MA415/615 Assignment 4

Ziran Min

February 21st 2018

10.5.1

How can you tell if an object is a tibble? (Hint: try printing mtcars, which is a regular data frame).

```
print(mtcars)
```

```
##                     mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4          21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag      21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710         22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive     21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout  18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant            18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360         14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D          24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230           22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280           19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C          17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE         16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL         17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC        15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood 10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4  8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial  14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128           32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic        30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla     33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona      21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger   15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin        15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28         13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird   19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9          27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2      26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa       30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L     15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino       19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora      15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E         21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

```
class(mtcars)
```

```
## [1] "data.frame"
```

```
#we can make data as tibbles by using as_tibble() and check its class by class()
as_tibble(mtcars)
```

```
## # A tibble: 32 x 11
##      mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
```

```
##  * <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1  21.0  6.00   160 110    3.90  2.62  16.5  0     1.00  4.00  4.00
##  2  21.0  6.00   160 110    3.90  2.88  17.0  0     1.00  4.00  4.00
##  3  22.8  4.00   108  93.0  3.85  2.32  18.6  1.00  1.00  4.00  1.00
##  4  21.4  6.00   258 110    3.08  3.22  19.4  1.00  0     3.00  1.00
##  5  18.7  8.00   360 175    3.15  3.44  17.0  0     0     3.00  2.00
##  6  18.1  6.00   225 105    2.76  3.46  20.2  1.00  0     3.00  1.00
##  7  14.3  8.00   360 245    3.21  3.57  15.8  0     0     3.00  4.00
##  8  24.4  4.00   147  62.0  3.69  3.19  20.0  1.00  0     4.00  2.00
##  9  22.8  4.00   141  95.0  3.92  3.15  22.9  1.00  0     4.00  2.00
## 10  19.2  6.00   168 123    3.92  3.44  18.3  1.00  0     4.00  4.00
## # ... with 22 more rows
```

```
class(as_tibble(mtcars))
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

Tibbles only prints the toppest few rows of the data and the class of each columns.

10.5.2

Compare and contrast the following operations on a data.frame and equivalent tibble. What is different? Why might the default data frame behaviours cause you frustration?

```
#df does partial matching
df <- data.frame(abc = 1, xyz = "a")
df$x
```

```
## [1] a
## Levels: a
```

```
#df returns a factor
df[, "xyz"]
```

```
## [1] a
## Levels: a
```

```
#returns data frame
df[, c("abc", "xyz")]
```

```
##   abc xyz
## 1   1   a
```

```
#tibble doesn't do partial matching
dftibble <- as_tibble(df)
dftibble$x
```

```
## Warning: Unknown or uninitialised column: 'x'.
```

```
## NULL
```

```
#tibble returns a data frame
dftibble[, "xyz"]
```

```
## # A tibble: 1 x 1
##   xyz
##   <fct>
## 1 a
```

```
#tibbles have class in top of each column
dftibble[, c("abc", "xyz")]
```

```
## # A tibble: 1 x 2
##     abc xyz
##   <dbl> <fct>
## 1  1.00 a
```

10.5.3

If you have the name of a variable stored in an object, e.g. var <- “mpg”, how can you extract the reference variable from a tibble?

```
var <- "abc"
dftibble[[var]]
```

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

```
dftibble[var]
```

```
## # A tibble: 1 x 1
##     abc
##   <dbl>
## 1  1.00
```

10.5.4

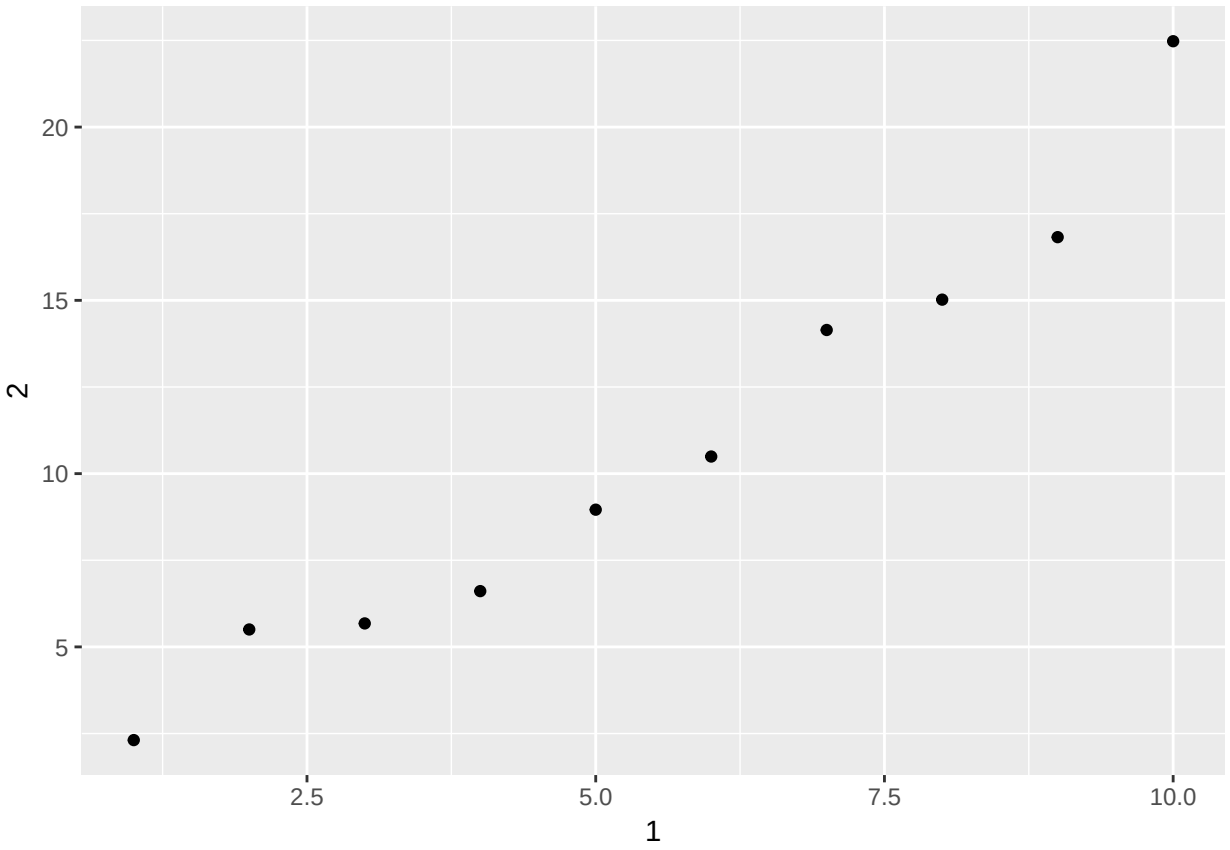
Practice referring to non-syntactic names in the following data frame by:

1. Extracting the variable called 1.
2. Plotting a scatterplot of 1 vs 2.
3. Creating a new column called 3 which is 2 divided by 1.
4. Renaming the columns to one, two and three.

```
annoying <- tibble(
  `1` = 1:10,
  `2` = `1` * 2 + rnorm(length(`1`))
)
#1. Extracting the variable called 1.
annoying$`1`
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
#2. Plotting a scatterplot of 1 vs 2.
ggplot(annoying, aes(x = `1`, y = `2`)) + geom_point()
```




```
#3. Creating a new column called 3 which is 2 divided by 1.
annoying[["3"]] <- annoying$`2` / annoying$`1`
#4. Renaming the columns to one, two and three.
annoying <- rename(annoying, one = `1`, two = `2`, three = `3`)
glimpse(annoying)
```

```
## Observations: 10
## Variables: 3
## $ one   <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
## $ two   <dbl> 2.310784, 5.503039, 5.677568, 6.609291, 8.959926, 10.493...
## $ three <dbl> 2.310784, 2.751519, 1.892523, 1.652323, 1.791985, 1.7488...
```

10.5.5

What does tibble::enframe() do? When might you use it?

enframe() converts named atomic vectors or lists to two-column data frames. For unnamed vectors, the natural sequence is used as name column.

The usage is: enframe(x, name = "name", value = "value")

```
enframe(c(a = 5, b = 7))
```

```
## # A tibble: 2 x 2
##   name  value
##   <chr> <dbl>
## 1 a      5.00
## 2 b      7.00
```

10.5.6

What option controls how many additional column names are printed at the footer of a tibble?

```
#use print.tbl_df
?print.tbl_df
#examples
print(as_tibble(mtcars), n = 5)
```

```
## # A tibble: 32 x 11
##     mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
## * <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1  21.0  6.00   160 110    3.90  2.62  16.5  0     1.00  4.00  4.00
## 2  21.0  6.00   160 110    3.90  2.88  17.0  0     1.00  4.00  4.00
## 3  22.8  4.00   108  93.0  3.85  2.32  18.6  1.00  1.00  4.00  1.00
## 4  21.4  6.00   258 110    3.08  3.22  19.4  1.00  0     3.00  1.00
## 5  18.7  8.00   360 175    3.15  3.44  17.0  0     0     3.00  2.00
## # ... with 27 more rows
```

12.6.1

1. In this case study I set na.rm = TRUE just to make it easier to check that we had the correct values. Is this reasonable? Think about how missing values are represented in this dataset. Are there implicit missing values? What's the difference between an NA and zero?
2. What happens if you neglect the mutate() step? (mutate(key = stringr::str_replace(key, "newrel", "new_rel")))
3. I claimed that iso2 and iso3 were redundant with country. Confirm this claim.
4. For each country, year, and sex compute the total number of cases of TB. Make an informative visualisation of the data.