QBS181_ProblemSet3

Matthew Chan

10/28/2021

Part One

Question One

Recall the HigherMe dataset. On the canvas homepage, I've linked a csv file which contains the data cleaning up to the step where we need to convert those invoices which are quarterly to monthly invoices. Using the same logic you applied in the excel project, convert quarterly invoices to monthly invoices in R. Display the first 10 rows of your updated dataset.

```
#your code here
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v stringr 1.4.0
## v tidyr   1.1.3     v forcats 0.5.1
## v readr   2.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x lubridate::as.difftime() masks base::as.difftime()
## x lubridate::date()        masks base::date()
## x dplyr::filter()          masks stats::filter()
## x lubridate::intersect()   masks base::intersect()
## x dplyr::lag()             masks stats::lag()
## x lubridate::setdiff()     masks base::setdiff()
## x lubridate::union()       masks base::union()
```

```
library(data.table)
```

```
## 
## Attaching package: 'data.table'

## The following object is masked from 'package:purrr':
## 
##     transpose

## The following objects are masked from 'package:lubridate':
## 
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year

## The following objects are masked from 'package:dplyr':
## 
##     between, first, last
```

```
Higherme <- read.csv("HigherMeDataForRData.csv")

names(Higherme)[names(Higherme) == "ï..Invoice.Number"] <- "Invoice_Number"
Higherme$Invoice.Date <- as.Date(Higherme$Invoice.Date, format = "%m/%d/%y")
head(Higherme)
```

```
##   Invoice_Number Invoice.Date      Amount Status       Paid.On Refunded.Amount
## 1          49236   2021-09-14    $115.00    Paid 9/14/21 22:33               0
## 2          49235   2021-09-14    $115.00    Paid 9/14/21 22:33               0
## 3          49213   2021-09-14    $360.00    Paid 9/14/21 21:42               0
## 4          49212   2021-09-14  $4,455.00    Paid 9/14/21 21:28               0
## 5          49183   2021-09-14     $57.50    Paid  9/14/21 5:05               0
## 6          49149   2021-09-14  $2,640.00    Paid  9/14/21 0:02               0
##   Recurring Quarterly.Monthly First.Invoice   Company Tax.Total Amount.Due
## 1     FALSE           MONTHLY         FALSE Company 1     $-         $-
## 2     FALSE           MONTHLY         FALSE Company 1     $-         $-
## 3      TRUE         QUARTERLY         FALSE Company 2     $-         $-
## 4      TRUE         QUARTERLY         FALSE Company 3     $-         $-
## 5     FALSE           MONTHLY         FALSE Company 3     $-         $-
## 6      TRUE         QUARTERLY         FALSE Company 4     $-         $-
##   Adjustments Credits.Applied    Payments Write.Off.Amount Currency
## 1       $-              $-       $115.00             $-         USD
## 2       $-              $-       $115.00             $-         USD
## 3       $-              $-       $360.00             $-         USD
## 4       $-              $-     $4,455.00             $-         USD
## 5       $-              $-        $57.50             $-         USD
## 6       $-              $-     $2,640.00             $-         USD
##        Due.Date Customer.Billing.Country  X
## 1 9/14/21 22:33                          NA
## 2 9/14/21 22:33                          NA
## 3 9/14/21 21:42                       US NA
## 4 9/14/21 21:28                          NA
## 5  9/14/21 5:05                          NA
## 6  9/14/21 0:02                          NA
```

```r
# Remove dollar signs function as provided in class
rmCurrency<-function(x){

  x<-trimws(x) #trim whitespace

  if(grepl("\\$",x[1])){ #if '$' found in x
    x<-sub("\\$", "",x) #remove $
    x<-sub("\\,", "",x) #remove ,
    x[x=="-"]<-0 #recode zeros
  }
  return(x)
}
Higherme$Amount <- rmCurrency(Higherme$Amount)
Higherme$Amount <- as.integer(Higherme$Amount)
Higherme$Amount <- Higherme$Amount/ 3
head(Higherme)
```

```
##   Invoice_Number Invoice.Date     Amount Status       Paid.On Refunded.Amount
## 1          49236   2021-09-14   38.33333   Paid 9/14/21 22:33               0
## 2          49235   2021-09-14   38.33333   Paid 9/14/21 22:33               0
## 3          49213   2021-09-14  120.00000   Paid 9/14/21 21:42               0
## 4          49212   2021-09-14 1485.00000   Paid 9/14/21 21:28               0
## 5          49183   2021-09-14   19.00000   Paid  9/14/21 5:05               0
## 6          49149   2021-09-14  880.00000   Paid  9/14/21 0:02               0
##   Recurring Quarterly.Monthly First.Invoice   Company Tax.Total Amount.Due
## 1     FALSE           MONTHLY         FALSE Company 1     $-         $-
## 2     FALSE           MONTHLY         FALSE Company 1     $-         $-
## 3      TRUE         QUARTERLY         FALSE Company 2     $-         $-
## 4      TRUE         QUARTERLY         FALSE Company 3     $-         $-
## 5     FALSE           MONTHLY         FALSE Company 3     $-         $-
## 6      TRUE         QUARTERLY         FALSE Company 4     $-         $-
##   Adjustments Credits.Applied    Payments Write.Off.Amount Currency
## 1       $-              $-       $115.00             $-         USD
## 2       $-              $-       $115.00             $-         USD
## 3       $-              $-       $360.00             $-         USD
## 4       $-              $-     $4,455.00             $-         USD
## 5       $-              $-        $57.50             $-         USD
## 6       $-              $-     $2,640.00             $-         USD
##        Due.Date Customer.Billing.Country  X
## 1 9/14/21 22:33                          NA
## 2 9/14/21 22:33                          NA
## 3 9/14/21 21:42                       US NA
## 4 9/14/21 21:28                          NA
## 5  9/14/21 5:05                          NA
## 6  9/14/21 0:02                          NA
```

```
quarterly2 <- Higherme %>%
  filter(Quarterly.Monthly == "QUARTERLY") %>%
  arrange(Invoice_Number)
quarterly3 <- Higherme %>%
  filter(Quarterly.Monthly == "QUARTERLY") %>%
  arrange(Invoice_Number)
```

```
quarterly2$Invoice.Date <- ymd(quarterly2$Invoice.Date) - 30
head(quarterly2)
```

```
##   Invoice_Number Invoice.Date   Amount Status       Paid.On Refunded.Amount
## 1           5937   2018-02-12  70.0000   Paid 3/14/18 21:40               0
## 2           6406   2018-03-15  70.0000   Paid 4/14/18 21:40               0
## 3           6701   2018-04-03  25.0000   Paid  5/3/18 15:53               0
## 4           6714   2018-04-04 883.3333   Paid  5/4/18 13:01               0
## 5           6766   2018-04-08 166.6667   Paid  5/8/18 17:06             300
## 6           6788   2018-04-09 704.3333   Paid  5/9/18 20:26               0
##   Recurring Quarterly.Monthly First.Invoice    Company Tax.Total Amount.Due
## 1      TRUE         QUARTERLY          TRUE  Company 2     $-         $-
## 2      TRUE         QUARTERLY         FALSE  Company 2     $-         $-
## 3      TRUE         QUARTERLY          TRUE Company 28     $-         $-
## 4      TRUE         QUARTERLY          TRUE Company 27     $-         $-
## 5      TRUE         QUARTERLY          TRUE Company 19     $-         $-
## 6      TRUE         QUARTERLY         FALSE Company 27     $-         $-
##   Adjustments Credits.Applied    Payments Write.Off.Amount Currency
## 1       $-              $-       $210.00             $-         USD
## 2       $-              $-       $210.00             $-         USD
## 3       $-              $-        $75.00             $-         USD
## 4       $-              $-     $2,650.00             $-         USD
## 5       $-              $-       $500.00             $-         CAD
## 6       $-              $-     $2,113.44             $-         USD
##        Due.Date Customer.Billing.Country  X
## 1 3/14/18 21:40                       US NA
## 2 4/14/18 21:40                       US NA
## 3  5/3/18 15:53                          NA
## 4  5/4/18 13:01                          NA
## 5  5/8/18 17:06                       CA NA
## 6  5/9/18 20:26                          NA
```

```
quarterly3$Invoice.Date <- ymd(quarterly3$Invoice.Date) - 60
head(quarterly3)
```

```
##   Invoice_Number Invoice.Date   Amount Status       Paid.On Refunded.Amount
## 1           5937   2018-01-13  70.0000   Paid 3/14/18 21:40               0
## 2           6406   2018-02-13  70.0000   Paid 4/14/18 21:40               0
## 3           6701   2018-03-04  25.0000   Paid  5/3/18 15:53               0
## 4           6714   2018-03-05 883.3333   Paid  5/4/18 13:01               0
## 5           6766   2018-03-09 166.6667   Paid  5/8/18 17:06             300
## 6           6788   2018-03-10 704.3333   Paid  5/9/18 20:26               0
##   Recurring Quarterly.Monthly First.Invoice    Company Tax.Total Amount.Due
## 1      TRUE         QUARTERLY          TRUE  Company 2     $-          $-
## 2      TRUE         QUARTERLY         FALSE  Company 2     $-          $-
## 3      TRUE         QUARTERLY          TRUE Company 28     $-          $-
## 4      TRUE         QUARTERLY          TRUE Company 27     $-          $-
## 5      TRUE         QUARTERLY          TRUE Company 19     $-          $-
## 6      TRUE         QUARTERLY         FALSE Company 27     $-          $-
##   Adjustments Credits.Applied    Payments Write.Off.Amount Currency
## 1       $-              $-       $210.00             $-         USD
## 2       $-              $-       $210.00             $-         USD
## 3       $-              $-        $75.00             $-         USD
## 4       $-              $-     $2,650.00             $-         USD
## 5       $-              $-       $500.00             $-         CAD
## 6       $-              $-     $2,113.44             $-         USD
##        Due.Date Customer.Billing.Country  X
## 1 3/14/18 21:40                       US NA
## 2 4/14/18 21:40                       US NA
## 3  5/3/18 15:53                          NA
## 4  5/4/18 13:01                          NA
## 5  5/8/18 17:06                       CA NA
## 6  5/9/18 20:26                          NA
```

```
# Binding my quarterly3 and quarterly2 dataframe together
merged <- rbind(quarterly3, quarterly2)
```

```
# Putting together the final dataframe
merged_final <- rbind(merged, Higherme)

# Changing all the values to Monthly from Quarterly
merged_final$Quarterly.Monthly[merged_final$Quarterly.Monthly=="QUARTERLY"]<- "MONTHLY"
merged_final <- merged_final %>%
  arrange(merged_final$Invoice_Number)
head(merged_final)
```

```
##   Invoice_Number Invoice.Date Amount Status       Paid.On Refunded.Amount
## 1           5937   2018-01-13     70   Paid 3/14/18 21:40               0
## 2           5937   2018-02-12     70   Paid 3/14/18 21:40               0
## 3           5937   2018-03-14     70   Paid 3/14/18 21:40               0
## 4           6406   2018-02-13     70   Paid 4/14/18 21:40               0
## 5           6406   2018-03-15     70   Paid 4/14/18 21:40               0
## 6           6406   2018-04-14     70   Paid 4/14/18 21:40               0
##   Recurring Quarterly.Monthly First.Invoice   Company Tax.Total Amount.Due
## 1      TRUE           MONTHLY          TRUE Company 2     $-         $-
## 2      TRUE           MONTHLY          TRUE Company 2     $-         $-
## 3      TRUE           MONTHLY          TRUE Company 2     $-         $-
## 4      TRUE           MONTHLY         FALSE Company 2     $-         $-
## 5      TRUE           MONTHLY         FALSE Company 2     $-         $-
## 6      TRUE           MONTHLY         FALSE Company 2     $-         $-
##   Adjustments Credits.Applied  Payments Write.Off.Amount Currency      Due.Date
## 1       $-              $-     $210.00             $-         USD 3/14/18 21:40
## 2       $-              $-     $210.00             $-         USD 3/14/18 21:40
## 3       $-              $-     $210.00             $-         USD 3/14/18 21:40
## 4       $-              $-     $210.00             $-         USD 4/14/18 21:40
## 5       $-              $-     $210.00             $-         USD 4/14/18 21:40
## 6       $-              $-     $210.00             $-         USD 4/14/18 21:40
##   Customer.Billing.Country  X
## 1                       US NA
## 2                       US NA
## 3                       US NA
## 4                       US NA
## 5                       US NA
## 6                       US NA
```

Question Two

Recall the hospital database. Recreate the physician-referral table from Question 9 on your SQL homework using R. I've loaded a zip folder of all the necessary csv files to do this on canvas, as well as a snapshot of what the table should look like for your reference.

```
#your code here
physician <- fread("Physician.csv")
patient <- fread("Patient.csv")
appointment <- fread("Appointment.csv")
undergoes <- fread("Undergoes.csv")
med_procedure <- fread("Medical_Procedure.csv")
affiliated_with <- fread("Affiliated_With.csv")
department <- fread("Department.csv")
```

```
sub_patient <- patient %>%
  left_join(appointment, by=c("SSN" = "Patient")) %>%
  filter(PCP!= Physician) %>%
  mutate("Patient_SSN" = SSN, "Cost" = 0)
```

```
Table2_join <- patient %>% left_join(undergoes, by=c("SSN"="Patient")) %>% left_join(med_procedure, by=

# Merging both into the dataframe
rawdata <- union_all(sub_patient, Table2_join )
rawdata
```

```
##            SSN                Name            Address    Phone InsuranceID PCP
##   1: 100000001           John Smith     42 Foobar Lane 555-0256    68476213   1
##   2: 100000004           Dennis Doe 1100 Foobaz Avenue 555-2048    68421879   3
##   3: 100000004           Dennis Doe 1100 Foobaz Avenue 555-2048    68421879   3
##   4: 100000004           Dennis Doe 1100 Foobaz Avenue 555-2048    68421879   3
##   5: 100000006 Rebecca Carrannante     3 Orange Court 555-7601    46268734   1
##  ---
## 181:        NA                <NA>               <NA>     <NA>          NA   1
## 182:        NA                <NA>               <NA>     <NA>          NA   2
## 183:        NA                <NA>               <NA>     <NA>          NA   1
## 184:        NA                <NA>               <NA>     <NA>          NA   1
## 185:        NA                <NA>               <NA>     <NA>          NA   1
##      AppointmentID PrepNurse Physician               Start                 End
##   1:      76983231        NA         3 2008-04-26 12:00:00 2008-04-26 13:00:00
##   2:      46846589       103         4 2008-04-25 10:00:00 2008-04-25 11:00:00
##   3:      59871321        NA         4 2008-04-26 10:00:00 2008-04-26 11:00:00
##   4:      86213939       102         9 2008-04-27 10:00:00 2008-04-21 11:00:00
##   5:      33556494       101         9 2008-05-12 00:00:00 2008-05-12 00:32:00
##  ---
## 181:            NA        NA         7                <NA>                <NA>
## 182:            NA        NA         6                <NA>                <NA>
## 183:            NA        NA         7                <NA>                <NA>
## 184:            NA        NA         7                <NA>                <NA>
## 185:            NA        NA         3                <NA>                <NA>
##      ExaminationRoom Patient_SSN Cost
##   1:               C   100000001    0
##   2:               B   100000004    0
##   3:               C   100000004    0
##   4:               A   100000004    0
##   5:               B   100000006    0
##  ---
## 181:            <NA>   100000100 3750
## 182:            <NA>   100000101 3750
## 183:            <NA>   100000102 1500
## 184:            <NA>   100000103 1500
## 185:            <NA>   100000103 1500
```

```
# Making the department summary table by merging the affiliated_with table
department_summary <- merge(affiliated_with, department, by.x = 'Department', by.y='DepartmentID')
department_summary <- department_summary %>%
  select(Department, Name) %>%
  distinct(Department, Name) %>%
  rename(Department_Name = Name)

# Making EmployeeID-Physician name as referential keys
physician_summary <- merge(affiliated_with, physician, by.x = 'Physician', by.y='EmployeeID')
physician_summary <- physician_summary %>% select(Physician, Department, Name) %>% rename(Physician_Nam


# Make a big reference table with the Physician and Department summaries combined
physician_department_summary <- merge(physician_summary, department_summary, by='Department')
physician_department_summary
```

```
##     Department Physician    Physician_Name  Department_Name
##  1:          1         1       John Dorian General Medicine
##  2:          1         2       Elliot Reid General Medicine
##  3:          1         3  Christopher Turk General Medicine
##  4:          1         4      Percival Cox General Medicine
##  5:          1         5         Bob Kelso General Medicine
##  6:          1         7          John Wen General Medicine
##  7:          1         8 Keith Dudemeister General Medicine
##  8:          2         3  Christopher Turk          Surgery
##  9:          2         6      Todd Quinlan          Surgery
## 10:          2         7          John Wen          Surgery
## 11:          3         9       Molly Clock       Psychiatry
```

```
# I replace the column name Referral and Referring Physicians in this table
finaltable <- rawdata %>%
  left_join(physician_department_summary, by=c("PCP" = "Physician")) %>%
  rename('Referring_Physician' = Physician_Name) %>%
  left_join(physician_department_summary, by=c("Physician")) %>%
  mutate(Referral = Physician_Name) %>%
  select(Patient_SSN, Referral, Referring_Physician, Cost)

finaltable
```

```
##      Patient_SSN         Referral Referring_Physician Cost
##   1:   100000001 Christopher Turk         John Dorian    0
##   2:   100000001 Christopher Turk         John Dorian    0
##   3:   100000004     Percival Cox    Christopher Turk    0
##   4:   100000004     Percival Cox    Christopher Turk    0
##   5:   100000004     Percival Cox    Christopher Turk    0
##  ---
## 288:   100000102         John Wen         John Dorian 1500
## 289:   100000103         John Wen         John Dorian 1500
## 290:   100000103         John Wen         John Dorian 1500
## 291:   100000103 Christopher Turk         John Dorian 1500
## 292:   100000103 Christopher Turk         John Dorian 1500
```

```
#Get the aggregation in the number of shared patients and total shared patient's cost
num_shared_patients_costs <- finaltable %>%
  group_by(Referring_Physician, Referral) %>%
  summarise(shared_billing_costs= sum(Cost),Shared_patients = n_distinct(Patient_SSN)) %>% arrange(Refe
```

```
## ‘summarise()‘ has grouped output by ’Referring_Physician’. You can override using the ‘.groups‘ argu
```

```
# Adjusted_Affiliated with table grouping by physician
adjusted_affiliated_with <- affiliated_with %>%
  group_by(Physician) %>%
  filter(row_number() == 1) %>%
  select(Physician, Department)

# Making Department Code-Department as referential keys
adjusted_department_summary <- merge(adjusted_affiliated_with, department, by.x = 'Department', by.y='De

# Making EmployeeID-Physician name as referential keys
adjusted_physician_summary <- merge(adjusted_affiliated_with, physician, by.x='Physician', by.y='Employe
  rename(Physician_Name = Name) %>%
  select(Physician, Department, Physician_Name)


# Combining the both adjusted summaries together
adjusted_physician_deptartment_summary <- merge(adjusted_physician_summary, adjusted_department_summary
adjusted_physician_deptartment_summary<- adjusted_physician_deptartment_summary %>%
  select(Physician_Name, Department_Name)

#Combining the final tables
final_merged_result <- merge(num_shared_patients_costs, adjusted_physician_deptartment_summary, by.x='Re
  rename(Primary_Department=Department_Name) %>%
  merge(adjusted_physician_deptartment_summary, by.x='Referral', by.y='Physician_Name') %>% rename(Refe
  arrange(Referring_Physician, desc(Shared_patients)) %>%
  select(Referring_Physician, Referral, Primary_Department,  Referral_Department, Shared_patients, shar
final_merged_result
```

```
##    Referring_Physician         Referral Primary_Department Referral_Department
## 1     Christopher Turk      John Dorian   General Medicine    General Medicine
## 2     Christopher Turk      Elliot Reid   General Medicine    General Medicine
## 3     Christopher Turk      Molly Clock   General Medicine          Psychiatry
## 4     Christopher Turk     Percival Cox   General Medicine    General Medicine
## 5     Christopher Turk         John Wen   General Medicine    General Medicine
## 6     Christopher Turk     Todd Quinlan   General Medicine             Surgery
## 7          Elliot Reid      John Dorian   General Medicine    General Medicine
## 8          Elliot Reid Christopher Turk   General Medicine    General Medicine
## 9          Elliot Reid      Molly Clock   General Medicine          Psychiatry
## 10         Elliot Reid     Percival Cox   General Medicine    General Medicine
## 11         Elliot Reid         John Wen   General Medicine    General Medicine
## 12         Elliot Reid     Todd Quinlan   General Medicine             Surgery
## 13         John Dorian      Elliot Reid   General Medicine    General Medicine
## 14         John Dorian     Percival Cox   General Medicine    General Medicine
## 15         John Dorian      Molly Clock   General Medicine          Psychiatry
## 16         John Dorian Christopher Turk   General Medicine    General Medicine
## 17         John Dorian         John Wen   General Medicine    General Medicine
## 18         John Dorian     Todd Quinlan   General Medicine             Surgery
##    Shared_patients shared_billing_costs
## 1               17                    0
## 2               15                    0
## 3                9                    0
## 4                8                    0
## 5                4                33000
## 6                2                17298
## 7               18                    0
## 8                9                31500
## 9                9                    0
## 10               4                    0
## 11               2                15000
## 12               2                 7500
## 13              16                    0
## 14              12                    0
## 15              10                    0
## 16               8                27700
## 17               7                30050
## 18               1                 3750
```

Question Three

Which tool did you find it 'easiest' to use while completing these exercises? What advice would you give novice data wranglers when it comes to choosing between Excel, SQL, and R? Please make your answer either a different text colour, or bolded, when you knit this document so TA's can find it.

I found R to be the easiest while completing these exercises because R has a lot of built in functions/ packages that are necessary for me to clean data like removing uneccesary columns or null values.Excel has a lot of built in functions and I recommend to use the help function for all 3 platforms because the syntax can get messy when applying the same function to a specific column. When using Excel, I recommend to import some data and play around with the visual tools and some basic functions and queries necessary to transform and preprocess

data. When using SQL, I recommend to be familiar with the common keywords that are used to query data. Once they get a grasp on the common keywords, then they can move on to some complex queries. Finally, when using R, the help function really should help novice data wranglers be familiar with the packages that are necessary to read in data, query data, subset dataframes, filter out data frames, and remove unnecessary column. They should also have a basic understanding on the functions used to generate common visual plots as well.

Part Two

We are going download US Census data using the Census API. To start, you will need to request a key here: https://api.census.gov/data/key_signup.html.

We'll be using the following package:

A vignette demonstrating much of the functionality of this package can be found here https://walker-data.com/census-r/index.html

Start by setting your API key.

```
census_api_key("5c4e75b1d344c195de1c421444bb52400f92c18e")
```

```
## To install your API key for use in future sessions, run this function with ‘install = TRUE‘.
```

The function ‘get_acs()’ will download the American Community Survey (ACS) Census data. You will need to know the variable ID - and there are thousands of variables across the different files. To rapidly search for variables, use the commands ‘load_variables()’ and ‘View()’. We'll do this below:

```
v19 <- load_variables(2019, "acs5", cache = TRUE)
View(v19)
```

As you can see, there are many types of data avaiable to us in the census. In the View table, you can user filters to explore the kind of data that is available to you. For instance, try fitering by ‘income’ in the concept column.

The full metadata is available here https://www.socialexplorer.com/data/ACS2019_5yr/metadata/.

For now, we'll use the following:

```
newEngDat <- get_acs(geography = "new england city and town area",
              year = 2019,
              variables = c(popn = "B03002_001",
                            white = "B03002_003", blk = "B03002_004",
                            asn = "B03002_006", hisp = "B03002_012",
                            medHouseInc="B19013_001", hlthInsCov="B27001_001",
                            workPop="B08604_001",workTravel="B08013_001",
                            workHome="B08006_017", mthExp="B25088_001",
                            mthHousing="B25105_001"),
              survey = "acs5",
              output = "wide")
```

```
## Getting data from the 2015-2019 5-year ACS
```

In the above code, we specified the following arguments:

geography: The level of geography we want the data in **year:** The end year of the data (because we want 2015-2019, we use 2019). **variables:** The variables we want to bring in as specified in a vector you create using the function c(). Note that we created variable names of our own (e.g. "popn") and we put the ACS IDs in quotes ("B03002_001"). **survey:** The specific Census survey were extracting data from. We want data from the 5-year American Community Survey, so we specify "acs5". The ACS comes in 1-, 3-, and 5-year varieties. **output:** gives us a traditional dataset, alternatively "tidy" would give us a tibble.

See ?get_acs for more variables you could request.

We then have the following columns in our data:

GEOID: A unique ID variable of the geography **Name:** The Name of the geographic area **popn:** The total population **white:** The population of people who identify as white **blk:** The population of people who identify as black **asn:** The population of people who identify as asian **hisp:** The population of people who identify as hispanic **medHouseInc:** The median household income **hlthInsCov:** The population who have health insurance coverage **workPop:** The worker population **workTravel:** Aggregrate travel time to work, in minutes **workHome:** Number of workers who work from home **mthExp:** Median monthly cost of living estimate **mthHousing:** Median housing costs per month

You'll notice that there is an 'E' and an 'M' beside each of the column names in your dataset. The 'E' stands for estimate, and 'M' margin of error. While important, we will not be analyzing margins of error.

Question Four

Remove the margin of error columns, and then remove the 'E' from the end of the other column names.

```
# your code here



# Cols stores the columns with the M removed
cols <- c("GEOID", "NAME")

# This will store the updated columns with the Capital E removed at the end
revised_col <- c("GEOID", "NAME")
names_new_eng_dat <- names(newEngDat)



# I loop through starting the 3rd column and then first append the columns with E to a temporary vector

# cols and then I remove the "E, the last character of each column
for (i in 3: length(names_new_eng_dat)){
  if(str_sub(names_new_eng_dat[i], -1 ) == 'E'){
    cols <- append(cols, names_new_eng_dat[i])
    new_columns <- substr(names_new_eng_dat[i], 1, nchar(names_new_eng_dat[i]) - 1)
    revised_col <- append(revised_col, new_columns)
  }
}

revised_col
```

```
##  [1] "GEOID"       "NAME"        "popn"        "white"       "blk"
##  [6] "asn"         "hisp"        "medHouseInc" "hlthInsCov"  "workPop"
## [11] "workTravel"  "workHome"    "mthExp"      "mthHousing"
```

```
# Store the updated columns with 'M' removed at the end
newEngDat<- newEngDat[cols]


names(newEngDat) <- revised_col
newEngDat
```

```
## # A tibble: 40 x 14
##    GEOID NAME    popn  white    blk    asn   hisp medHouseInc hlthInsCov workPop
##    <chr> <chr>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>       <dbl>      <dbl>   <dbl>
##  1 70450 Atho~ 2.15e4 1.99e4    118    122    841       54442      21367    7201
##  2 70600 Augu~ 7.70e4 7.29e4    699    547   1144       56876      75655   42372
##  3 70750 Bang~ 1.33e5 1.24e5   1186   1446   1704       54158     131960   66185
##  4 70900 Barn~ 2.41e5 2.14e5   6658   3355   7657       73714     238685  111832
##  5 71050 Barr~ 4.31e4 4.07e4    433    277    652       60882      42649   26168
##  6 71350 Benn~ 2.23e4 2.09e4    137    140    504       54797      21844   11380
##  7 71500 Berl~ 1.53e4 1.35e4    640     86    745       45597      13418    6195
##  8 71650 Bost~ 4.95e6 3.48e6 374214 390064 558137       91213    4905130 2794666
##  9 71950 Brid~ 9.21e5 5.83e5  97391  48097 170245       94222     914158  450822
## 10 73050 Dove~ 1.41e5 1.28e5   1254   4212   3617       72636     140114   58321
## # ... with 30 more rows, and 4 more variables: workTravel <dbl>,
## #   workHome <dbl>, mthExp <dbl>, mthHousing <dbl>
```

Question 5

Which 10 communities have the largest proportion of their working population work from home?

```
# your code here

newEngDat2 <- newEngDat %>%
  mutate(workprop = workHome/workPop) %>%
  select(NAME, workprop) %>%
  arrange(desc(workprop))
head(newEngDat2,10)
```

```
## # A tibble: 10 x 2
##    NAME                                               workprop
##    <chr>                                                 <dbl>
##  1 Vineyard Haven, MA Micropolitan NECTA               0.118
##  2 Keene, NH Micropolitan NECTA                        0.0831
##  3 Greenfield Town, MA Micropolitan NECTA              0.0824
##  4 Barnstable Town, MA Metropolitan NECTA              0.0744
##  5 Bennington, VT Micropolitan NECTA                   0.0731
##  6 Sanford, ME Micropolitan NECTA                      0.0700
##  7 Brunswick, ME Micropolitan NECTA                    0.0676
##  8 Portland-South Portland, ME Metropolitan NECTA      0.0671
##  9 Leominster-Gardner, MA Metropolitan NECTA           0.0656
## 10 Worcester, MA-CT Metropolitan NECTA                 0.0631
```

Question 6

We'll define discretionary income as Income-Expenses. Right now, you have annual income and monthly expenses. Create a new column to calculate monthly discretionay income, and display the towns with the highest amounts of discretionary income.

```
# your code here
newEngDat$yearInc  <- newEngDat$medHouseInc / 12
newEngDat$monthly_disc_in <- newEngDat$yearInc - newEngDat$mthExp
newEngDat_new <- newEngDat %>%
  select(NAME, "Discretionary_Income" = monthly_disc_in) %>%
  arrange(desc(Discretionary_Income))
head(newEngDat_new, 10)
```

```
## # A tibble: 10 x 2
##    NAME                                                     Discretionary_Income
##    <chr>                                                                   <dbl>
##  1 Portsmouth, NH-ME Metropolitan NECTA                                    5648.
##  2 Danbury, CT Metropolitan NECTA                                          5644.
##  3 Bridgeport-Stamford-Norwalk, CT Metropolitan NECTA                      5619.
##  4 Boston-Cambridge-Newton, MA-NH Metropolitan NECTA                       5598.
##  5 Concord, NH Micropolitan NECTA                                          4868.
##  6 Hartford-East Hartford-Middletown, CT Metropolitan NECTA                4788.
##  7 Barnstable Town, MA Metropolitan NECTA                                  4661.
##  8 Manchester, NH Metropolitan NECTA                                       4650.
##  9 Lebanon, NH-VT Micropolitan NECTA                                       4631.
## 10 Worcester, MA-CT Metropolitan NECTA                                     4573.
```

Question 7

Which 5 towns have the largest proportional gaps in healthcare coverage?

```
# your code here
newEngDat$prop_gap <- (newEngDat$popn - newEngDat$hlthInsCov) / newEngDat$popn
newEngDatprop <- newEngDat %>%
  select("Town Names" = NAME, "Proportional_Gap" = prop_gap) %>%
  arrange(desc(Proportional_Gap))
head(newEngDatprop,5)
```

```
## # A tibble: 5 x 2
##   ‘Town Names‘                                          Proportional_Gap
##   <chr>                                                            <dbl>
## 1 Berlin, NH Micropolitan NECTA                                   0.120
## 2 Norwich-New London-Westerly, CT-RI Metropolitan NECTA           0.0443
## 3 Concord, NH Micropolitan NECTA                                  0.0372
## 4 Claremont, NH Micropolitan NECTA                                0.0236
## 5 Bennington, VT Micropolitan NECTA                               0.0201
```

Question 8

The divesity index of a geographic area is the probability that two people selected at random will be the same race. Create a function which will sample from the reported ethnic population in each geographic area and return the diversity index. Display the top 5 diverse towns.

```
# your code here

diversity = function(area){
  newEngDat$whitepropind <- (newEngDat$white / newEngDat$popn) ** 2
  newEngDat$blackpropind <- (newEngDat$blk / newEngDat$popn) **2
  newEngDat$asianpropind <- (newEngDat$asn / newEngDat$popn) **2
  newEngDat$hispanicpropind <- (newEngDat$hisp / newEngDat$popn) **2
  total_index <- newEngDat$whitepropind  + newEngDat$blackpropind+ newEngDat$asianpropind + newEngDat$hi
  return(total_index)
}

head(diversity(newEngDat$NAME),5)
```

```
## [1] 0.8572081 0.8985311 0.8683063 0.7891973 0.8947059
```

Question 9

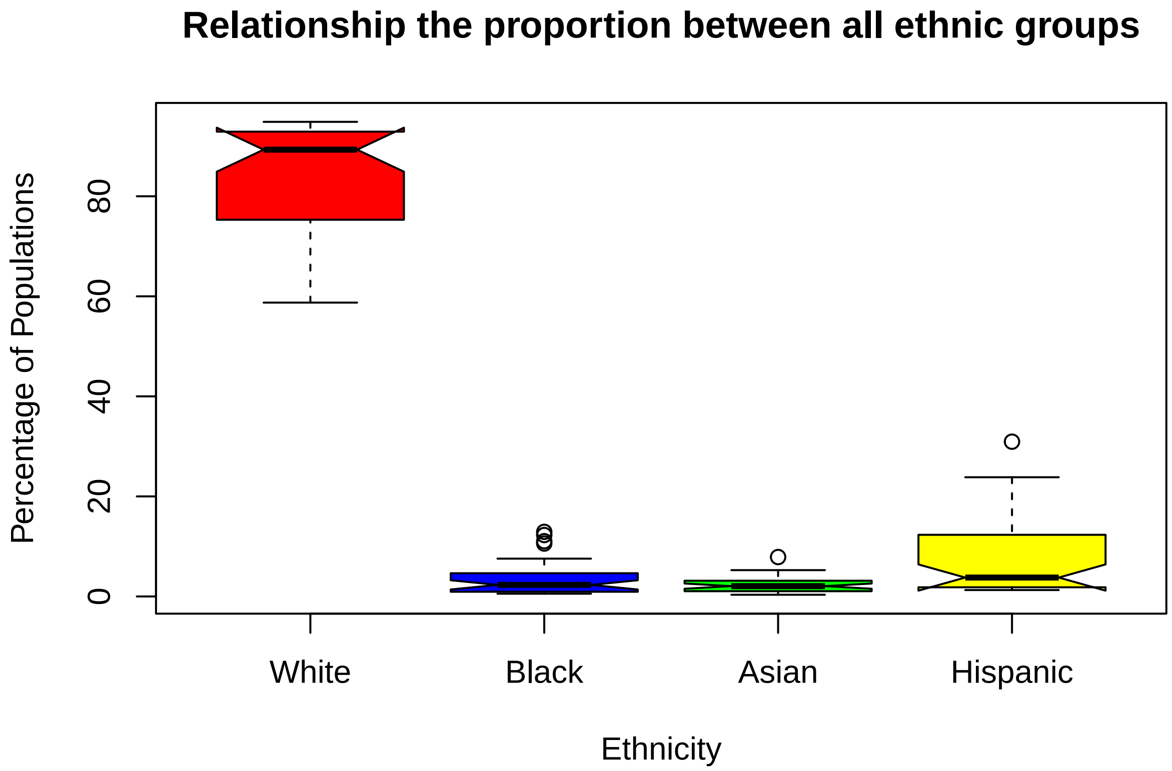
Convert the ethnicity columns to be percentages. Make a boxplot where each ethnicity is represented on the x-axis, and percent is on the y-axis. Points will be awarded for 'prettier' plots!

```
#your code here
newEngDat$whiteprop <- (newEngDat$white / newEngDat$popn) * 100
newEngDat$blackprop <- (newEngDat$blk / newEngDat$popn) * 100
newEngDat$asianprop <- (newEngDat$asn / newEngDat$popn) * 100
newEngDat$hispanicprop <- (newEngDat$hisp / newEngDat$popn) * 100
newEngDat_Eth <- newEngDat %>%
  select(whiteprop, blackprop, asianprop, hispanicprop)



library(ggplot2)
library(lattice)
require(reshape2)
```

```
## Loading required package: reshape2

##
## Attaching package: 'reshape2'

## The following objects are masked from 'package:data.table':
##
##     dcast, melt

## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
boxplot(newEngDat$whiteprop, newEngDat$blackprop, newEngDat$asianprop, newEngDat$hispanicprop, xlab = '
```

```
## Warning in (function (z, notch = FALSE, width = NULL, varwidth = FALSE, : some
## notches went outside hinges ('box'): maybe set notch=FALSE
```



Question 10

Ask a question of your choosing. Output both the head of a table, and a simple plot answering your question. Feel free to use the API to import extra variables that may be of interest to you.

Question: Display the top 10 names with the highest poverty rate and make a simple plot establishing the relationship between poverty proportion and working proportion where poverty proportion is on the x axis and working proportion is on the Y axis

```
# I only select clumns that are necessary to answer my own question. I did not remove the E at the end
newEngDataq10 <- get_acs(geography = "new england city and town area",
              year = 2019,
              variables = c(popn = "B03002_001",
                            pov = "B17023_001",
                            hlthInsCov="B27001_001",
                            workPop="B08604_001"),
              survey = "acs5",
              output = "wide")
```

```
## Getting data from the 2015-2019 5-year ACS
```

```
newEngDataq10
```

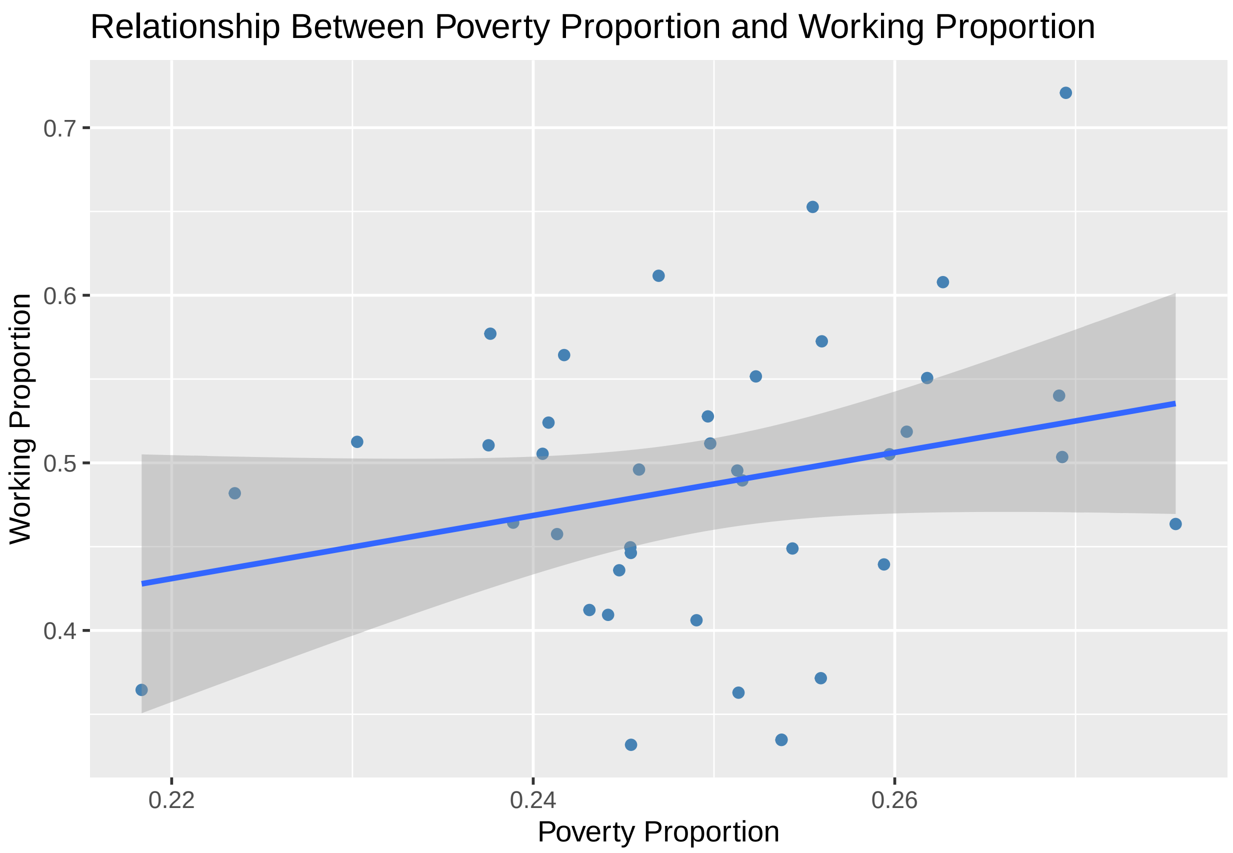
```
## # A tibble: 40 x 10
##    GEOID NAME          popnE popnM   povE  povM hlthInsCovE hlthInsCovM workPopE
##    <chr> <chr>         <dbl> <dbl>  <dbl> <dbl>       <dbl>       <dbl>    <dbl>
##  1 70450 Athol, MA M~ 2.15e4   233 5.46e3   257       21367         240     7201
##  2 70600 Augusta, ME~ 7.70e4   455 2.01e4   580       75655         494    42372
##  3 70750 Bangor, ME ~ 1.33e5   424 3.28e4   725      131960         455    66185
##  4 70900 Barnstable ~ 2.41e5    36 6.65e4  1114      238685         327   111832
##  5 71050 Barre, VT M~ 4.31e4   311 1.13e4   370       42649         367    26168
##  6 71350 Bennington,~ 2.23e4    75 5.30e3   261       21844         195    11380
##  7 71500 Berlin, NH ~ 1.53e4   131 3.80e3   186       13418         280     6195
##  8 71650 Boston-Camb~ 4.95e6   384 1.20e6  5068     4905130        1209  2794666
##  9 71950 Bridgeport-~ 9.21e5   195 2.32e5  1932      914158         549   450822
## 10 73050 Dover-Durha~ 1.41e5    39 3.44e4   802      140114         295    58321
## # ... with 30 more rows, and 1 more variable: workPopM <dbl>
```

```
newEngDat_q10 <- newEngDataq10 %>%
  mutate(PovProp = povE/popnE)%>%
  mutate(workprop = workPopE / popnE) %>%
  select(NAME, PovProp, workprop) %>%
  arrange(desc(PovProp))

head(newEngDat_q10,10)
```

```
## # A tibble: 10 x 3
##    NAME                                                  PovProp workprop
##    <chr>                                                   <dbl>    <dbl>
##  1 Barnstable Town, MA Metropolitan NECTA                  0.276    0.464
##  2 Portsmouth, NH-ME Metropolitan NECTA                    0.269    0.721
##  3 Laconia, NH Micropolitan NECTA                          0.269    0.503
##  4 Brunswick, ME Micropolitan NECTA                        0.269    0.540
##  5 Barre, VT Micropolitan NECTA                            0.263    0.608
##  6 Augusta, ME Micropolitan NECTA                          0.262    0.551
##  7 Norwich-New London-Westerly, CT-RI Metropolitan NECTA   0.261    0.519
##  8 Pittsfield, MA Metropolitan NECTA                       0.260    0.505
##  9 Lewiston-Auburn, ME Metropolitan NECTA                  0.259    0.439
## 10 Rutland, VT Micropolitan NECTA                          0.256    0.573
```

```
ggplot(newEngDat_q10 ,
       aes(x = PovProp,
           y = workprop)) +
  geom_point(color= "steelblue") +
  geom_smooth(method = "lm") + labs(x = "Poverty Proportion", y = "Working Proportion") + ggtitle("Rela
```

```
## ‘geom_smooth()‘ using formula ’y ~ x’
```


Relationship Between Poverty Proportion and Working Proportion
Working Proportion
0.7
0.6
0.5
0.4
0.22
0.24
0.26
Poverty Proportion