Project 06 Answer Key

https://datamine.purdue.edu/seminars/fall2019/stat19000project6.html

Question 1a

Read in the `5000_transactions.csv` data (from 8451) into a data frame to be called `myDF`.

```
# Use read.csv() to bring in the transaction data
myDF = read.csv('/class/datamine/data/8451/The_Complete_Journey_2_Master/5000_t
ransactions.csv')
```

Question 1b

Split the data frame `myDF`, using the `STORE_R` column, and store the results of the split into a new variable called `myresults`. Use the split command to achieve this. Remember that we can read about the split command using: `?split`

```
# Use split() along STORE_R to create a new data frame
myresults = split(myDF, myDF$STORE_R)
```

Question 1c

What is the class of `myresults`? What is the length of `myresults`? What are the names of `myresults`? (Use class, length, and names on myresults.)

```
# Use class() to view the data type
# Use length() to view the data length
# Use names() to view the data field names
class(myresults)
length(myresults)
names(myresults)
```

```
>>>
> class(myresults)
[1] "list"
> length(myresults)
[1] 4
> names(myresults)
[1] "CENTRAL" "EAST   " "SOUTH  " "WEST   "
```

Question 1d

Check the dimensions (`dim`) and the head of `myresults[["CENTRAL"]]`.

```
# Use dim() to see the dimension of the object
dim(myresults[["CENTRAL"]])
# Use head() to see the first 6 rows of the object
head(myresults[["CENTRAL"]])
```

```
>>>
> dim(myresults[["CENTRAL"]])
[1] 2463343       9
> head(myresults[["CENTRAL"]])
   BASKET_NUM HSHD_NUM PURCHASE_ PRODUCT_NUM SPEND UNITS STORE_R
   13        462      807 03-JAN-16      208846  3.99     1 CENTRAL
   15        591      999 03-JAN-16       93067  2.00     1 CENTRAL
   20        834      907 04-JAN-16     5423151  1.88     1 CENTRAL
   25       1424     4231 05-JAN-16     5180739  1.67     1 CENTRAL
   26       1494     1944 03-JAN-16     4667776  1.99     1 CENTRAL
   30       1583     3135 03-JAN-16       95584  1.00     1 CENTRAL
      WEEK_NUM YEAR
      13        1 2016
      15        1 2016
      20        1 2016
      25        1 2016
      26        1 2016
      30        1 2016
```

Question 1e

Now manually make a data frame that has all of the same columns as `myDF` but only has rows for which `myDF$STORE_R` is equal to `"CENTRAL"`: `centralresults <- myDF[myDF$STORE_R == "CENTRAL", ]` Verify that the `dim` and `head` of `myresults[["CENTRAL"]]` and `centralresults` look the same.

```
# Use square brackets to get myDF rows where the value of STORE_R is "CENTRAL"
centralresults = myDF[myDF$STORE_R == "CENTRAL", ]
# Use dim() to see the dimension of the object
dim(centralresults)
# Use head() to see the first 6 rows of the object
head(centralresults)
```

```
>>>
> dim(centralresults)
[1] 2463343       9
> head(centralresults)
   BASKET_NUM HSHD_NUM PURCHASE_ PRODUCT_NUM SPEND UNITS STORE_R
   13        462      807 03-JAN-16      208846  3.99     1 CENTRAL
   15        591      999 03-JAN-16       93067  2.00     1 CENTRAL
   20        834      907 04-JAN-16     5423151  1.88     1 CENTRAL
   25       1424     4231 05-JAN-16     5180739  1.67     1 CENTRAL
   26       1494     1944 03-JAN-16     4667776  1.99     1 CENTRAL
   30       1583     3135 03-JAN-16       95584  1.00     1 CENTRAL
      WEEK_NUM YEAR
      13        1 2016
      15        1 2016
      20        1 2016
      25        1 2016
      26        1 2016
      30        1 2016
```

Question 2a

Read in the `5000_products.csv` data (from 8451) into a data frame to be called `myproducts`.

```
myproducts = read.csv('/class/datamine/data/8451/The_Complete_Journey_2_Master/
5000_products.csv')
```

Question 2b

Merge the data frames `myDF` and `myproducts`, according to the `"PRODUCT_NUM"` column (which is common to both data frames). Store the results of the merge into a new variable called `mybigDF`. Remember that we can read about the merge command using: `?merge`. Hint: You can use `by="PRODUCT_NUM"`

```
# Use merge() to combine two data frames on PRODUCT_NUM
mybigDF = merge(myDF, myproducts, "PRODUCT_NUM")
```

Question 3a

Take a subset of the data frame `myDF` that shows all of data about the purchases made on 23 December 2017. You do not need to store the results of the subset function anywhere. Remember that we can read about the subset command using: `?subset`

```
# Use subset() to get rows of myDF where the value of PURCHASE_ is '23-DEC-17'
head(subset(myDF, myDF$PURCHASE_=='23-DEC-17'))
```

```
>>>
> head(subset(myDF, myDF$PURCHASE_=='23-DEC-17'))
      BASKET_NUM HSHD_NUM PURCHASE_ PRODUCT_NUM SPEND UNITS
      19628     102294     1048 23-DEC-17     5819718  3.99     1
      19629     102296      728 23-DEC-17     4433700  3.19     1
      19630     102296     4260 23-DEC-17     5484136  1.19     1
      19631     102300     1200 23-DEC-17      318095  2.00     3
      19632     102301     1486 23-DEC-17     3987776  1.25     1
      19633     102304     2310 23-DEC-17      765373  1.79     1
            STORE_R WEEK_NUM YEAR
            19628 CENTRAL      103 2017
            19629 CENTRAL      103 2017
            19630 SOUTH        103 2017
            19631 EAST         103 2017
            19632 WEST         103 2017
            19633 EAST         103 2017
```

Question 3b

Take a subset of the data frame `myDF` that shows only the dollar amounts of the purcases made on 23 December 2017.

```
# Use subset() to get rows of myDF where the value of PURCHASE_ is '23-DEC-17'
# Set the 'select' parameter to 'SPEND' to include that field in the output
head(subset(myDF, myDF$PURCHASE_=='23-DEC-17', select='SPEND'))
```

```
>>>
> head(subset(myDF, myDF$PURCHASE_=='23-DEC-17', select='SPEND'))
      SPEND
      19628  3.99
      19629  3.19
      19630  1.19
      19631  2.00
      19632  1.25
      19633  1.79
```

Question 3c

Take a subset of the data frame `myDF` that shows only the dates and dollar amounts of the purcases made on 23 December 2017.

```
# Use subset() to get rows of myDF where the value of PURCHASE_ is '23-DEC-17'
# Set the 'select' parameter to a vector containing 'PURCHASE_' and 'SPEND' to
# include those fields in the output
head(subset(myDF, myDF$PURCHASE_=='23-DEC-17', select=c('PURCHASE_', 'SPEND')))
```

```
>>>
> head(subset(myDF, myDF$PURCHASE_=='23-DEC-17', select=c('PURCHASE_', 'SPEND')
))
      PURCHASE_ SPEND
      19628 23-DEC-17  3.99
      19629 23-DEC-17  3.19
      19630 23-DEC-17  1.19
      19631 23-DEC-17  2.00
      19632 23-DEC-17  1.25
      19633 23-DEC-17  1.79
```

Question 3d

Take a subset of the data frame `myDF` that shows only the dates and dollar amounts and stores of the purcases made on 23 December 2017.

```
# Use subset() to get rows of myDF where the value of PURCHASE_ is '23-DEC-17'
# Set the 'select' parameter to a vector containing 'PURCHASE_', 'SPEND', and
# 'STORE_R' to include those fields in the output
head(subset(myDF, myDF$PURCHASE_=='23-DEC-17', select=c('PURCHASE_', 'SPEND',
'STORE_R')))
```

```
>>>
> head(subset(myDF, myDF$PURCHASE_=='23-DEC-17', select=c('PURCHASE_', 'SPEND',
'STORE_R')))
      PURCHASE_ SPEND STORE_R
      19628 23-DEC-17  3.99 CENTRAL
      19629 23-DEC-17  3.19 CENTRAL
      19630 23-DEC-17  1.19 SOUTH
      19631 23-DEC-17  2.00 EAST
      19632 23-DEC-17  1.25 WEST
      19633 23-DEC-17  1.79 EAST
```

Question 3e

On December 23, 2017, which store had the largest total amount (in dollars) of purchases? Hint: Use the `tapply` function.

```
# Store the previous code in a variable
myDFdec23 = subset(myDF, myDF$PURCHASE_=='23-DEC-17', select=c('SPEND', 'STORE_
R'))
# Use tapply() to calculate the total purchases for each store
spend_by_store = tapply(myDFdec23$SPEND, myDFdec23$STORE_R, sum)
# Use sort() to see the store with the highest total dollar purchase amount
sort(spend_by_store)
```

```
>>>
> sort(spend_by_store)
 SOUTH    CENTRAL  WEST     EAST
 22409.95 23899.64 26020.28 36078.54
```