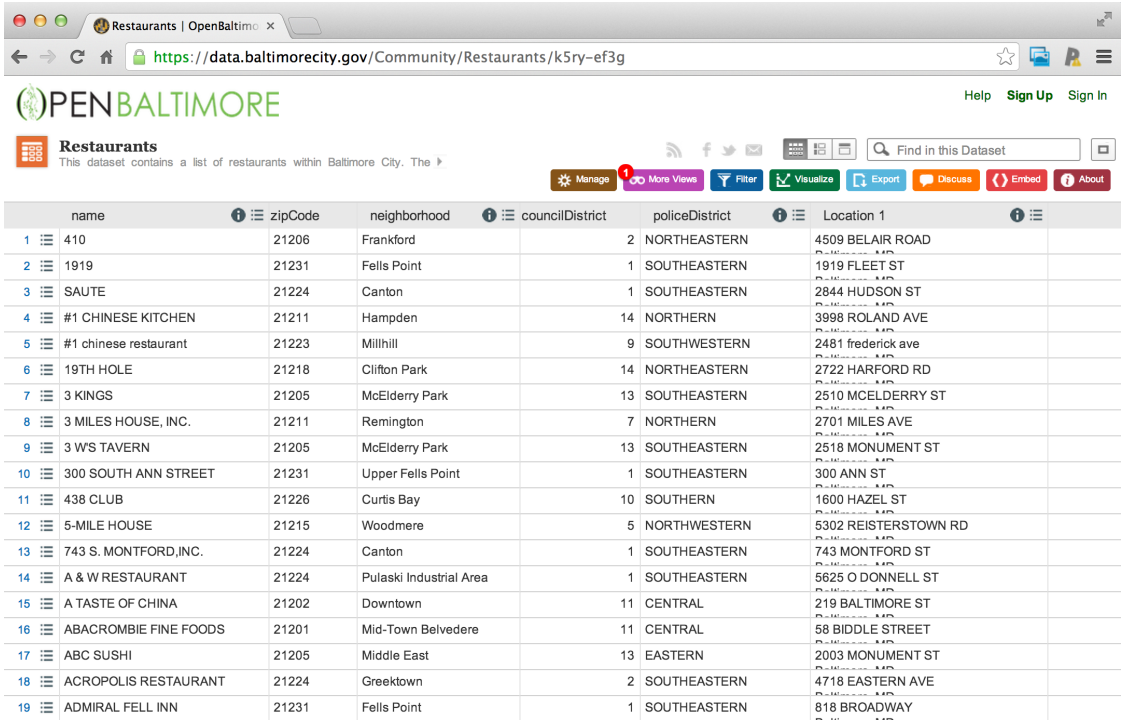
Creating new variables

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



	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 MCELDERRY ST
8	3 MILES HOUSE, INC.	21211	Remington	7	NORTHERN	2701 MILES AVE
9	3 W'S 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

                     21214 21215 21216 21217 21218 21220 21222 21223 21224 21225 21226 21227
```

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 21212 21213 21214 ... 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)` roun(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

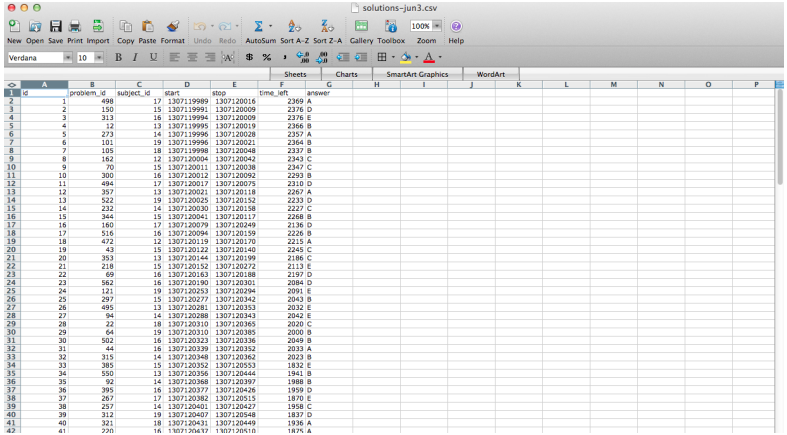
Notes and further reading

- 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



id	problem_id	subject_id	start	stop	time_left	answer
1	498	17	1307119989	1307120016	2369	A
2	150	15	1307119991	1307120009	2376	D
3	313	16	1307119994	1307120009	2376	E
4	12	13	1307119995	1307120019	2366	B
5	273	14	1307119996	1307120028	2357	A
6	181	19	1307119996	1307120021	2364	B
7	105	18	1307119998	1307120048	2337	B
8	162	12	1307120004	1307120042	2343	C
9	70	15	1307120011	1307120038	2347	C
10	300	16	1307120012	1307120092	2293	B
11	494	17	1307120017	1307120075	2310	D
12	357	13	1307120021	1307120118	2267	A
13	522	19	1307120025	1307120152	2233	D
14	232	14	1307120030	1307120158	2227	C
15	344	15	1307120041	1307120117	2268	B
16	160	17	1307120079	1307120249	2136	D
17	516	16	1307120094	1307120159	2226	B
18	472	12	1307120119	1307120170	2215	A
19	43	15	1307120122	1307120140	2245	C
20	353	13	1307120144	1307120199	2186	C
21	218	15	1307120152	1307120272	2113	E
22	69	16	1307120163	1307120188	2197	D
23	562	16	1307120190	1307120301	2084	D
24	121	19	1307120253	1307120294	2091	E
25	297	15	1307120277	1307120342	2043	B
26	495	13	1307120281	1307120353	2032	E
27	94	14	1307120288	1307120343	2042	E
28	22	18	1307120310	1307120365	2020	C
29	64	19	1307120310	1307120385	2000	B
30	502	16	1307120323	1307120336	2049	B
31	44	16	1307120339	1307120352	2033	A
32	315	14	1307120348	1307120362	2023	B
33	385	15	1307120352	1307120553	1832	E
34	550	13	1307120356	1307120444	1941	B
35	92	14	1307120368	1307120397	1988	B
36	395	16	1307120377	1307120426	1959	D
37	267	17	1307120382	1307120515	1870	E
38	257	14	1307120401	1307120427	1958	C
39	312	19	1307120407	1307120548	1837	D
40	321	18	1307120431	1307120449	1936	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
```


http://www.statmethods.net/management/reshape.html

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
```


http://www.statmethods.net/management/reshape.html

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
```

http://www.r-bloggers.com/a-quick-primer-on-split-apply-combine-problems/

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

Managing Data Frames with `dplyr`

December 30, 2014

dplyr

The data frame is a key data structure in statistics and in R.

- There is one observation per row
- Each column represents a variable or measure or characteristic
- Primary implementation that you will use is the default R implementation
- Other implementations, particularly relational databases systems

dplyr

- Developed by Hadley Wickham of RStudio
- An optimized and distilled version of `plyr` package (also by Hadley)
- Does not provide any "new" functionality per se, but **greatly** simplifies existing functionality in R
- Provides a "grammar" (in particular, verbs) for data manipulation
- Is **very** fast, as many key operations are coded in C++

dplyr Verbs

- `select`: return a subset of the columns of a data frame
- `filter`: extract a subset of rows from a data frame based on logical conditions
- `arrange`: reorder rows of a data frame
- `rename`: rename variables in a data frame
- `mutate`: add new variables/columns or transform existing variables
- `summarise` / `summarize`: generate summary statistics of different variables in the data frame, possibly within strata

There is also a handy `print` method that prevents you from printing a lot of data to the console.

dplyr Properties

- The first argument is a data frame.
- The subsequent arguments describe what to do with it, and you can refer to columns in the data frame directly without using the $ operator (just use the names).
- The result is a new data frame
- Data frames must be properly formatted and annotated for this to all be useful

Load the dplyr package

This step is important!

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
##
## The following object is masked from 'package:stats':
##
##     filter
##
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

select

```
chicago <- readRDS("chicago.rds")
dim(chicago)
```

```
## [1] 6940    8
```

```
head(select(chicago, 1:5))
```

```
##    city tmpd   dptp       date pm25tmean2
## 1 chic 31.5 31.500 1987-01-01          NA
## 2 chic 33.0 29.875 1987-01-02          NA
## 3 chic 33.0 27.375 1987-01-03          NA
## 4 chic 29.0 28.625 1987-01-04          NA
## 5 chic 32.0 28.875 1987-01-05          NA
## 6 chic 40.0 35.125 1987-01-06          NA
```

select

```
names(chicago)[1:3]
```

```
## [1] "city" "tmpd" "dptp"
```

```
head(select(chicago, city:dptp))
```

```
##   city tmpd   dptp
## 1 chic 31.5 31.500
## 2 chic 33.0 29.875
## 3 chic 33.0 27.375
## 4 chic 29.0 28.625
## 5 chic 32.0 28.875
## 6 chic 40.0 35.125
```

select

In dplyr you can do

```
head(select(chicago, -(city:dptp)))
```

Equivalent base R

```
i <- match("city", names(chicago))
j <- match("dptp", names(chicago))
head(chicago[, -(i:j)])
```

filter

```
chic.f <- filter(chicago, pm25tmean2 > 30)
head(select(chic.f, 1:3, pm25tmean2), 10)
```

```
##     city tmpd dptp pm25tmean2
## 1   chic   23 21.9      38.10
## 2   chic   28 25.8      33.95
## 3   chic   55 51.3      39.40
## 4   chic   59 53.7      35.40
## 5   chic   57 52.0      33.30
## 6   chic   57 56.0      32.10
## 7   chic   75 65.8      56.50
## 8   chic   61 59.0      33.80
## 9   chic   73 60.3      30.30
## 10 chic   78 67.1      41.40
```

filter

```
chic.f <- filter(chicago, pm25tmean2 > 30 & tmpd > 80)
head(select(chic.f, 1:3, pm25tmean2, tmpd), 10)
```

```
##     city tmpd dptp pm25tmean2
## 1   chic   81 71.2    39.6000
## 2   chic   81 70.4    31.5000
## 3   chic   82 72.2    32.3000
## 4   chic   84 72.9    43.7000
## 5   chic   85 72.6    38.8375
## 6   chic   84 72.6    38.2000
## 7   chic   82 67.4    33.0000
## 8   chic   82 63.5    42.5000
## 9   chic   81 70.4    33.1000
## 10 chic   82 66.2    38.8500
```

arrange

Reordering rows of a data frame (while preserving corresponding order of other columns) is normally a pain to do in R.

```
chicago <- arrange(chicago, date)
head(select(chicago, date, pm25tmean2), 3)
```

```
##         date pm25tmean2
## 1 1987-01-01          NA
## 2 1987-01-02          NA
## 3 1987-01-03          NA
```

```
tail(select(chicago, date, pm25tmean2), 3)
```

```
##            date pm25tmean2
## 6938 2005-12-29    7.45000
## 6939 2005-12-30   15.05714
## 6940 2005-12-31   15.00000
```

arrange

Columns can be arranged in descending order too.

```
chicago <- arrange(chicago, desc(date))
head(select(chicago, date, pm25tmean2), 3)
```

```
##         date pm25tmean2
## 1 2005-12-31   15.00000
## 2 2005-12-30   15.05714
## 3 2005-12-29    7.45000
```

```
tail(select(chicago, date, pm25tmean2), 3)
```

```
##             date pm25tmean2
## 6938 1987-01-03         NA
## 6939 1987-01-02         NA
## 6940 1987-01-01         NA
```

rename

Renaming a variable in a data frame in R is surprising hard to do!

```
head(chicago[, 1:5], 3)
```

```
##    city tmpd dptp       date pm25tmean2
## 1 chic   35 30.1 2005-12-31   15.00000
## 2 chic   36 31.0 2005-12-30   15.05714
## 3 chic   35 29.4 2005-12-29    7.45000
```

```
chicago <- rename(chicago, dewpoint = dptp,
                  pm25 = pm25tmean2)
head(chicago[, 1:5], 3)
```

```
##    city tmpd dewpoint       date     pm25
## 1 chic   35     30.1 2005-12-31 15.00000
## 2 chic   36     31.0 2005-12-30 15.05714
## 3 chic   35     29.4 2005-12-29  7.45000
```

mutate

```
chicago <- mutate(chicago,
                  pm25detrend=pm25-mean(pm25, na.rm=TRUE))
head(select(chicago, pm25, pm25detrend))
```

```
##       pm25 pm25detrend
## 1 15.00000   -1.230958
## 2 15.05714   -1.173815
## 3  7.45000   -8.780958
## 4 17.75000    1.519042
## 5 23.56000    7.329042
## 6  8.40000   -7.830958
```

group_by

Generating summary statistics by stratum

```
chicago <- mutate(chicago,
                  tempcat = factor(1 * (tmpd > 80),
                                   labels = c("cold", "hot"
hotcold <- group_by(chicago, tempcat)
summarize(hotcold, pm25 = mean(pm25, na.rm = TRUE),
          o3 = max(o3tmean2),
          no2 = median(no2tmean2))
```

```
## Source: local data frame [3 x 4]
##
##   tempcat     pm25        o3      no2
## 1    cold 15.97807 66.587500 24.54924
## 2     hot 26.48118 62.969656 24.93870
## 3      NA 47.73750  9.416667 37.44444
```

group_by

Generating summary statistics by stratum

```
chicago <- mutate(chicago,
                  year = as.POSIXlt(date)$year + 1900)
years <- group_by(chicago, year)
summarize(years, pm25 = mean(pm25, na.rm = TRUE),
          o3 = max(o3tmean2, na.rm = TRUE),
          no2 = median(no2tmean2, na.rm = TRUE))
```

```
## Source: local data frame [19 x 4]
##
##    year     pm25       o3      no2
## 1  1987      NaN 62.96966 23.49369
## 2  1988      NaN 61.67708 24.52296
## 3  1989      NaN 59.72727 26.14062
## 4  1990      NaN 52.22917 22.59583
## 5  1991      NaN 63.10417 21.38194
## 6  1992      NaN 50.82870 24.78921
## 7  1993      NaN 44.30093 25.76993
```

%>%

```
chicago %>% mutate(month = as.POSIXlt(date)$mon + 1)
    %>% group_by(month)
    %>% summarize(pm25 = mean(pm25, na.rm = TRUE),
          o3 = max(o3tmean2, na.rm = TRUE),
          no2 = median(no2tmean2, na.rm = TRUE))

## Source: local data frame [12 x 4]
##
##    month     pm25       o3      no2
## 1      1 17.76996 28.22222 25.35417
## 2      2 20.37513 37.37500 26.78034
## 3      3 17.40818 39.05000 26.76984
## 4      4 13.85879 47.94907 25.03125
## 5      5 14.07420 52.75000 24.22222
## 6      6 15.86461 66.58750 25.01140
## 7      7 16.57087 59.54167 22.38442
## 8      8 16.93380 53.96701 22.98333
## 9      9 15.91279 57.48864 24.47917
```

dplyr

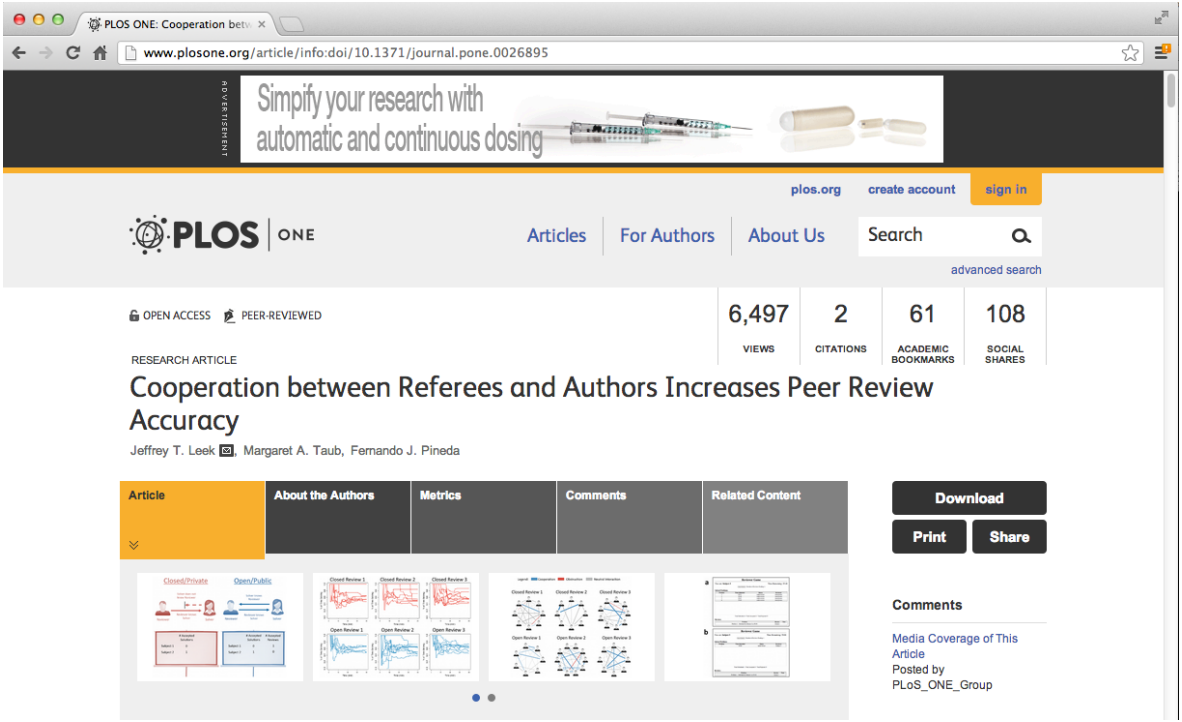
Once you learn the dplyr "grammar" there are a few additional benefits

- dplyr can work with other data frame "backends"
- `data.table` for large fast tables
- SQL interface for relational databases via the DBI package

Merging data

Jeffrey Leek
Johns Hopkins Bloomberg School of Public Health

Peer review experiment data



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)
```

```
  id solution_id reviewer_id      start       stop time_left accept
1  1           3          27 1304095698 1304095758      1754      1
2  2           4          22 1304095188 1304095206      2306      1
```

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

```
  id problem_id subject_id      start       stop time_left answer
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          29
2           2  6          29 1304095471 1304095513        1999      1        269          25
3           3  1          27 1304095698 1304095758        1754      1         34          22
4           4  2          22 1304095188 1304095206        2306      1         19          23
5           5  3          28 1304095276 1304095320        2192      1        605          26
6           6 16          22 1304095303 1304095471        2041      1        384          27
     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)
```

```
   id       x       y
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))