In-Class Exercise 3

Nathan Van Ymeren

Preface:

In previous documents I reproduced the question text entirely. This time to make it clearer I'm going to omit the question text and just have some explanatory prose in each section.

First let's load the tidyverse and our dataset:

```
library(tidyverse)
companies = readRDS("North American Stock Market 1994-2018.rds")
```

Question 1, 1.25 pts

The question asks us, across all 288,158 observations in all 41 variables in the "companies" dataset, how many NA's are there in total? There's a handy hint that `is.na()` works with data frames, which is great because we can just do the following:

```
sum(is.na(companies))
```

Which returns our answer, which is choice D.

```
1637332
```

Question 2, 1.25 pts

We're being asked to find the minimum value of the `sale` variable within the dataset, and the answer choices are rounded to one decimal place. For that we'll use the janky "dollar sign" syntax to access the `sale` column, and give that as an argument to the `min()` function, which returns the minimum value in a list or a vector. Then we'll give all *that* as the argument to the `round()` function, which will round to the desired number of decimal places.

Remember from Q1, though, that there are like 1.6 million NA's in this data set so if we just ask for the minimum value we'll probably get NA. That means we should set `na.rm=TRUE` when we call `min()`:

```
round( min( companies$sale, na.rm=TRUE ), digits=1 )
```

And the answer is B:

```
-15009.3
```

Question 3, 1.25 pts

Now we're being asked to find all the observations (rows) where the variables `at` and `lt` are greater than or equal to $10M, but since the numbers are stored in millions we're going to be doing something like `lt >= 10`. We want to keep all the observations (rows) where this is true, meaning we don't want to drop columns, so this suggests we should reach for the `filter()` function and pass it a predicate stipulating `whatever >= 10`. Then, we need to take that screened sample and find the highest ration of `at` to `lt`. So my first instinct is to do it this way:

1. Make a new data frame by filtering `companies`
2. Make a new column in that new data frame where the values are `at/lt`
3. Find the max of that.

Let's see if it works, using `magrittr`-style pipes. I'm going to add the `gvkey` variable because I see we'll need it for the next question.

```
  q3 = companies %>%
    select(gvkey, lt, at) %>%
    filter(lt>=10, at>=10, !is.na(lt), !is.na(at)) %>%
    mutate(ratio = at/lt)

max(q3$ratio)
```

We could round that if we wanted but it's close enough to see that the answer is B:

```
423.585221582374
```

Question 4, 1.25 pts

We can reuse a bunch of the work from the previous question to find the `gvkey`, which is like an index number, of the company with the highest ratio:

```
q3 %>%
  filter(ratio == max(ratio)) %>%
  pull(gvkey)
```

What we're doing here is taking the dataset we made in question 3, filtering it such that we only keep rows where the ratio is equal to the maximum ratio (and there should only be one row, but in this case it's easy to check in the console), and then thanks to the magic of `pull()` we can just extract the value in the `gvkey` column, which is answer E:

```
151971
```

Question 5, 1.25 pts

Okay so what's interesting here is it's asking us to actually read and understand code rather than just run it. Off the top of my head I can think of two or three ways to calculate the fraction of companies where `at > 1000` in fiscal year 2007, but one of the (few) nice things about R is never having to write loops, so if we're doing it "the `dplyr` way" we'd need to filter for:

1. fiscal year
2. assets

And then we'd need to count all the companies in that set. Then we'd create a second set where we only filter for fiscal year, count each set, and then divide the first set by the second. You can see right away by looking at the answers that only two of them use the `n()` function which returns the number of items in a vector/list, so right away we know those are the only two viable choices (unless it's none of them). The first, answer B, uses `select()` to compute the numerator and denominator. This is an error, because `select()` operates on columns, but you want to drop rows where `fyear != 2007` and the way to exclude rows is by using `filter()`. So logically we can conclude the only choice is between answers D or E.

Let's just see what D gives us in greater detail. First we compute the numerator:

```
numerator1 = companies %>%
  filter(fyear == 2007, at > 1000) %>%
  summarize(n())
```

Which gives:

```
2849
```

Seems reasonable. Let's check the denominator.

```
denominator = companies %>%
  filter(fyear == 2007) %>%
  summarize(n())
```

And that gives:

```
10870
```

Also seems reasonable. We don't have an answer to compare against but we can divide them and round just for fun:

```
round(numerator1/denominator, 3)
```

And that comes out to:

```
0.262
```

So, 26% of companies in 2007 have assets greater than $1B? I buy it. I chose answer D.

Question 6, 1.25 pts

We want "only the following columns": company name, employment, fiscal year. That we want to choose entire columns is our cue to use `select()`, and also remove any rows were employment is missing values, which is our cue to use `filter()`, and then again we also want to limit our analysis to fiscal year 2010. This is a slight variation on the previous question, so we can follow a similar approach:

1. Run a selection on `companies` for `conm`, `emp`, and `fyear`
2. Filter that selection for `fyear == 2010` and `!is.na(emp)`

We can see by inspecting the available choices that only options C and D correctly use `select()` and of those two, only C uses the predicate correctly. Answer D uses `na.rm=TRUE` which is not a Boolean expression and thus not a predicate. For calls to `filter()` we need to use expressions that evaluate to either TRUE or FALSE, which means we need to call the `is.na()` predicate function, thus the answer is C.

Question 7, 1.25 pts

Now we get to actually run the code from Question 6 so let's do that, and pipe that into `max()` and `min()`, and then subtract them.

```
  q7max = companies %>%
    select(conm, emp, fyear) %>%
    filter(!is.na(emp), fyear ==2010) %>%
    summarize(maxemp = max(emp))

  q7min = companies %>%
    select(conm, emp, fyear) %>%
    filter(!is.na(emp), fyear ==2010) %>%
    summarize(minemp = min(emp))

q7max - q7min
```

That's pretty straightforward. I bet there's an elegant way to do the whole thing in one pipeline but whatever. Our answer is B:

2100

Question 8, 1.25 pts

Continuing from the previous, we're asked to take the dataset (which the quiz calls `df1` but I called something else) and add a new variable. This is our cue right away to start thinking of `mutate()` which adds variables. It wants us to list actual employment, which in the data is represented in thousands. So it should be a simple matter of calling something like `mutate(emp_actual = emp * 1000)`. Looking at the choices we can see that D is multiplying by too large a constant. Answer C is using `==`, the equality test operator, and not `=` which is the assignment operator. So that leaves answers A and B, of which A starts the pipeline with the full `companies` dataset. But it doesn't do all the steps to filter and select from the original data, so we'd have way too many columns and rows if we went with A. Thus by process of elimination the answer is B.