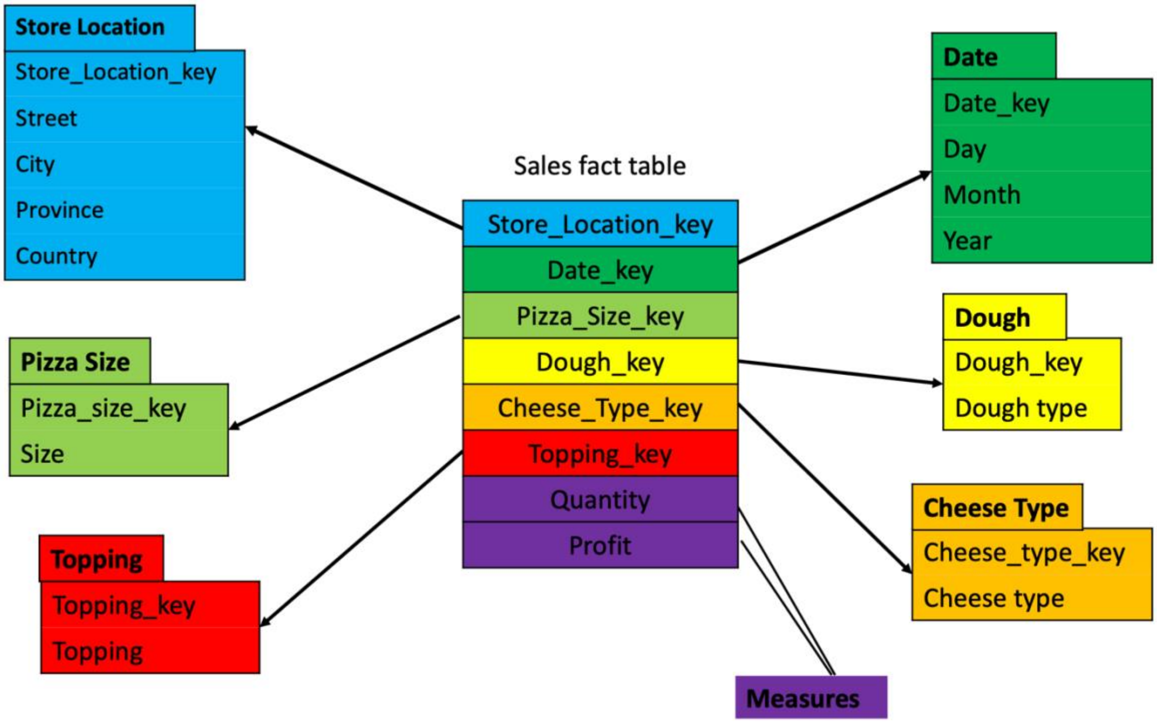
Assignment 2

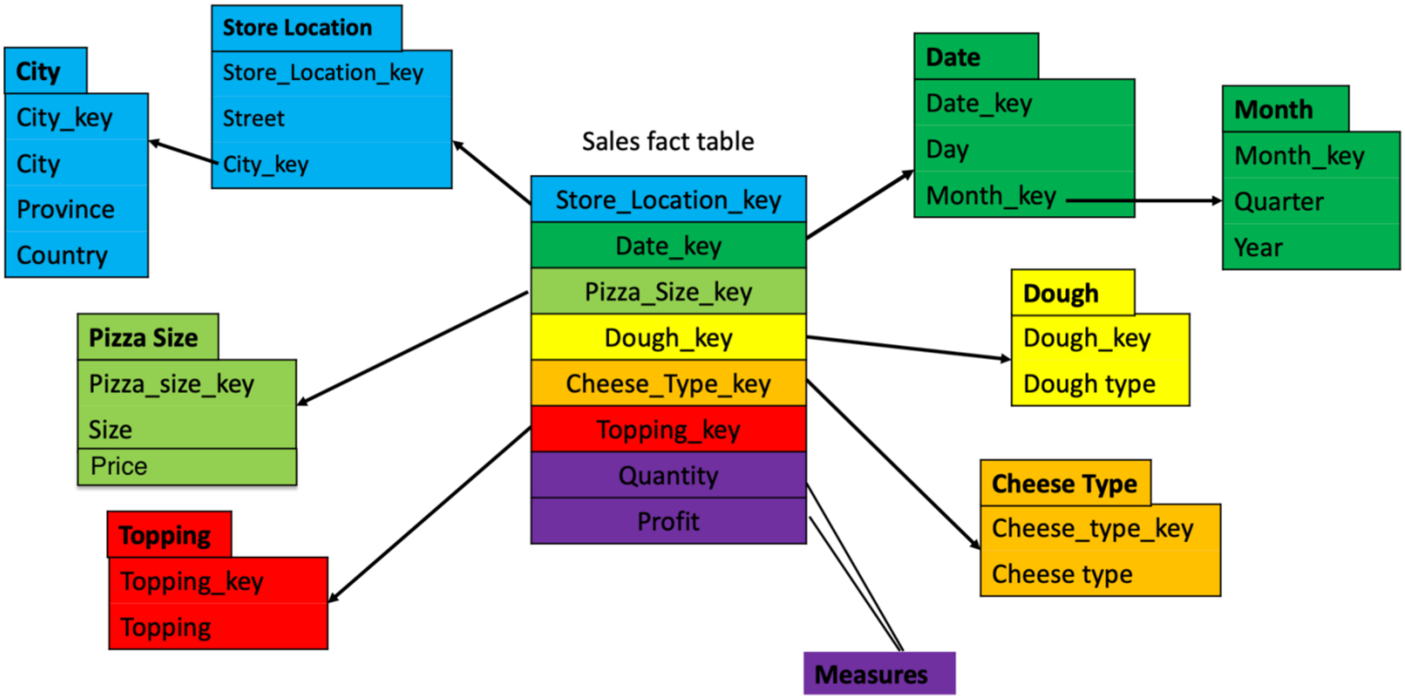
Alireza Houshidari, Reza Alikhani

Part 1

1-a Star Schema.



1-B Snowflake Schema.



1-C In this section, tables were built and CSV files created. Also, 500 samples generated.

```
store_table <-
  data.frame(key=c("Bank", "Younge", "Papineau", "Boundary", "Bertrand"),
             Street=c("Bank St", "Younge St", "Rue Papineau", "Boundray Road", "Rue
Bertrand"),
             city=c("OT", "TR", "Mo", "Va", "QU"))

city_info <-
  data.frame(
    city=c("OT", "TR", "MO", "VA", "QU"),
    name=c("Ottawa", "Toronto", "Montreal", "Vancouver", "Quebec City"),
    country=c("Canada", "Canada", "Canada", "Canada", "Canada")
  )

date_table <-
  data.frame(key=1:12,
             month=c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12),
             quarter=c("Q1","Q1","Q1","Q2","Q2","Q2","Q3","Q3","Q3","Q4","Q4","Q4"),
             year=c(2022, 2023))

key=c("personal", "small", "medium", "large", "xlarge")
size_table <-data.frame(key=factor(x=key,levels=c("personal", "small", "medium", "large",
"xlarge"),
                                   ordered=TRUE),
                        price =c(7, 9, 11, 13, 15))

dough_table <-
  data.frame(key=c("thin", "regular", "stuffed crust"))

cheese_table <-
  data.frame(key=c("mozzarella", "cheddar", "Goda"))

topping_table <-
  data.frame(key=c("Pepperoni", "Tomato", "Bacon", "mushroom"))

gen_orders <- function (no_recs) {
  OrderID <- 1:no_recs
  store <- sample(store_table$key, no_recs, replace = TRUE)
  time_year <- sample(c(2022, 2023), no_recs, replace = TRUE, prob = c(1,     1.7))
  time_month <- sample(date_table$month, no_recs, replace = TRUE, prob = c(1,       1, 1,
1, 1, 1,
                                                                           1,       1, 1,
1, 1, 1))
  size <- sample(size_table$key, no_recs, replace = TRUE, prob = c(1, 1, 1,   1.5, 2))
  dough <- sample(dough_table$key, no_recs, replace = TRUE)
  cheese <- sample(cheese_table$key, no_recs, replace = TRUE)
  topping <- sample(topping_table$key, no_recs, replace = TRUE)
  quantity <- sample(1:5, no_recs, replace = TRUE, prob = c(5, 2, 1, 1, 1))
  profit <- quantity*size_table[size, ]$price
```

```r
  orders <- data.frame(month=time_month,
                      year=time_year,
                      loc=store,
                      size=size,
                      dough=dough,
                      cheese=cheese,
                      topping=topping,
                      quantity=quantity,
                      profit=profit)

  # Sort the records by time order
  orders <- orders[order(orders$year, orders$month),]
  row.names(orders) <- NULL
  return(orders)

}

orders_fact <- gen_orders(1000)
head(orders_fact)

##   month year      loc     size         dough     cheese   topping quantity
## 1     1 2022   Younge personal       regular    cheddar    Tomato        2
## 2     1 2022     Bank   medium stuffed crust       Goda Pepperoni        3
## 3     1 2022 Bertrand   xlarge       regular       Goda     Bacon        2
## 4     1 2022 Boundary    small       regular    cheddar Pepperoni        2
## 5     1 2022 Boundary personal          thin mozzarella    Tomato        5
## 6     1 2022 Papineau   xlarge stuffed crust       Goda    Tomato        3
##   profit
## 1     14
## 2     33
## 3     30
## 4     18
## 5     35
## 6     45

write.csv(store_table, "store_location.csv", row.names = FALSE)
write.csv(city_info, "city_info.csv", row.names = FALSE)
write.csv(size_table, "pizza_size.csv", row.names = FALSE)
write.csv(dough_table, "dough.csv", row.names = FALSE)
write.csv(cheese_table, "cheese_type.csv", row.names = FALSE)
write.csv(topping_table, "topping.csv", row.names = FALSE)
write.csv(orders_fact, "orders.csv", row.names = FALSE)
```

2- In this section, OLAP cube got created and named revenue_cube.

```r
revenue_cube <-
    tapply(orders_fact$profit,
           orders_fact[,c("size", "month", "year", "loc")],
           function(x){return(sum(x))})

revenue_cube
```

```
## , , year = 2022, loc = Bank
## 
##           month
## size        1  2   3  4   5   6  7  8  9  10 11 12
##   personal 35  7  21 21   7  NA NA 14 42  NA NA 21
##   small    18  9 135 18  NA 144 27 27 NA   9 18 18
##   medium   66 NA  66 NA  NA  33 NA 11 99  NA 44 NA
##   large    13 65  65 65  NA 156 13 65 52  26 26 13
##   xlarge   30 15  45 NA 135  15 15 NA 30 105 45 45
## 
## , , year = 2023, loc = Bank
## 
##           month
## size        1   2   3   4   5   6  7   8   9 10  11 12
##   personal  7  21  NA   7  21   7 28  35  21 21  35 21
##   small    45   9  18  27  27  NA 99  18  54 NA  45 27
##   medium   NA 121  NA  11  55  NA NA  11  NA 33  99 55
##   large    52 104 143  NA 104 156 13  65  39 26  13 91
##   xlarge   30  30  45 180  90  45 NA 195 105 90 135 15
## 
## , , year = 2022, loc = Bertrand
## 
##           month
## size        1  2   3   4  5   6  7   8  9 10  11 12
##   personal NA NA  NA   7 NA  21 NA  21 NA NA  21  7
##   small    NA NA  NA  NA  9  NA NA  NA 54 36  NA NA
##   medium   22 77  NA  NA 22  11 NA  NA 44 NA  NA 33
##   large    26 26 104  NA 39  39 52  13 NA 39  65 NA
##   xlarge   45 NA 195 120 45 135 15 180 45 30 300 60
## 
## , , year = 2023, loc = Bertrand
## 
##           month
## size        1   2  3   4   5   6  7  8   9  10 11 12
##   personal 28  14 28  14  42  84 NA 42  14  NA 14 70
##   small    NA  36 NA   9  54  45 36 36  27  63  9 18
##   medium   22  55 66  NA  77  11 33 66  77  44 77 33
##   large    NA 156 13  13 169  26 26 78 117  NA 65 13
##   xlarge   60 105 75 180 120 105 75 75  75 240 45 NA
## 
## , , year = 2022, loc = Boundary
## 
##           month
## size        1   2  3  4  5   6  7   8  9  10 11  12
##   personal 35  35 14 28 21  NA  7  21 NA   7 28  28
##   small    36  54 NA 99 18  NA NA  18 NA  36 36   9
##   medium   NA  33 NA NA 55  NA 22  77 NA  NA 22  NA
##   large    NA  39 13 NA 13  39 NA  13 52  91 13 130
##   xlarge   15 150 75 45 NA 120 90 150 60 150 15  15
## 
## , , year = 2023, loc = Boundary
## 
##           month
## size         1  2  3  4   5   6   7   8   9 10 11 12
##   personal   7  7 35 42  49  63  NA  21  28 42 14 NA
##   small      9 36 18  9  45  18  45  63  45  9 99 54
##   medium    NA NA 33 44  11  NA  44  22  44 33 NA NA
##   large    169 NA 78 39  13  39  65 130 143 65 78 NA
##   xlarge   120 45 75 NA 315 105 150  75 105 15 NA 75
## 
## , , year = 2022, loc = Papineau
## 
##           month
## size          1  2  3  4   5  6  7   8  9 10 11 12
##   personal  NA  7 35 14  NA  7 28  NA NA 28 NA NA
##   small       9 18 NA NA  36 45 99 108 63 36 27 NA
##   medium     NA NA 99 NA  33 NA 33  22 99 NA NA 22
##   large      13 26 52 26  13 13 NA 104 NA 65 13 65
##   xlarge    150 45 15 60 105 60 75  60 30 60 15 30
## 
## , , year = 2023, loc = Papineau
## 
##           month
## size        1   2  3  4   5   6   7   8  9  10  11  12
##   personal 56  28  7 49   7  28  21  NA 49  28  21  28
##   small    NA  NA NA  9  18  45   9  45 18  NA  27   9
##   medium   22  NA 11 55  22  55  22  22 55  22  55  33
##   large    78  NA 26 52  65  78  26  13 52  78  39 312
##   xlarge   15 150 90 75 105 105 135 195 NA 135 270  60
## 
## , , year = 2022, loc = Younge
## 
##           month
## size        1   2  3  4  5   6   7  8  9  10 11 12
##   personal 14  70 21 NA NA  NA  14 NA 14   7 28 NA
##   small    NA  54 NA  9  9   9  18 81 NA  36 27 NA
##   medium   44  11 11 11 NA  11  NA 44 11  11 NA 11
##   large    65  13 91 13 NA  NA  26 26 65  13 26 13
##   xlarge   15 240 15 15 15 150 150 15 30 105 75 60
## 
## , , year = 2023, loc = Younge
## 
##           month
## size        1   2   3   4  5   6   7  8   9 10  11 12
##   personal 21   7  49  NA 14   7  56 42  77 42  NA 28
##   small    36  45  45  45 27  45   9 27   9 36   9  9
##   medium   44  22 132  66 33  33  55 22  33 44  66 22
##   large    13  26  78 130 39  39  13 13 143 39 182 39
##   xlarge   45 120 150 105 60 195 240 90  45 75 195 75
```

3- First, lets do a drill down operation to see if it helps.

```
apply(revenue_cube, c("year", "month", "size"),
      FUN=function(x) {return(sum(x, na.rm=TRUE))})
```

```
## , , size = personal
## 
```

```
##       month
## year     1   2   3   4   5   6   7   8   9  10 11  12
##   2022  84 119  91  70  28  28  49  56  56  42 77  56
##   2023 119  77 119 112 133 189 105 140 189 133 84 147
## 
## , , size = small
## 
##       month
## year    1   2   3   4   5   6   7   8   9  10  11  12
##   2022 63 135 135 126  72 198 144 234 117 153 108  27
##   2023 90 126  81  99 171 153 198 189 153 108 189 117
## 
## , , size = medium
## 
##       month
## year     1   2   3   4   5  6   7   8   9  10  11  12
##   2022 132 121 176  11 110 55  55 154 253  11  66  66
##   2023  88 198 242 176 198 99 154 143 209 176 297 143
## 
## , , size = large
## 
##       month
## year     1   2   3   4   5   6   7   8   9  10  11  12
##   2022 117 169 325 104  65 247  91 221 169 234 143 221
##   2023 312 286 338 234 390 338 143 299 494 208 377 455
## 
## , , size = xlarge
## 
##       month
## year     1   2   3   4   5   6   7   8   9  10  11  12
##   2022 255 450 345 240 300 480 345 405 195 450 450 210
##   2023 270 450 435 540 690 555 600 630 330 555 645 225
```

```
revenue_cube
```

```
## , , year = 2022, loc = Bank
## 
##           month
## size        1  2   3  4   5   6  7  8  9  10 11 12
##   personal 35  7  21 21   7  NA NA 14 42  NA NA 21
##   small    18  9 135 18  NA 144 27 27 NA   9 18 18
##   medium   66 NA  66 NA  NA  33 NA 11 99  NA 44 NA
##   large    13 65  65 65  NA 156 13 65 52  26 26 13
##   xlarge   30 15  45 NA 135  15 15 NA 30 105 45 45
## 
## , , year = 2023, loc = Bank
## 
##           month
## size        1   2   3   4   5   6  7   8   9 10  11 12
##   personal  7  21  NA   7  21   7 28  35  21 21  35 21
##   small    45   9  18  27  27  NA 99  18  54 NA  45 27
##   medium   NA 121  NA  11  55  NA NA  11  NA 33  99 55
##   large    52 104 143  NA 104 156 13  65  39 26  13 91
##   xlarge   30  30  45 180  90  45 NA 195 105 90 135 15
## 
## , , year = 2022, loc = Bertrand
```

```
##
##           month
## size        1  2   3   4  5   6  7   8  9 10  11 12
##   personal NA NA  NA   7 NA  21 NA  21 NA NA  21  7
##   small    NA NA  NA  NA  9  NA NA  NA 54 36  NA NA
##   medium   22 77  NA  NA 22  11 NA  NA 44 NA  NA 33
##   large    26 26 104  NA 39  39 52  13 NA 39  65 NA
##   xlarge   45 NA 195 120 45 135 15 180 45 30 300 60
##
## , , year = 2023, loc = Bertrand
##
##           month
## size        1   2  3   4   5   6  7  8   9  10 11 12
##   personal 28  14 28  14  42  84 NA 42  14  NA 14 70
##   small    NA  36 NA   9  54  45 36 36  27  63  9 18
##   medium   22  55 66  NA  77  11 33 66  77  44 77 33
##   large    NA 156 13  13 169  26 26 78 117  NA 65 13
##   xlarge   60 105 75 180 120 105 75 75  75 240 45 NA
##
## , , year = 2022, loc = Boundary
##
##           month
## size        1   2  3  4  5   6  7   8  9  10 11  12
##   personal 35  35 14 28 21  NA  7  21 NA   7 28  28
##   small    36  54 NA 99 18  NA NA  18 NA  36 36   9
##   medium   NA  33 NA NA 55  NA 22  77 NA  NA 22  NA
##   large    NA  39 13 NA 13  39 NA  13 52  91 13 130
##   xlarge   15 150 75 45 NA 120 90 150 60 150 15  15
##
## , , year = 2023, loc = Boundary
##
##           month
## size          1  2  3  4   5   6   7   8   9 10 11 12
##   personal   7  7 35 42  49  63  NA  21  28 42 14 NA
##   small      9 36 18  9  45  18  45  63  45  9 99 54
##   medium    NA NA 33 44  11  NA  44  22  44 33 NA NA
##   large    169 NA 78 39  13  39  65 130 143 65 78 NA
##   xlarge   120 45 75 NA 315 105 150  75 105 15 NA 75
##
## , , year = 2022, loc = Papineau
##
##           month
## size          1  2  3  4   5  6  7   8  9 10 11 12
##   personal  NA  7 35 14  NA  7 28  NA NA 28 NA NA
##   small      9 18 NA NA  36 45 99 108 63 36 27 NA
##   medium    NA NA 99 NA  33 NA 33  22 99 NA NA 22
##   large     13 26 52 26  13 13 NA 104 NA 65 13 65
##   xlarge   150 45 15 60 105 60 75  60 30 60 15 30
##
## , , year = 2023, loc = Papineau
##
##           month
## size        1   2  3  4   5   6   7   8  9  10  11  12
##   personal 56  28  7 49   7  28  21  NA 49  28  21  28
##   small    NA  NA NA  9  18  45   9  45 18  NA  27   9
##   medium   22  NA 11 55  22  55  22  22 55  22  55  33
##   large    78  NA 26 52  65  78  26  13 52  78  39 312
##   xlarge   15 150 90 75 105 105 135 195 NA 135 270  60
##
## , , year = 2022, loc = Younge
##
##           month
## size        1   2  3  4  5   6   7  8  9  10 11 12
##   personal 14  70 21 NA NA  NA  14 NA 14   7 28 NA
##   small    NA  54 NA  9  9   9  18 81 NA  36 27 NA
##   medium   44  11 11 11 NA  11  NA 44 11  11 NA 11
##   large    65  13 91 13 NA  NA  26 26 65  13 26 13
##   xlarge   15 240 15 15 15 150 150 15 30 105 75 60
##
## , , year = 2023, loc = Younge
##
##           month
## size        1   2   3   4  5   6   7  8   9 10  11 12
##   personal 21   7  49  NA 14   7  56 42  77 42  NA 28
##   small    36  45  45  45 27  45   9 27   9 36   9  9
##   medium   44  22 132  66 33  33  55 22  33 44  66 22
##   large    13  26  78 130 39  39  13 13 143 39 182 39
##   xlarge   45 120 150 105 60 195 240 90  45 75 195 75
```

Now let’s do an OLAP roll up operation to get some insight.

```
apply(revenue_cube, c("year", "size"),
      FUN=function(x) {return(sum(x, na.rm=TRUE))})
```

```
##       size
## year   personal small medium large xlarge
##   2022      756  1512   1210  2106   4125
##   2023     1547  1674   2123  3874   5925
```

As can be seen in the above results, costumers tend to buy bigger pizzas and the gap further increased from 2022 to 2023 which means people are beginning to prefer bigger pizzas.

Part 2

1- Firstly, the CSV file is not separated using commas. It is separated using “;” and the first step is to change the delimiter to “;”. To do so I adde the attribute “sep =”;““. Then, we build a new data frame with the interested columns using ”subset” function.

```
databank.df <- read.csv("/Users/alireza/Desktop/DTI/Semester 1/Fundamentals of Applied
Data Science/assignment 2/bank-additional-full 2.csv", sep = ";")
newdatabank.df <- subset(databank.df, select = c(age, education, previous, pdays, y))
head(newdatabank.df)
```

```
##   age   education previous pdays  y
## 1  56    basic.4y        0   999 no
## 2  57 high.school        0   999 no
## 3  37 high.school        0   999 no
## 4  40    basic.6y        0   999 no
## 5  56 high.school        0   999 no
## 6  45    basic.9y        0   999 no
```

2- In this section the value 999 gets replaced with “NA”.

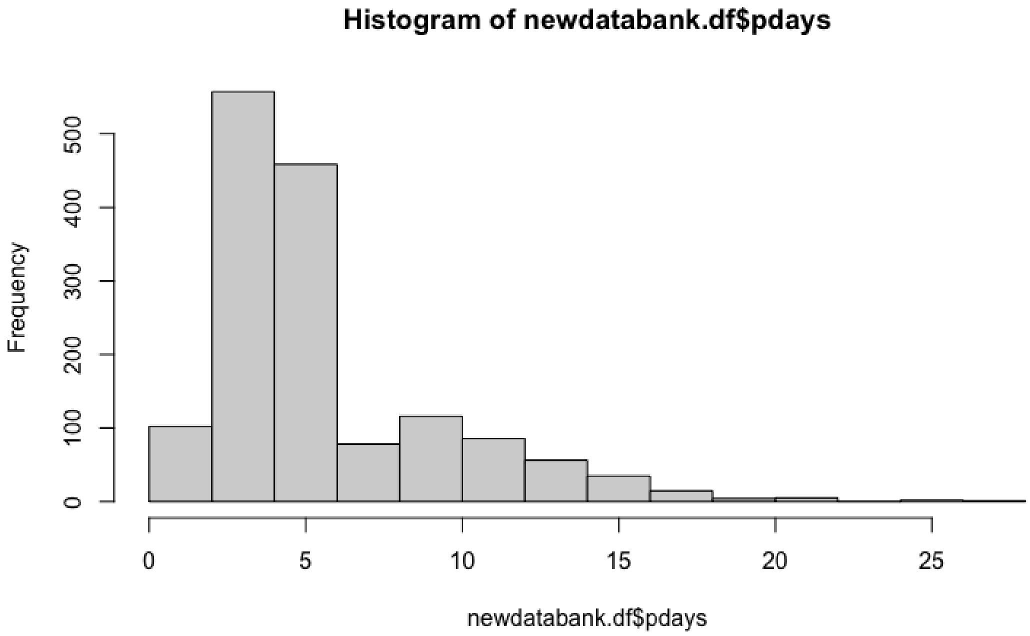
```
newdatabank.df["pdays"][newdatabank.df["pdays"] == 999] <- NA
head(newdatabank.df)
```

```
##   age   education previous pdays  y
## 1  56    basic.4y        0    NA no
## 2  57 high.school        0    NA no
## 3  37 high.school        0    NA no
## 4  40    basic.6y        0    NA no
## 5  56 high.school        0    NA no
## 6  45    basic.9y        0    NA no
```

3 If we don’t change the 999 value to NA, it will affect our analysis and makes this column useless as it wrongly skews the mean of this column. The number 999 is a place holder for missing values and if we leave it as it is in the data it will inflate the data which will lead to inaccurate insights and wrong results of the analysis. Thus, it must be replaced before analyzing pdays column.

4- In this section, firstly, I created a new data frame and copy the main data in it. Then, I excluded the NA rows and created the histogram.

```
new.df <- newdatabank.df

library(tidyr)
newdatabank.df <- newdatabank.df %>% drop_na()
hist(newdatabank.df$pdays)
```



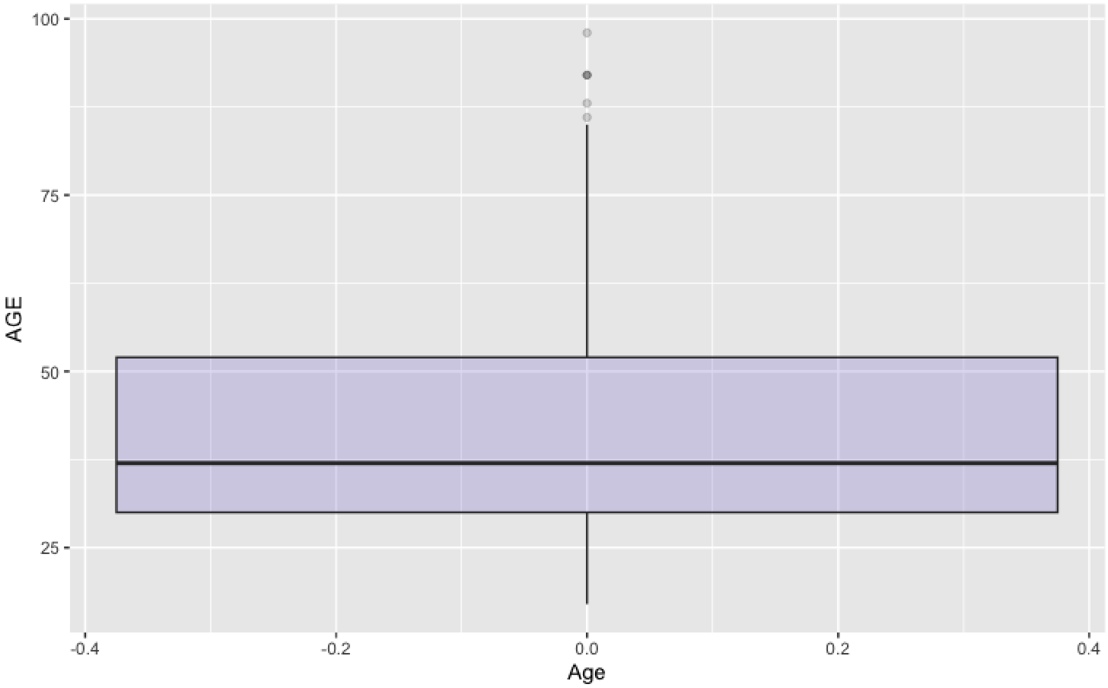
5- In this section characteristic values got transformed into numeric values as asked.

```
a <- 1
newdatabank.df["education"][newdatabank.df["education"] == "illiterate"] <- 0
newdatabank.df["education"][newdatabank.df["education"] == "basic.4y"] <- 4
newdatabank.df["education"][newdatabank.df["education"] == "basic.6y"] <- 6
newdatabank.df["education"][newdatabank.df["education"] == "basic.9y"] <- 9
newdatabank.df["education"][newdatabank.df["education"] == "high.school"] <- 12
newdatabank.df["education"][newdatabank.df["education"] == "professional.course"] <- 12 ^
a
newdatabank.df["education"][newdatabank.df["education"] == "university.degree"] <- 16
newdatabank.df["education"][newdatabank.df["education"] == "unknown"] <- NA
newdatabank.df$education <- as.numeric(newdatabank.df$education)
summary(newdatabank.df)
```

```
##       age          education        previous         pdays       
##  Min.   :17.00   Min.   : 0.00   Min.   :1.000   Min.   : 0.000  
##  1st Qu.:30.00   1st Qu.:12.00   1st Qu.:1.000   1st Qu.: 3.000  
##  Median :37.00   Median :12.00   Median :1.000   Median : 6.000  
##  Mean   :41.85   Mean   :12.39   Mean   :1.661   Mean   : 6.015  
##  3rd Qu.:52.00   3rd Qu.:16.00   3rd Qu.:2.000   3rd Qu.: 7.000  
##  Max.   :98.00   Max.   :16.00   Max.   :7.000   Max.   :27.000  
##                  NA's   :98                                      
##       y            
##  Length:1515       
##  Class :character  
##  Mode  :character  
##                    
##                    
##                    
##
```

6- Using mean, median functions we got the results. For calculating mode of the data we should first create a function for it and use it. Function getmode created to calculate the mode. Then, I plotted the boxplot and the 5 number derived from it are as follows: Min = 17 , 1st Quantile = 30, Median = 37, 3rd Quantile = 52. Also, we could use the summary function to find all these statistics.

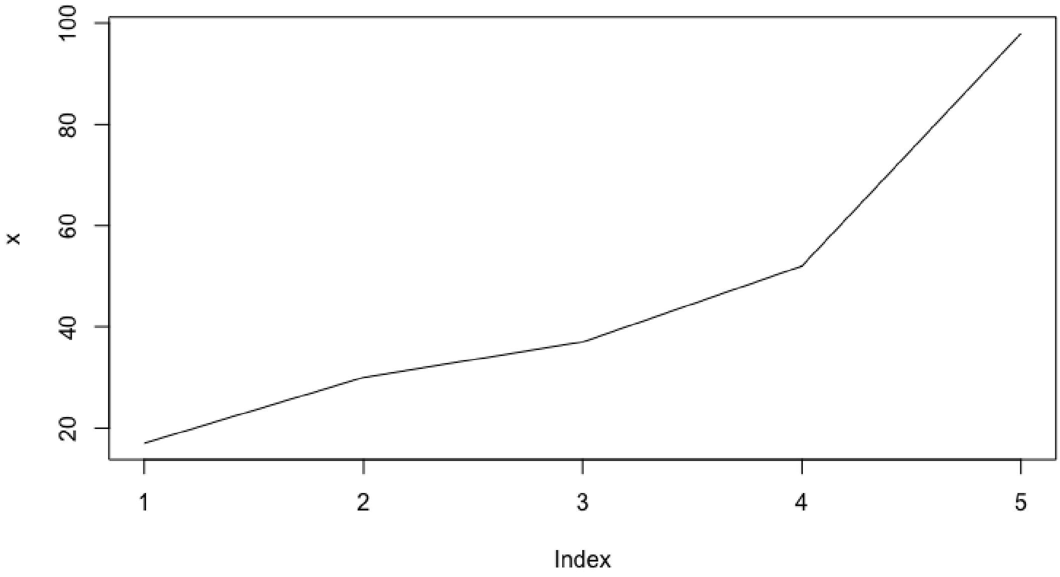
```
library(ggplot2)
AGE <- newdatabank.df$age
mean(AGE)

## [1] 41.85281

median(AGE)

## [1] 37

getmode <- function(v) {
   uniqv <- unique(v)
   uniqv[which.max(tabulate(match(v, uniqv)))]
}
getmode(AGE)

## [1] 29

ggplot(newdatabank.df, aes(y=AGE)) +
    geom_boxplot(fill="slateblue", alpha=0.2) +
    xlab("Age")
```



```
summary(AGE)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   17.00   30.00   37.00   41.85   52.00   98.00
```

And Here is the quantile plot.

```
library(ggplot2)
x <- quantile(AGE)
plot(x, type = "l")
```

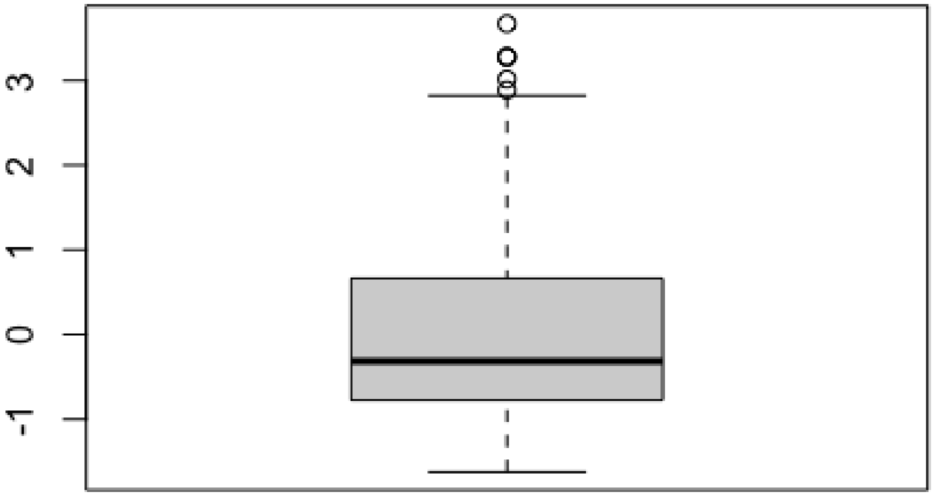


7- In this stage, I normalized the age column using the z-score standardization and assigned it to the age_z variable.

```
age_z <- (AGE - mean(AGE)) / sd(AGE)
```

8- In this stage we detect outliers. While analyzing an standard normal variable, values bigger than 3 and smaller than -3 consider as outliers. In this case, we have 5 outliers.

```
age_z <- as.data.frame(age_z)
#head(age_z)
boxplot(age_z)
```



```
#remove(age_z_out)
(age_z_out <- age_z$age_z[age_z$age_z > 3 | age_z$age_z < -3])

## [1] 3.016923 3.670684 3.278427 3.278427 3.278427
```

Also, to show which rows contain this outliers I used the following code. As there were 5 outliers with more than 3 z-score, I can sort the values by the age column and the 5 values that have the highest age are the outliers.

```
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

df2 <- newdatabank.df %>% arrange(desc(age))
df2[1:5,]

##   age education previous pdays   y
## 1  98         4        2     2 yes
## 2  92        NA        2     6  no
## 3  92        NA        1     3 yes
## 4  92        NA        4     3 yes
## 5  88         4        1     6 yes
```