IDENTIFYING THE CORE MEMBER IN THE NETWORK

In the following text:-

BLUE IS CODE
RED IS OUTPUT
BLACK IS THE EXPLANATION WHAT IS THE PURPOSE OF THE CODE

AIM:-

1. FIND THE CORE MEMBER
2. BUSSIEST AIRPORTS
3. ANALYSE DIFFERENT DELAYS

#Here we are importing different libraries which we will use in our project.
#ggplot2 is used to plot graphs
#igraph is used to perform different functions like centrality measures on graph
#Rno4j is used to access Neo4j server from R environment
#dplyr contain functions for data manipulation
#knitr is used to convert notebook into word format of pdf format

```
library(ggplot2)
library(igraph)
library(magrittr)
library(RNeo4j)
library(dplyr)
library(knitr)
```

```
#here we are reading or importing data file in R dataframe

flights <- read.csv(file.choose(), sep = ",", header = TRUE)
flights$YEAR <- factor(flights$YEAR)
flights$MONTH <- factor(flights$MONTH)
flights$DAY_OF_MONTH <- factor(flights$DAY_OF_MONTH)
flights$ORIGIN_AIRPORT_ID <- factor(flights$ORIGIN_AIRPORT_ID)
flights$DEST_AIRPORT_ID <- factor(flights$DEST_AIRPORT_ID)
flights$DAY_OF_WEEK <- factor(flights$DAY_OF_WEEK)
```

#We take the `flights` dataframe, group it by `ORIGIN` , `DEST` , `CARRIER` , and `FL_DATE` and then summmarize to get the number of flights between each origin and destination. We also find the average of departure delay, arrival delay, security delay, carrier delay and other delays. We have used the `summarize` function to calculate the averages and make new dataframe edges.

```
flights %>%
select(CARRIER,ORIGIN,DEST,WEATHER_DELAY,SECURITY_DELAY,
NAS_DELAY,LATE_AIRCRAFT_DELAY,CARRIER_DELAY,DEP_DELAY_NEW,ARR_DELAY_NEW,
FL_DATE) %>%
na.omit()%>%
group_by(CARRIER,ORIGIN,DEST,FL_DATE) %>%
summarize(nflights = n(),
Arr_delay = mean(ARR_DELAY_NEW, na.rm=TRUE),
Dep_delay = mean(DEP_DELAY_NEW, na.rm=TRUE),
We_delay = mean(WEATHER_DELAY, na.rm=TRUE),
Sec_delay = mean(SECURITY_DELAY, na.rm=TRUE),
Nas_delay = mean(NAS_DELAY, na.rm=TRUE),
Late_air_delay = mean(LATE_AIRCRAFT_DELAY, na.rm=TRUE),
Car_delay = mean(CARRIER_DELAY, na.rm=TRUE)) %>%
group_by() %>%
{.} -> edges.df

#below is the structure of edge dataframe

str(edges.df)

## Classes 'tbl_df', 'tbl' and 'data.frame':    62246 obs. of  12 variables:
##  $ CARRIER       : Factor w/ 13 levels "AA","AS","B6",..: 1 1 1 1 1 1 1 1
1 1 ...
##  $ ORIGIN        : Factor w/ 307 levels "ABE","ABI","ABQ",..: 3 3 3 3 3 3
3 3 3 3 ...
##  $ DEST          : Factor w/ 307 levels "ABE","ABI","ABQ",..: 82 82 82 82
82 82 82 82 82 82 ...
##  $ FL_DATE       : Factor w/ 31 levels "01/12/2015","02/12/2015",..: 1 2 9
10 13 18 21 22 23 26 ...
##  $ nflights      : int  1 1 2 1 1 2 1 1 1 1 ...
##  $ Arr_delay     : num  23 28 34 26 69 76 15 36 54 16 ...
##  $ Dep_delay     : num  24 0 32 41 0 79.5 15 29 66 0 ...
##  $ We_delay      : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Sec_delay     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Nas_delay     : num  0 28 7.5 0 69 0 15 7 0 16 ...
##  $ Late_air_delay: num  0 0 0 26 0 8.5 0 29 54 0 ...
##  $ Car_delay     : num  23 0 26.5 0 0 67.5 0 0 0 0 ...
##  - attr(*, "na.action")= 'omit' Named int  1 2 4 5 6 7 8 9 10 11 ...
##   ..- attr(*, "names")= chr  "1" "2" "4" "5" ...



#here we convert all values in edge data frame to integer. This will help us to calculate delay

cols <- c(6:12)
edges.df[cols] <- lapply(edges.df[cols], as.integer)

new_df =
edges.df[,c("We_delay","Sec_delay","Nas_delay","Late_air_delay","Car_delay")]
```

```
Cause_Of_delay = colnames(new_df)[apply(new_df,1,which.max)]
edges.df = cbind(edges.df,Cause_Of_delay)

str(edges.df)
```

```
## 'data.frame':    62246 obs. of  13 variables:
##  $ CARRIER       : Factor w/ 13 levels "AA","AS","B6",..: 1 1 1 1 1 1 1 1
1 1 ...
##  $ ORIGIN        : Factor w/ 307 levels "ABE","ABI","ABQ",..: 3 3 3 3 3 3
3 3 3 3 ...
##  $ DEST          : Factor w/ 307 levels "ABE","ABI","ABQ",..: 82 82 82 82
82 82 82 82 82 82 ...
##  $ FL_DATE       : Factor w/ 31 levels "01/12/2015","02/12/2015",..: 1 2 9
10 13 18 21 22 23 26 ...
##  $ nflights      : int  1 1 2 1 1 2 1 1 1 1 ...
##  $ Arr_delay     : int  23 28 34 26 69 76 15 36 54 16 ...
##  $ Dep_delay     : int  24 0 32 41 0 79 15 29 66 0 ...
##  $ We_delay      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Sec_delay     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Nas_delay     : int  0 28 7 0 69 0 15 7 0 16 ...
##  $ Late_air_delay: int  0 0 0 26 0 8 0 29 54 0 ...
##  $ Car_delay     : int  23 0 26 0 0 67 0 0 0 0 ...
##  $ Cause_Of_delay: Factor w/ 5 levels "Car_delay","Late_air_delay",..: 1 3
1 2 3 1 3 2 2 3 ...
```

```
#we start neo4j server here the database is at localhost at port 7474. Neo4j
help to store data in graph form
```

```
graph = startGraph("http://localhost:7474/db/data/",username =
"neo4j",password = "hello")
```

#here first we clear any previous data store in graph database than we create nodes of airport name and edges from source to destination using edge.df

```
clear(graph, input = FALSE)
addConstraint(graph, "Airport", "name")
lapply(unique(c(as.character(edges.df$ORIGIN),
as.character(edges.df$DEST))),
function(a.airport) {
createNode(graph,
"Airport",
name=a.airport)
}
)
```

```
## [[1]]
## < Node >
## Airport
##
## $name
## [1] "ABQ"
```

```
##
##
## [[2]]
## < Node >
## Airport
##
## $name
## [1] "ALB"
##
##
## [[3]]
## < Node >
## Airport
##
## $name
## [1] "ANC"
##
##
```

#We write edges.df to edges.csv file

```
write.csv(edges.df,
"/Users/magress/Documents/neo4jDatabases/database-bb60fc0e-ec3e-4a82-bc33-
31dd9e68283a/installation-3.4.1/import/edges.csv")
```

#Cypher is used to query the Neo4j database. In the below query, we are loading the edges.csv in Neo4j and matching the source and destination airports. We are creating the connection between them to see which airport is connected to which one.

```
cypher(graph,
"LOAD CSV WITH HEADERS FROM 'file:/edges.csv' AS row
MATCH (src:Airport {name:row.ORIGIN}),
(tgt:Airport {name:row.DEST})
CREATE (src)-[:flight {carrier:row.CARRIER,
number:row.nflights,
arr_delay:row.Arr_delay,
cause:row.Cause_Of_delay,
dep_delay:row.Dep_delay}]->(tgt)")
```

#we find out which airports have the highest average departure delay. We find 15 such airports.

```
mostDepDelay15 = cypher(graph,
"MATCH (src)-[r:flight]->(tgt)
WITH src.name as Origin,
AVG(toInt(r.dep_delay)) AS Avg_dep_delay_time,
count(tgt) as numberOfDest
ORDER BY Avg_dep_delay_time desc
RETURN Origin, numberOfDest,
```

```
Avg_dep_delay_time LIMIT 15")
mostDepDelay15
```

```
##    Origin numberOf Dest Avg_dep_delay_time
## 1     PPG             1           798.0000
## 2     RKS             7           229.2857
## 3     OTH             4           214.0000
## 4     SWF             8           179.5000
## 5     ESC             4           152.5000
## 6     COU            13           152.0000
## 7     GCK             7           144.0000
## 8     SJT             8           139.1250
## 9     SPI            21           138.2857
## 10    BPT            13           136.0000
## 11    VLD             9           132.7778
## 12    MBS            23           127.5217
## 13    CSG            11           126.5455
## 14    CHO            29           126.4828
## 15    GGG            10           125.6000
```

#Now we select some airports and analyse what are the measure causes of delays on these airports.

```
investigation_1 = cypher(graph,
"MATCH (src)-[r:flight]->(tgt)
WITH src.name AS Origin, r.cause as Cause,
tgt.name as Destination,
AVG(toInt(r.dep_delay)) AS Avg_dep_delay_time
ORDER BY Avg_dep_delay_time desc
WHERE src.name IN ['ESC','PIH','ABR','VEL','SMX','EAU']
RETURN Origin, Destination, Cause, Avg_dep_delay_time")
investigation_1
```

```
##    Origin Destination          Cause Avg_dep_delay_time
## 1     ABR         MSP      Car_delay          257.00000
## 2     ESC         DTW Late_air_delay          220.50000
## 3     ESC         DTW       We_delay          155.00000
## 4     PIH         SLC Late_air_delay           78.00000
## 5     SMX         SFO      Nas_delay           77.83333
## 6     SMX         SFO Late_air_delay           69.14286
## 7     EAU         ORD Late_air_delay           51.14286
## 8     ABR         MSP Late_air_delay           49.50000
## 9     PIH         SLC      Car_delay           33.00000
## 10    ABR         MSP       We_delay           30.00000
## 11    ESC         DTW      Nas_delay           14.00000
## 12    ABR         MSP      Sec_delay           11.00000
## 13    PIH         SLC      Nas_delay            7.87500
```

```
## 14    ABR         MSP      Nas_delay            0.00000
## 15    EAU         ORD      Nas_delay            0.00000
```

#HERE we investigate the carrier responsible for delays

```
culpritCarriers = cypher(graph,
"MATCH (src)-[r:flight]->(tgt)
WITH src.name AS Origin, r.cause as Cause,
tgt.name as Destination, r.carrier as Carrier,
AVG(toInt(r.dep_delay)) AS Avg_dep_delay_time
ORDER BY Avg_dep_delay_time desc
WHERE src.name IN ['ESC','PIH','ABR','VEL','SMX','EAU']
AND r.cause = 'Car_delay'
RETURN Origin, Destination, Cause, Avg_dep_delay_time, Carrier")
culpritCarriers
```

```
##   Origin Destination     Cause Avg_dep_delay_time Carrier
## 1    ABR         MSP Car_delay                257      OO
## 2    PIH         SLC Car_delay                 33      OO
```

#here we will findout the airports with very low connectivity

```
airportsLeastConnected = cypher(graph,
"MATCH (src)-[r:flight]->(tgt)
WITH tgt.name as Destination,
count(distinct src) as flightsFromAirport
ORDER BY flightsFromAirport asc
RETURN Destination, flightsFromAirport")

#[1] 75
```

#here we investigate about the possible ways to reach from one airport to another.

```
HowToReachThere = cypher(graph,
"MATCH (src)-[r:flight]->(tgt)
WITH src.name as Origin,
tgt.name as Destination,
r.carrier as Carrier,
sum(toInt(r.number)) as numberOfFlights
WHERE tgt.name IN ['DVL','BTM','GUM']
RETURN Origin, Destination, Carrier, numberOfFlights")
HowToReachThere
```

```
##   Origin Destination Carrier numberOfFlights
## 1    JMS         DVL      OO               7
## 2    HNL         GUM      UA               3
## 3    SLC         BTM      OO              14
```

#In above result we find out that to reach GUM there are flights only from HNL airport. Here we will findout about the connectivity of HNL airpot.

```
ToHonolulu = cypher(graph,
"MATCH (src)-[r:flight]->(tgt)
WITH tgt.name as Destination,
sum(toInt(r.number)) as numberOfFlights,
count(distinct src) as numberOfOrigins
WHERE tgt.name = 'HNL'
RETURN Destination, numberOfOrigins, numberOfFlights")
ToHonolulu
```

```
##   Destination numberOfOrigins numberOfFlights
## 1         HNL              27             417
```

#Below we are creating a graph dataframe which has source airport and destination airport as vertices.

```
edges.df %>%
select(ORIGIN, DEST) %>%
distinct() -> edges.df.igraph
ig = graph.data.frame(edges.df.igraph)



#here we find out the indegree of airports

ig_degree_in = sort(degree(ig, mode = "in"),decreasing = TRUE)
ig_degree_in
```

```
## ATL ORD DFW DEN IAH MSP DTW EWR SLC PHX SFO LAS LAX SEA MCO BWI CLT MDW
## 158 150 140 126 111 104 100  82  79  75  75  73  72  69  67  62  62  61
## FLL JFK BOS TPA LGA DAL HOU MIA DCA STL PDX BNA SAN AUS MCI MSY PHL RDU
##  58  57  55  54  50  50  45  44  43  43  42  41  41  38  37  36  36  32
## OAK IAD RSW SAT CLE HNL MKE PIT CVG CMH IND SJC SMF SJU SNA ABQ ANC PBI
##  31  29  29  28  27  27  26  26  26  25  25  25  25  23  22  21  20  20
## BDL MEM OKC JAX CHS OGG OMA BOI BUF SDF TUL TUS BHM ONT RIC ALB DSM ELP
##  19  19  19  18  17  16  16  15  15  15  15  15  14  13  13  12  12  12
## JAC GRR ORF PSP PVD RNO ROC STT BUR LIT KOA LGB BZN TTN EGE FAT GEG HDN
##  12  12  11  11  11  11  11  11  11  11  10  10  10  10   9   9   9   9
## LIH MTJ GSP ICT MSN COS DAY SYR SAV MHT XNA ASE GSO LBB PWM HPN SRQ CAK
##   9   9   9   9   9   8   8   8   8   8   8   8   7   7   7   7   7   7
## JNU BTV FNT PNS TYS CID MAF MLI PIA ACY SBA CAE FAR HSV LEX MYR SGF AMA
##   6   6   6   6   6   6   6   6   6   6   6   5   5   5   5   5   5   5
## BMI GJT SBN FWA PSC ISP GUC MDT STX BLI KTN BQN ATW AVP BIS CHA ECP EVV
##   5   5   5   5   5   5   4   4   4   4   4   4   4   4   4   4   4   4
## FSD GRB JAN MOB MSO RAP LNK SHV LBE BFL FAI AVL BIL BTR CHO CRW GPT LFT
##   4   4   4   4   4   4   4   4   4   4   3   3   3   3   3   3   3   3
## TLH VPS ABE AEX AZO CRP ELM GRK HRL ISN LAN MLU RST SCE ITO DRO EUG FCA
##   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
## GTF HLN MFR MRY RDM SBP SUN ILM MFE BRW CDV OME PSG SCC SIT WRG YAK ORH
##   3   3   3   3   3   3   3   2   2   2   2   2   2   2   2   2   2   2
## PSE SWF BGR DLH EYW GNV ROA BRO CLL FSM LCH LRD MEI MGM SPI TVC TYR CMI
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
```

```
## COU LSE MHK MQT CPR GCC HIB IDA IMT INL JMS MBS MMH MOT RHI SGU ADK ADQ
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   1   1
## BET OTZ AGS DAB FAY MLB TRI ABY ACT BPT BQK CSG CWA DHN ERI EWN GTR HOB
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## JLN LAW OAJ PHF PIB ROW SAF TXK VLD UST PPG ABI ALO DBQ GCK GGG GRI SJT
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## SPS SUX TOL IAG PBG ABR ACV APN BGM BJI BRD BTM CDC CIU CMX COD DVL EAU
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## EKO ESC FLG GFK HYS ITH LAR LWS MKG OTH PAH PIH PLN RDD RKS SMX TWF YUM
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## GUM
##   1
```

#here we find out the out-degree of airports

```
ig_degree_out = sort(degree(ig, mode = "out"),decreasing = TRUE)
ig_degree_out
```

```
## ATL ORD DFW DEN IAH MSP DTW EWR SLC PHX SFO LAS LAX SEA MCO BWI CLT MDW
## 158 152 140 126 110 106  99  82  78  76  75  73  71  69  66  62  62  61
## FLL JFK BOS TPA LGA DAL HOU MIA DCA STL BNA SAN PDX AUS MCI MSY PHL OAK
##  57  57  55  54  51  50  46  45  43  43  42  42  41  40  37  36  36  33
## RDU IAD RSW SAT CLE CMH HNL IND MKE SMF CVG PIT SJC SJU SNA ABQ BDL OKC
##  31  29  29  28  27  26  26  26  26  24  24  23  23  22  22  21  20  20
## ANC JAX MEM PBI OGG CHS OMA BUF SDF TUS BOI TUL BHM JAC ONT RIC ALB DSM
##  19  19  19  19  18  16  16  15  15  15  14  14  14  13  13  13  12  12
## ELP GRR ORF PSP PVD RNO ROC BUR LIT EGE KOA MTJ STT LGB BZN TTN DAY FAT
##  12  12  11  11  11  11  11  11  11  10  10  10  10  10  10  10   9   9
## GEG HDN GSP ICT XNA COS LIH SYR MHT MSN ASE GSO LBB PWM HPN SAV SRQ CAK
##   9   9   9   9   9   8   8   8   8   8   8   7   7   7   7   7   7   7
## TYS JNU BTV FNT PNS CID MAF MLI PIA ACY SBA CAE FAR HSV LEX MYR RAP SGF
##   7   6   6   6   6   6   6   6   6   6   6   5   5   5   5   5   5   5
## AMA BMI GJT SBN FWA PSC ISP GUC MDT STX BLI KTN BQN SWF ATW AVP BIL BIS
##   5   5   5   5   5   5   5   4   4   4   4   4   4   4   4   4   4   4
## CHA ECP EVV FSD GRB JAN MOB MSO LNK SHV BFL EUG BRW AVL BTR CHO CRW GPT
##   4   4   4   4   4   4   4   4   4   4   4   4   3   3   3   3   3   3
## LFT TLH VPS ABE AEX AZO CRP ELM GRK HRL ISN LAN MLU RST DRO LBE GTF MFR
##   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
## MRY RDM SUN ILM MFE CDV FAI OTZ PSG SIT WRG YAK ORH PSE BGR DLH EYW GNV
##   3   3   3   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## ROA BRO CLL FSM LCH LRD MEI MGM PIB SCE SPI TVC TYR ITO CMI COU LSE MHK
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## MQT COD CPR DVL FCA GCC HIB HLN IDA IMT INL JMS MBS MMH MOT RHI SBP SGU
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## ADK ADQ BET OME SCC AGS DAB FAY MLB TRI ABY ACT BPT BQK CSG CWA DHN ERI
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## EWN GTR HOB JLN LAW OAJ PHF ROW SAF TXK VLD UST PPG ABI ALO DBQ GCK GGG
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## GRI SJT SPS SUX TOL IAG PBG ABR ACV APN BGM BJI BRD BTM CDC CIU CMX EAU
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
```

```
## EKO ESC FLG GFK HYS ITH LAR LWS MKG OTH PAH PIH PLN RDD RKS SMX TWF YUM
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## GUM
##   0
```

#here we find out degree of airports

```
ig_degree_out = sort(degree(ig),decreasing = TRUE)
ig_degree_out
```

```
## ATL ORD DFW DEN IAH MSP DTW EWR SLC PHX SFO LAS LAX SEA MCO BWI CLT MDW
## 316 302 280 252 221 210 199 164 157 151 150 146 143 138 133 124 124 122
## FLL JFK BOS TPA LGA DAL HOU MIA DCA STL BNA PDX SAN AUS MCI MSY PHL OAK
## 115 114 110 108 101 100  91  89  86  86  83  83  83  78  74  72  72  64
## RDU IAD RSW SAT CLE HNL MKE CMH IND CVG PIT SMF SJC SJU SNA ABQ ANC BDL
##  63  58  58  56  54  53  52  51  51  50  49  49  48  45  44  42  39  39
## OKC PBI MEM JAX OGG CHS OMA BUF SDF TUS BOI TUL BHM ONT RIC JAC ALB DSM
##  39  39  38  37  34  33  32  30  30  30  29  29  28  26  26  25  24  24
## ELP GRR ORF PSP PVD RNO ROC BUR LIT STT KOA LGB BZN TTN EGE MTJ FAT GEG
##  24  24  22  22  22  22  22  22  22  21  20  20  20  20  19  19  18  18
## HDN GSP ICT DAY LIH MSN XNA COS SYR MHT ASE SAV GSO LBB PWM HPN SRQ CAK
##  18  18  18  17  17  17  17  16  16  16  16  15  14  14  14  14  14  14
## TYS JNU BTV FNT PNS CID MAF MLI PIA ACY SBA CAE FAR HSV LEX MYR SGF AMA
##  13  12  12  12  12  12  12  12  12  12  12  10  10  10  10  10  10  10
## BMI GJT SBN FWA PSC ISP RAP GUC MDT STX BLI KTN BQN ATW AVP BIS CHA ECP
##  10  10  10  10  10  10   9   8   8   8   8   8   8   8   8   8   8   8
## EVV FSD GRB JAN MOB MSO LNK SHV BFL BIL LBE EUG SWF AVL BTR CHO CRW GPT
##   8   8   8   8   8   8   8   8   8   7   7   7   6   6   6   6   6   6
## LFT TLH VPS ABE AEX AZO CRP ELM GRK HRL ISN LAN MLU RST DRO GTF MFR MRY
##   6   6   6   6   6   6   6   6   6   6   6   6   6   6   6   6   6   6
## RDM SUN BRW FAI SCE ITO FCA HLN SBP ILM MFE CDV PSG SIT WRG YAK ORH PSE
##   6   6   5   5   5   5   5   5   5   4   4   4   4   4   4   4   4   4
## BGR DLH EYW GNV ROA BRO CLL FSM LCH LRD MEI MGM SPI TVC TYR CMI COU LSE
##   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4
## MHK MQT CPR GCC HIB IDA IMT INL JMS MBS MMH MOT RHI SGU OME OTZ SCC PIB
##   4   4   4   4   4   4   4   4   4   4   4   4   4   4   3   3   3   3
## COD DVL ADK ADQ BET AGS DAB FAY MLB TRI ABY ACT BPT BQK CSG CWA DHN ERI
##   3   3   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## EWN GTR HOB JLN LAW OAJ PHF ROW SAF TXK VLD UST PPG ABI ALO DBQ GCK GGG
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## GRI SJT SPS SUX TOL IAG PBG ABR ACV APN BGM BJI BRD BTM CDC CIU CMX EAU
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## EKO ESC FLG GFK HYS ITH LAR LWS MKG OTH PAH PIH PLN RDD RKS SMX TWF YUM
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## GUM
##   1
```

#here we find about the airports connected to ALT airport

```
E(ig)[incident(ig, v=V(ig)['ATL'])]
```

```
## + 316/4012 edges from 648c7d0 (vertex names):
##  [1] ATL->ABQ ATL->ALB ATL->AUS ATL->BDL ATL->BNA ATL->BOS ATL->BUF
##  [8] ATL->BWI ATL->CHS ATL->CLE ATL->CLT ATL->CMH ATL->COS ATL->DAY
## [15] ATL->DCA ATL->DEN ATL->DFW ATL->DSM ATL->DTW ATL->EGE ATL->ELP
## [22] ATL->EWR ATL->FLL ATL->GSO ATL->HDN ATL->HNL ATL->IAD ATL->IAH
## [29] ATL->ILM ATL->IND ATL->JAC ATL->JAX ATL->JFK ATL->LAS ATL->LAX
## [36] ATL->LGA ATL->MCI ATL->MCO ATL->MDT ATL->MEM ATL->MIA ATL->MKE
## [43] ATL->MSP ATL->MSY ATL->MTJ ATL->OAK ATL->OKC ATL->OMA ATL->ORD
## [50] ATL->ORF ATL->PBI ATL->PDX ATL->PHL ATL->PHX ATL->PIT ATL->PVD
## [57] ATL->PWM ATL->RDU ATL->RIC ATL->ROC ATL->RSW ATL->SAN ATL->SAT
## [64] ATL->SDF ATL->SEA ATL->SFO ATL->SJC ATL->SJU ATL->SLC ATL->SMF
```

#here we calculate betweenness

```
ig_betweeness = sort(betweenness(ig,directed = TRUE), decreasing = TRUE)
ig_betweeness[1:100]
```

```
##          ATL          DFW          ORD          DEN          MSP
## 1.794328e+04 1.640933e+04 1.639265e+04 1.110388e+04 8.500343e+03
##          DTW          SLC          IAH          ANC          SEA
## 6.985199e+03 6.705842e+03 6.031948e+03 5.257792e+03 5.007810e+03
##          SFO          PHX          FLL          MCO          LAX
## 4.836943e+03 2.870925e+03 2.617660e+03 2.090592e+03 1.818520e+03
##          JNU          EWR          HNL          LAS          JFK
## 1.492961e+03 1.491552e+03 1.435428e+03 1.059160e+03 9.936609e+02
##          CLT          KTN          TPA          MIA          LGA
## 8.932115e+02 8.609723e+02 6.636317e+02 6.537879e+02 6.457977e+02
##          TTN          BWI          PDX          BOS          HOU
## 6.104205e+02 5.869720e+02 5.372171e+02 3.993869e+02 3.845677e+02
##          MDW          BRW          MEI          JMS          AUS
## 3.617114e+02 3.050000e+02 3.040000e+02 3.040000e+02 2.966855e+02
##          RSW          DAL          PBI          OGG          OAK
## 2.813628e+02 2.578884e+02 2.280575e+02 1.938561e+02 1.701363e+02
##          BNA          SAN          SJC          PHL          CDV
## 1.678073e+02 1.245947e+02 9.356702e+01 9.236464e+01 7.803325e+01
##          SMF          DCA          SJU          RDU          STL
## 7.582010e+01 7.290015e+01 6.759755e+01 6.503626e+01 5.886785e+01
##          IAD          FAI          MCI          BOI          MSY
## 5.866572e+01 3.837864e+01 3.143590e+01 2.828587e+01 2.128840e+01
##          SAT          CVG          MYR          SNA          CMH
## 1.866901e+01 1.547010e+01 1.342506e+01 1.002035e+01 9.577986e+00
##          PIT          CLE          MKE          IND          ABQ
## 9.368787e+00 9.330441e+00 9.263270e+00 8.672587e+00 8.248844e+00
##          PSG          OKC          MEM          BDL          TUL
## 7.966667e+00 7.812695e+00 7.009155e+00 6.657085e+00 6.283280e+00
##          OMA          YAK          BHM          CHS          JAX
## 5.374374e+00 4.466667e+00 4.360354e+00 4.126267e+00 3.868406e+00
##          ICT          ACY          LIT          GSP          ELP
## 3.596081e+00 3.483028e+00 3.110545e+00 2.790945e+00 2.630152e+00
```

```
##          KOA          ONT          SDF          ORF          HPN
## 2.421235e+00 2.197151e+00 1.872896e+00 1.502112e+00 1.433028e+00
##          DSM          TUS          LBB          MAF          BUF
```

#here we calculate closeness

```
ig_closeness = sort(closeness(ig),decreasing = TRUE)
head(ig_closeness,100)
```

```
       ORD         ATL         DFW         DEN         MSP         IAH
0.002145923 0.002136752 0.002057613 0.002032520 0.001949318 0.001934236
       DTW         PHX         SFO         EWR         SEA         LAX
0.001890359 0.001841621 0.001838235 0.001834862 0.001831502 0.001824818
       SLC         LAS         MCO         CLT         BWI         JFK
0.001814882 0.001805054 0.001779359 0.001766784 0.001763668 0.001751313
       BOS         FLL         PDX         DCA         MIA         STL
0.001742160 0.001730104 0.001709402 0.001706485 0.001706485 0.001706485
       TPA         SAN         AUS         BNA         MCI         MSY
0.001706485 0.001703578 0.001697793 0.001692047 0.001689189 0.001686341
       PHL         LGA         SAT         RDU         MDW         IAD
0.001686341 0.001661130 0.001658375 0.001655629 0.001655629 0.001647446
       MKE         OKC         OAK         CLE         CVG         IND
0.001639344 0.001639344 0.001628664 0.001623377 0.001623377 0.001620746
       SMF         HNL         PIT         ABQ         CMH         SJC
0.001620746 0.001612903 0.001612903 0.001610306 0.001610306 0.001607717
       SNA         DAL         OMA         RSW         TUS         BDL
0.001605136 0.001600000 0.001597444 0.001594896 0.001587302 0.001582278
       JAC         MEM         HOU         SDF         TUL         DSM
0.001582278 0.001579779 0.001579779 0.001574803 0.001572327 0.001569859
       JAX         GRR         PBI         SJU         MSN         DAY
0.001562500 0.001552795 0.001547988 0.001543210 0.001543210 0.001540832
       COS         BZN         LIT         XNA         OGG         BHM
0.001536098 0.001536098 0.001531394 0.001531394 0.001529052 0.001529052
       ASE         EGE         MTJ         RIC         ELP         PSP
0.001529052 0.001526718 0.001524390 0.001519757 0.001517451 0.001517451
       ICT         MLI         RNO         TYS         FAR         BOI
0.001517451 0.001517451 0.001512859 0.001508296 0.001499250 0.001497006
       HDN         CHS         SGF         PIA         ORF         CID
0.001494768 0.001488095 0.001481481 0.001481481 0.001474926 0.001474926
       RAP         BMI         FWA         ANC         ALB         GSP
0.001472754 0.001472754 0.001472754 0.001466276 0.001459854 0.001459854
       BUF         GSO         BIS         FSD
0.001455604 0.001455604 0.001453488 0.001449275
```

#We are making a data frame to plot the busiest airports. Below code creates a data frame edges.df.new .

```
flights %>%
select(CARRIER,ORIGIN,DEST,DEP_DELAY_NEW) %>%
na.omit()%>%
```

```
group_by(ORIGIN,DEST) %>%
summarize(nflights = n(),
Dep_delay = mean(DEP_DELAY_NEW, na.rm=TRUE)) %>%
group_by() %>%
{.} -> edges.df.new

edges.df.new %>%
arrange(., desc(nflights)) %>%
mutate(avgDailyFlights = ceiling(nflights / 31)) -> edges.df.new.1

summary(edges.df.new.1)
```

```
##  ORIGIN          DEST         nflights        Dep_delay
##  ATL    : 159   ATL    : 159   Min.   :   1.0   Min.   :  0.00
##  ORD    : 154   ORD    : 153   1st Qu.:  31.0   1st Qu.:  8.81
##  DFW    : 140   DFW    : 141   Median :  73.0   Median : 13.00
##  DEN    : 127   DEN    : 128   Mean   : 114.6   Mean   : 14.68
##  IAH    : 110   IAH    : 111   3rd Qu.: 144.5   3rd Qu.: 17.82
##  MSP    : 109   MSP    : 109   Max.   :1333.0   Max.   :256.00
##  (Other):3316   (Other):3314
##  avgDailyFlights
##  Min.   : 1.000
##  1st Qu.: 1.000
##  Median : 3.000
##  Mean   : 4.096
##  3rd Qu.: 5.000
##  Max.   :43.000
##
```

RESULTS:

1. CORE MEMBER IS ALT AIRPORT
2. BUSSIEST AIRPORTS ARE ALT, ORD, DFW
3. THE MAIN RESON FOR DELAY IS CARRIER DELAY AND OO AIRLINE IS THE MOST DELAYED AIRLINE.