GBA 464 Lab 2

Kang Huang

Aug 5, 2022

Array Dimension

- Suppose I have the following array, what is its dimension?

```
> myarray
, , 1

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

, , 2

     [,1] [,2] [,3]
[1,]    7    9   11
[2,]    8   10   12

, , 3

     [,1] [,2] [,3]
[1,]   13   15   17
[2,]   14   16   18
```

```
> myarray = array(1: 18, dim = c(2, 3, 3))
> myarray
, , 1

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

, , 2

     [,1] [,2] [,3]
[1,]    7    9   11
[2,]    8   10   12

, , 3

     [,1] [,2] [,3]
[1,]   13   15   17
[2,]   14   16   18
```

- First two dimensions are row and column for the “base matrix”, and the third dimension onwards are how we store these “base matrix”

- What if we have four dimension arrary?

```
> array(1: 24, dim = c(1, 2, 2, 3))
, , 1, 1

     [,1] [,2]
[1,]    1    2

, , 2, 1

     [,1] [,2]
[1,]    3    4

, , 1, 2

     [,1] [,2]
[1,]    5    6

, , 2, 2

     [,1] [,2]
[1,]    7    8

, , 1, 3

     [,1] [,2]
[1,]    9   10

, , 2, 3

     [,1] [,2]
[1,]   11   12
```

Why is an array helpful?

- It stores our data in a structural way. Easy to use. For example, [student, student characteristics, lab number]. Have to be same data type (atomic).

```
> myarray
, , 1

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

, , 2

     [,1] [,2] [,3]
[1,]    7    9   11
[2,]    8   10   12

, , 3

     [,1] [,2] [,3]
[1,]   13   15   17
[2,]   14   16   18
```

Subsetting array

Myarray [student x 2, student characteristics x 3, lab number x 3]

```
> myarray
, , 1

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

, , 2

     [,1] [,2] [,3]
[1,]    7    9   11
[2,]    8   10   12

, , 3

     [,1] [,2] [,3]
[1,]   13   15   17
[2,]   14   16   18
> # what would myarray[2,3,1] gives me?
```

myarray [student x 2, student characteristics x 3, lab number x 3]

```
> myarray
, , 1

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

, , 2

     [,1] [,2] [,3]
[1,]    7    9   11
[2,]    8   10   12

, , 3

     [,1] [,2] [,3]
[1,]   13   15   17
[2,]   14   16   18


> # how do I get the first characteristic for the second student of every lab?
```

```
> myarray[2,1,]
[1]  2  8 14
```

- Empty argument represents “all”

Data frame

```
> library(titanic)
> df = titanic_train[,1:6] # suppose I have a new data set


> str(df)
'data.frame':   891 obs. of  6 variables:
 $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
 $ Survived   : int  0 1 1 1 0 0 0 0 1 1 ...
 $ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
 $ Name       : chr  "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Fl
orence Briggs Thayer)" "Heikkinen, Miss. Laina" "Futrelle, Mrs. Jacques Heath
 (Lily May Peel)" ...
 $ Sex        : chr  "male" "female" "female" "female" ...
 $ Age        : num  22 38 26 35 35 NA 54 2 27 14 ...
```

Summary() gives distribution summary

```
> summary(df)
  PassengerId       Survived          Pclass          Name
 Min.   :  1.0   Min.   :0.0000   Min.   :1.000   Length:891
 1st Qu.:223.5   1st Qu.:0.0000   1st Qu.:2.000   Class :character
 Median :446.0   Median :0.0000   Median :3.000   Mode  :character
 Mean   :446.0   Mean   :0.3838   Mean   :2.309
 3rd Qu.:668.5   3rd Qu.:1.0000   3rd Qu.:3.000
 Max.   :891.0   Max.   :1.0000   Max.   :3.000

     Sex                 Age
 Length:891         Min.   : 0.42
 Class :character   1st Qu.:20.12
 Mode  :character   Median :28.00
                    Mean   :29.70
                    3rd Qu.:38.00
                    Max.   :80.00
                    NA's   :177
```

Always helpful to look at the actual data

```
> View(head(df,15))
```

	PassengerId	Survived	Pclass	Name	Sex	Age
1	1	0	3	Braund, Mr. Owen Harris	male	22
2	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	female	38
3	3	1	3	Heikkinen, Miss. Laina	female	26
4	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35
5	5	0	3	Allen, Mr. William Henry	male	35
6	6	0	3	Moran, Mr. James	male	*NA*
7	7	0	1	McCarthy, Mr. Timothy J	male	54
8	8	0	3	Palsson, Master. Gosta Leonard	male	2
9	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27
10	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14
11	11	1	3	Sandstrom, Miss. Marguerite Rut	female	4
12	12	1	1	Bonnell, Miss. Elizabeth	female	58
13	13	0	3	Saundercock, Mr. William Henry	male	20
14	14	0	3	Andersson, Mr. Anders Johan	male	39
15	15	0	3	Vestrom, Miss. Hulda Amanda Adolfina	female	14

	PassengerId	Survived	Pclass	Name	Sex	Age
2	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	female	38
3	3	1	3	Heikkinen, Miss. Laina	female	26
4	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35
9	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27
10	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14
11	11	1	3	Sandstrom, Miss. Marguerite Rut	female	4
12	12	1	1	Bonnell, Miss. Elizabeth	female	58
1	1	0	3	Braund, Mr. Owen Harris	male	22
5	5	0	3	Allen, Mr. William Henry	male	35
6	6	0	3	Moran, Mr. James	male	*NA*
7	7	0	1	McCarthy, Mr. Timothy J	male	54
8	8	0	3	Palsson, Master. Gosta Leonard	male	2
13	13	0	3	Saundercock, Mr. William Henry	male	20
14	14	0	3	Andersson, Mr. Anders Johan	male	39
15	15	0	3	Vestrom, Miss. Hulda Amanda Adolfina	female	14

Filter male

	PassengerId	Survived	Pclass	Name	Sex	Age
2	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	female	38
3	3	1	3	Heikkinen, Miss. Laina	female	26
4	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35
9	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27
10	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14
11	11	1	3	Sandstrom, Miss. Marguerite Rut	female	4
12	12	1	1	Bonnell, Miss. Elizabeth	female	58
1	1	0	3	Braund, Mr. Owen Harris	male	22
5	5	0	3	Allen, Mr. William Henry	male	35
6	6	0	3	Moran, Mr. James	male	*NA*
7	7	0	1	McCarthy, Mr. Timothy J	male	54
8	8	0	3	Palsson, Master. Gosta Leonard	male	2
13	13	0	3	Saundercock, Mr. William Henry	male	20
14	14	0	3	Andersson, Mr. Anders Johan	male	39
15	15	0	3	Vestrom, Miss. Hulda Amanda Adolfina	female	14

Use table() to get frequency

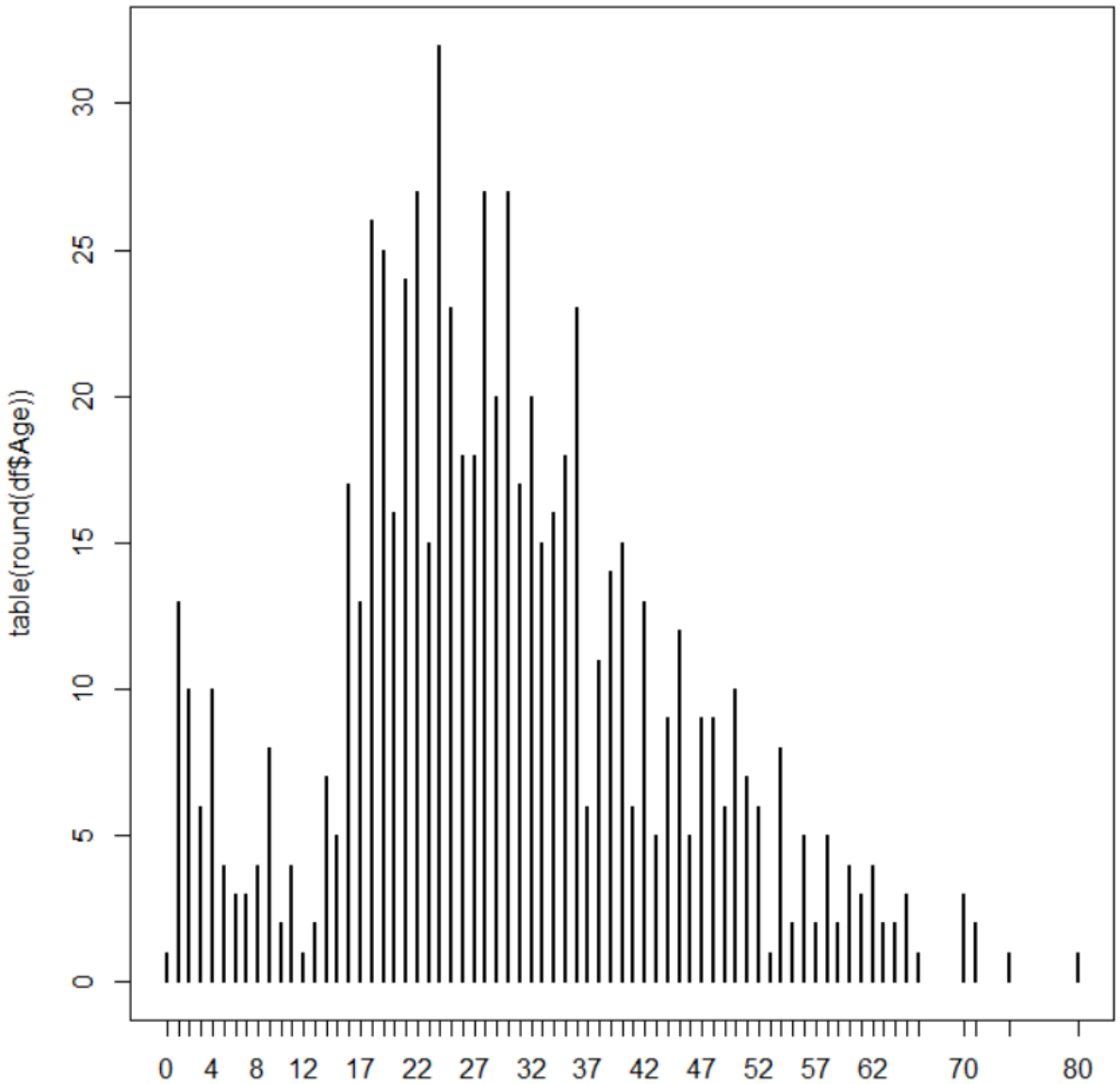
```
> table(df$Age)

0.42 0.67 0.75 0.83 0.92    1    2    3    4    5    6    7    8    9   10   11   12   13   14
   1    1    2    2    1    7   10    6   10    4    3    3    4    8    2    4    1    2    6
14.5   15   16   17   18   19   20 20.5   21   22   23 23.5   24 24.5   25   26   27   28 28.5
   1    5   17   13   26   25   15    1   24   27   15    1   30    1   23   18   18   25    2
  29   30 30.5   31   32 32.5   33   34 34.5   35   36 36.5   37   38   39   40 40.5   41   42
  20   25    2   17   18    2   15   15    1   18   22    1    6   11   14   13    2    6   13
  43   44   45 45.5   46   47   48   49   50   51   52   53   54   55 55.5   56   57   58   59
   5    9   12    2    3    9    9    6   10    7    6    1    8    2    1    4    2    5    2
  60   61   62   63   64   65   66   70 70.5   71   74   80
   4    3    4    2    2    3    1    2    1    2    1    1
```

```
> table(round(df$Age))

 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
 1 13 10  6 10  4  3  3  4  8  2  4  1  2  7  5 17 13 26 25 16 24 27 15 32 23 18 18 27 20 27
31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61
17 20 15 16 18 23  6 11 14 15  6 13  5  9 12  5  9  9  6 10  7  6  1  8  2  5  2  5  2  4  3
62 63 64 65 66 70 71 74 80
 4  2  2  3  1  3  2  1  1
```

```
> plot(table(round(df$Age)))
```



Recall how to do subsetting…

```
# how many female passengers older than 25 years old survived?
sum(df$Survived[df$Sex == 'female' & df$Age > 25])
```

```
## [1] NA
```

```r
sum(na.omit(df$Survived[df$Sex == 'female' & df$Age > 25]))
```

```
## [1] 110
```

```r
sum(df$Survived[df$Sex == 'female' & df$Age > 25 & !is.na(df$Age)])
```

```
## [1] 110
```

Merge: I have some additional information to add

- Suppose now I know if they could swim

```
swimdf =  data.frame(
  id = c(10,11),
  swim = c('yes','no')
)
```

- Merge by unique identifier in both table. How do I know if it is unique?

```
> length(unique(df$PassengerId))
[1] 891
> dim(df)
[1] 891  12
```

How do I know if the IDs are 1-1 matches?

```
> sum(swimdf$id %in% df$PassengerId)
[1] 2
```

```
> dfmatched = merge(df, swimdf, by.x = 'PassengerId', by.y = 'id')
> dfmatched
  PassengerId Survived Pclass                                     Name    Sex Age swim
1          10        1      2 Nasser, Mrs. Nicholas (Adele Achem) female  14  yes
2          11        1      3     Sandstrom, Miss. Marguerite Rut female   4   no
```

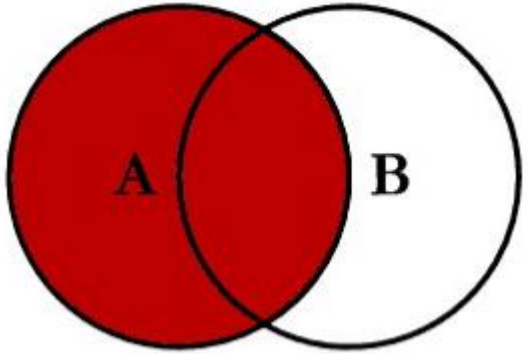
But swim data has only two observations?

```
> dfall = merge(df, swimdf, by.x = 'PassengerId', by.y = 'id', all = T)
```

8	8	0	3	Palsson, Master. Gosta Leonard	male	2.00	*NA*
9	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.00	*NA*
10	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.00	yes
11	11	1	3	Sandstrom, Miss. Marguerite Rut	female	4.00	no
12	12	1	1	Bonnell, Miss. Elizabeth	female	58.00	*NA*
13	13	0	3	Saundercock, Mr. William Henry	male	20.00	*NA*

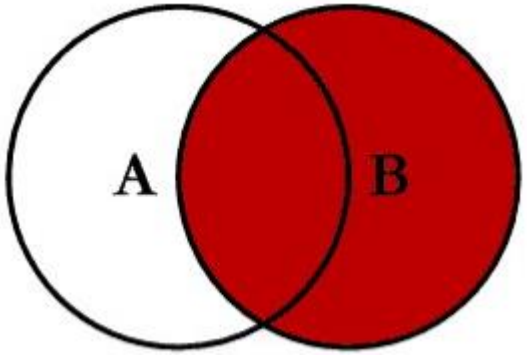
Conceptual relationship to join

- It’s all about the all = TRUE statement



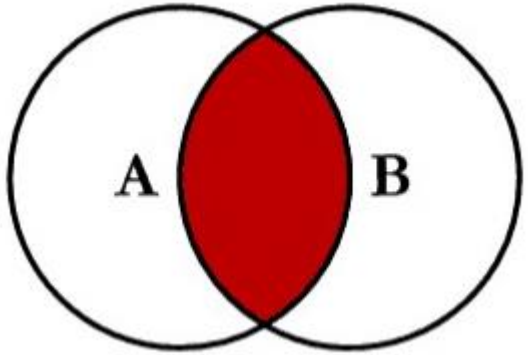
A left join B

all.x = TRUE



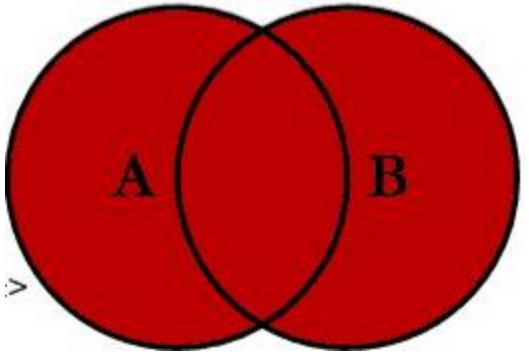
A right join B

all.y = TRUE



A inner join B

No all statement (default)



A outer join B

all = TRUE

1-n/n-n merge generates duplicates that might be wrong

- Suppose we have some data collection error in swim data

```
swimdf =  data.frame(
  id = c(10,11,11),
  swim = c('yes','yes','no')
)
```

```
> merge(df, swimdf, by.x = 'PassengerId', by.y = 'id')
  PassengerId Survived Pclass                                Name    Sex Age swim
1          10        1      2 Nasser, Mrs. Nicholas (Adele Achem) female  14  yes
2          11        1      3     Sandstrom, Miss. Marguerite Rut female   4  yes
3          11        1      3     Sandstrom, Miss. Marguerite Rut female   4   no
```

aggregate

- Collapse ***data*** by some ***variable*** to ***summarize***

```
aggregate(x = df$Survived, # what does this return?
          by = list(df$Pclass),
          FUN = mean)
```

```
#   Group.1       x
#     1    0.6296296
#     2    0.4728261
#     3    0.2423625
```

```r
aggregate(formula = Survived ~ Pclass, # alternative specification
          data = df,
          FUN = mean)
```

```
# Pclass  Survived
#      1 0.6296296
#      2 0.4728261
#      3 0.2423625
```

```r
# Average age by class and survived status
aggregate(formula = Age ~ Pclass + Survived,
          data = df,
          FUN = mean)
```

```
# Pclass Survived     Age
#    1        0       43.69531
#    2        0       33.54444
#    3        0       26.55556
#    1        1       35.36820
#    2        1       25.90157
#    3        1       20.64612
```

```r
# Average age by class and survived status within males
aggregate(formula = Age ~ Pclass + Survived,
          data = df[df$Sex == 'male',],
          FUN = mean)
```

```
# Pclass Survived      Age
#      1        0 44.58197
#      2        0  33.36905
#      3        0 27.25581
#      1        1 36.24800
#      2        1 16.02200
#      3        1 22.27421
```

Melt: Wide to Long

```
library(MASS)
library(reshape2)
library(reshape)
data(ships)
head(ships)
```

## type	year	period	service	incidents
## A	60	60	127	0
## A	60	75	63	0
## A	65	60	1095	3
## A	65	75	1095	4
## A	70	60	1512	6
## A	70	75	3353	18

```
> summary(ships)
 type       year          period          service           incidents
 A:8   Min.   :60.00   Min.   :60.0   Min.   :    0.0   Min.   : 0.0
 B:8   1st Qu.:63.75   1st Qu.:60.0   1st Qu.:  175.8   1st Qu.: 0.0
 C:8   Median :67.50   Median :67.5   Median :  782.0   Median : 2.0
 D:8   Mean   :67.50   Mean   :67.5   Mean   : 4089.3   Mean   : 8.9
 E:8   3rd Qu.:71.25   3rd Qu.:75.0   3rd Qu.: 2078.5   3rd Qu.:11.0
       Max.   :75.00   Max.   :75.0   Max.   :44882.0   Max.   :58.0
> str(ships)
'data.frame':   40 obs. of  5 variables:
 $ type     : Factor w/ 5 levels "A","B","C","D",..: 1 1 1 1 1 1 1 1 2 2 ...
 $ year     : int  60 60 65 65 70 70 75 75 60 60 ...
 $ period   : int  60 75 60 75 60 75 60 75 60 75 ...
 $ service  : int  127 63 1095 1095 1512 3353 0 2244 44882 17176 ...
 $ incidents: int  0 0 3 4 6 18 0 11 39 29 ...
```

```r
# keep type and year as constant (id variable), and melt the rest
melt(ships, id = c("type","year"))
```

```
##  type year period service incidents
##    A   60     60     127         0
##    A   60     75      63         0
##    A   65     60    1095         3
##    A   65     75    1095         4
##    A   70     60    1512         6
##    A   70     75    3353        18
```

```
##     type year  variable value
## 1      A   60    period    60
## 2      A   60    period    75
## 3      A   65    period    60
## 4      A   65    period    75
## 5      A   70    period    60
## 6      A   70    period    75
## 7      A   75    period    60
## 8      A   75    period    75
## 9      B   60    period    60
## 10     B   60    period    75
## 11     A   60   service   127
## 12     A   60   service    63
## 13     A   65   service  1095
## 14     A   65   service  1095
## 15     A   70   service  1512
## 16     A   70   service  3353
```

Cast: Long to Wide

```
# aggregate occurs when id variables do not identify indiviudal observations
molten.ships = melt(ships, id = c("type","year"))
cast(molten.ships, type + year ~ variable, sum)
```

```
##    type year  variable value
## 1     A   60    period    60
## 2     A   60    period    75
## 3     A   65    period    60
## 4     A   65    period    75
## 5     A   70    period    60
## 6     A   70    period    75
## 7     A   75    period    60
## 8     A   75    period    75
## 9     B   60    period    60
## 10    B   60    period    75
## 11    A   60   service   127
## 12    A   60   service    63
## 13    A   65   service  1095
## 14    A   65   service  1095
## 15    A   70   service  1512
## 16    A   70   service  3353
```

```
##    type year period service incidents
## 1     A   60    135     190         0
## 2     A   65    135    2190         7
## 3     A   70    135    4865        24
## 4     A   75    135    2244        11
## 5     B   60    135   62058        68
```

Cast is like aggregate, but different order

```
# if I only use type as the id, then aggregate values by type
molten.ships = melt(ships, id = c("type","year"))
cast(molten.ships, type ~ variable, sum)
```

```
##    type year  variable value
## 1     A   60    period    60
## 2     A   60    period    75
## 3     A   65    period    60
## 4     A   65    period    75
## 5     A   70    period    60
## 6     A   70    period    75
## 7     A   75    period    60
## 8     A   75    period    75
## 9     B   60    period    60
## 10    B   60    period    75
## 11    A   60   service   127
## 12    A   60   service    63
## 13    A   65   service  1095
## 14    A   65   service  1095
## 15    A   70   service  1512
## 16    A   70   service  3353
```

```
##    type period  service   incidents
## 1    A    540     9489        42
## 2    B    540    138317      253
## 3    C    540     6193        12
## 4    D    540     4444        17
## 5    E    540     5131        32
```