HW4
Author: Vamsitha

```
library(tidyverse)

## Warning: package 'ggplot2' was built under R version 4.3.3

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.4     ✔ readr     2.1.5
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ ggplot2   3.5.0     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.1
## ✔ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

library(tidygraph)

## Warning: package 'tidygraph' was built under R version 4.3.3

##
## Attaching package: 'tidygraph'
##
## The following object is masked from 'package:stats':
##
##     filter

library(ggraph)

## Warning: package 'ggraph' was built under R version 4.3.3

library(readxl)
library(deldir)

## deldir 2.0-4      Nickname: "Idol Comparison"
##
##      The syntax of deldir() has changed since version
##      0.0-10.  Read the help!!!.

nodes1 <- read_excel("C:/Users/vamsitha/Downloads/EmployeeEmails.xlsx", sheet = "Departments")
edges1 <- read_excel("C:/Users/vamsitha/Downloads/EmployeeEmails.xlsx", sheet = "Emails")
```

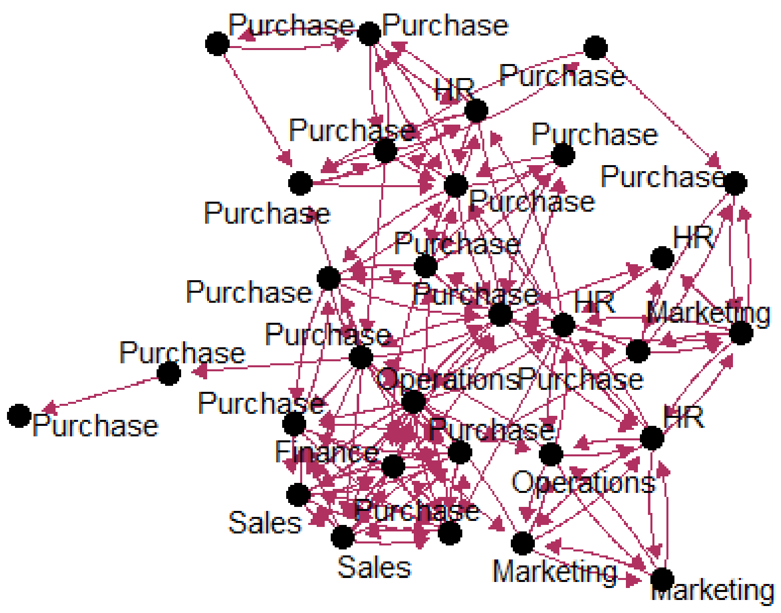
```
mydata<- tbl_graph(nodes=nodes1, edges=edges1, directed=TRUE)
mydata
```

```
## # A tbl_graph: 30 nodes and 149 edges
## #
## # A directed simple graph with 1 component
## #
## # Node Data: 30 × 2 (active)
##    employee department
##       <dbl> <chr>
##  1        1 Marketing
##  2        2 HR
##  3        3 Operations
##  4        4 Marketing
##  5        5 Marketing
##  6        6 HR
##  7        7 HR
##  8        8 HR
##  9        9 Purchase
## 10       10 Purchase
## # ℹ 20 more rows
## #
## # Edge Data: 149 × 3
##    from    to frequency
##   <int> <int>     <dbl>
## 1     1     2         1
## 2     1     6        13
## 3     1     7        22
## # ℹ 146 more rows
```

```
ggraph(mydata) +
 geom_edge_fan( color="maroon", arrow=arrow(length=unit(2,"mm"),
type="closed"), end_cap=circle(3,"mm"), angle_calc = "along", label_color =
"green", label_dodge = unit(5, "mm"))+  geom_node_point(size=4) +
  geom_node_text(aes(label= department), repel=TRUE, vjust=0.5) +
  theme_graph()
```

```
## Using "stress" as default layout
```

```
## Warning: Duplicated aesthetics after name standardisation: label_colour
```

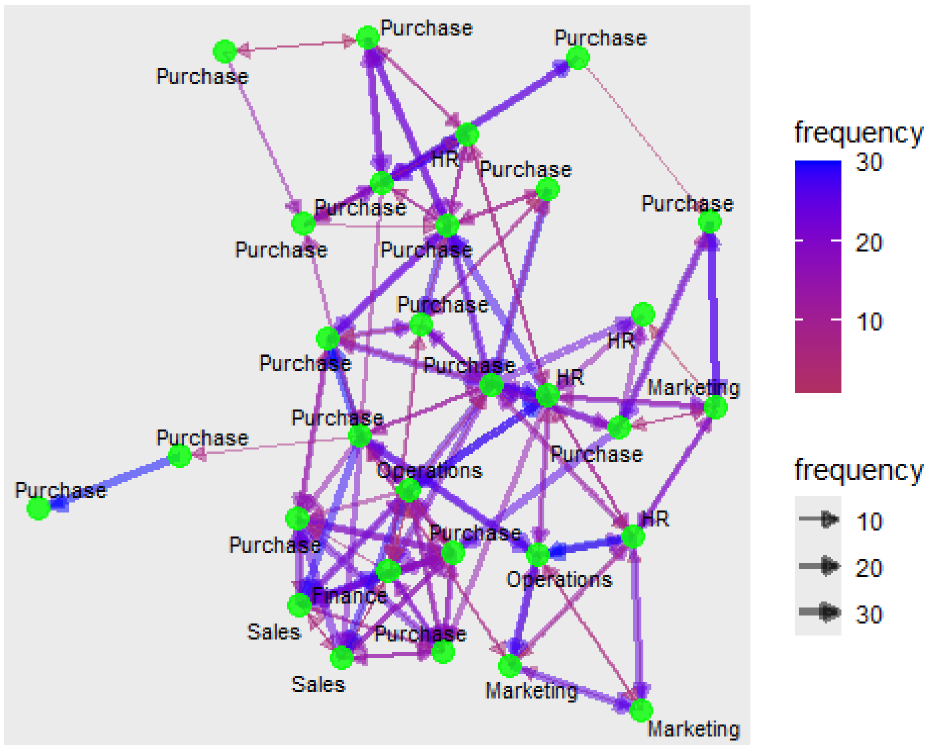


R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

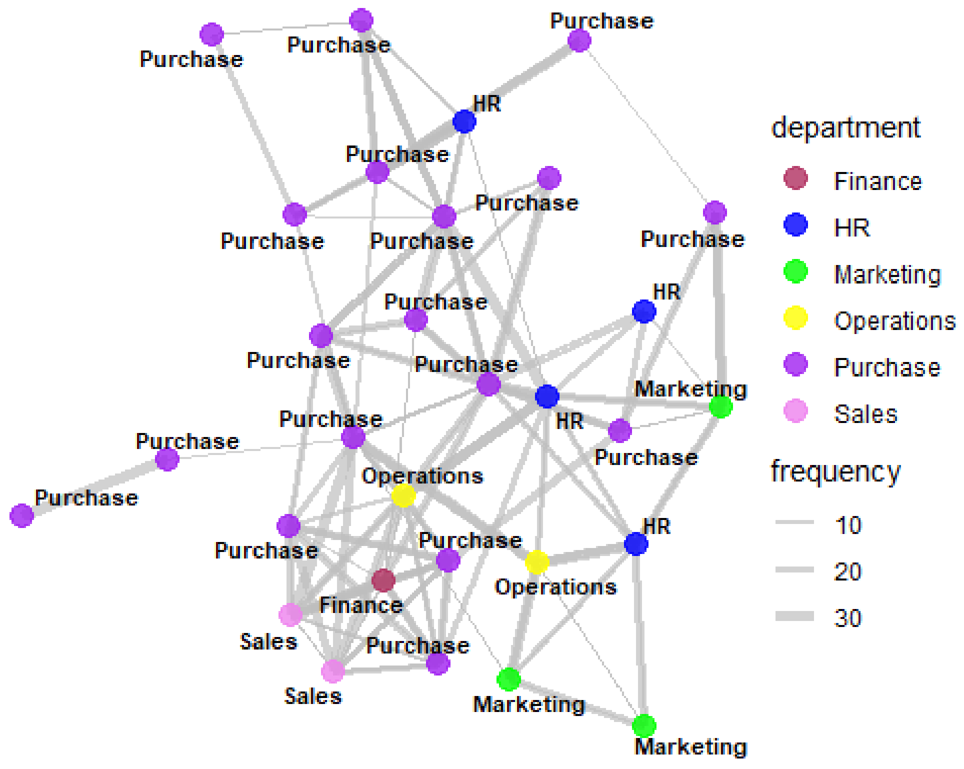
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```r
ggraph(mydata, layout = 'stress') +
  geom_edge_link(aes(width = frequency, color = frequency),
                 arrow = arrow(length = unit(2, "mm"), type = "closed"),
                 end_cap = circle(1.5, "mm"),
                 alpha = 0.5) +
  scale_edge_width(range = c(0.4, 1.75)) +
  scale_edge_color_gradient(low = "maroon", high = "blue") +
  geom_node_point(color = "green", size = 4, alpha = 0.8) +
  geom_node_text(aes(label = department),
                 repel = TRUE,
                 size = 3,
                 color = "black",
)
```



```
  theme_graph(base_size = 10, background = "white") +
  theme(legend.position = "right")

## List of 136
##  $ line                              :List of 6
##   ..$ colour       : chr "black"
##   ..$ linewidth    : num 0.455
##   ..$ linetype     : num 1
##   ..$ lineend      : chr "butt"
##   ..$ arrow        : logi FALSE
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_line" "element"
##  $ rect                              :List of 5
##   ..$ fill         : chr "white"
##   ..$ colour       : chr "black"
##   ..$ linewidth    : num 0.455
##   ..$ linetype     : num 1
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_rect" "element"
##  $ text                              :List of 11
##   ..$ family       : chr "Arial Narrow"
##   ..$ face         : chr "plain"
##   ..$ colour       : chr "black"
##   ..$ size         : num 10
##   ..$ hjust        : num 0.5
##   ..$ vjust        : num 0.5
##   ..$ angle        : num 0
##   ..$ lineheight   : num 0.9
##   ..$ margin       : 'margin' num [1:4] 0points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : logi FALSE
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ title                           : NULL
##  $ aspect.ratio                    : NULL
##  $ axis.title                      : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ axis.title.x                    :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 2.5points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.title.x.top                :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 0
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 2.5points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.title.x.bottom             : NULL
##  $ axis.title.y                    :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : num 90
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 2.5points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.title.y.left               : NULL
##  $ axis.title.y.right              :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : num -90
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 0points 2.5points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text                       : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ axis.text.x                     :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 2points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.x.top                 :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 0
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 2points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.x.bottom              : NULL
##  $ axis.text.y                     :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 1
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 2points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.y.left                : NULL
##  $ axis.text.y.right               :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 0points 2points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.theta                 : NULL
##  $ axis.text.r                     :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0.5
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 2points 0points 2points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.ticks                      : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ axis.ticks.x                    : NULL
##  $ axis.ticks.x.top                : NULL
##  $ axis.ticks.x.bottom             : NULL
##  $ axis.ticks.y                    : NULL
##  $ axis.ticks.y.left               : NULL
##  $ axis.ticks.y.right            : NULL
##  $ axis.ticks.theta              : NULL
##  $ axis.ticks.r                  : NULL
##  $ axis.minor.ticks.x.top        : NULL
##  $ axis.minor.ticks.x.bottom     : NULL
##  $ axis.minor.ticks.y.left       : NULL
##  $ axis.minor.ticks.y.right      : NULL
##  $ axis.minor.ticks.theta        : NULL
##  $ axis.minor.ticks.r            : NULL
##  $ axis.ticks.length             : 'simpleUnit' num 2.5points
##   ..- attr(*, "unit")= int 8
##  $ axis.ticks.length.x           : NULL
##  $ axis.ticks.length.x.top       : NULL
##  $ axis.ticks.length.x.bottom    : NULL
##  $ axis.ticks.length.y           : NULL
##  $ axis.ticks.length.y.left      : NULL
##  $ axis.ticks.length.y.right     : NULL
##  $ axis.ticks.length.theta       : NULL
##  $ axis.ticks.length.r           : NULL
##  $ axis.minor.ticks.length       : 'rel' num 0.75
##  $ axis.minor.ticks.length.x     : NULL
##  $ axis.minor.ticks.length.x.top   : NULL
##  $ axis.minor.ticks.length.x.bottom: NULL
##  $ axis.minor.ticks.length.y     : NULL
##  $ axis.minor.ticks.length.y.left  : NULL
##  $ axis.minor.ticks.length.y.right : NULL
##  $ axis.minor.ticks.length.theta   : NULL
##  $ axis.minor.ticks.length.r     : NULL
##  $ axis.line                     : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ axis.line.x                   : NULL
##  $ axis.line.x.top               : NULL
##  $ axis.line.x.bottom            : NULL
##  $ axis.line.y                   : NULL
##  $ axis.line.y.left              : NULL
##  $ axis.line.y.right             : NULL
##  $ axis.line.theta               : NULL
##  $ axis.line.r                   : NULL
##  $ legend.background             : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.margin                 : 'margin' num [1:4] 5points 5points 
5points 5points
##   ..- attr(*, "unit")= int 8
##  $ legend.spacing                : 'simpleUnit' num 10points
##   ..- attr(*, "unit")= int 8
##  $ legend.spacing.x              : NULL
##  $ legend.spacing.y              : NULL
##  $ legend.key                    : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.key.size               : 'simpleUnit' num 1.2lines
##   ..- attr(*, "unit")= int 3
##  $ legend.key.height               : NULL
##  $ legend.key.width                : NULL
##  $ legend.key.spacing              : 'simpleUnit' num 5points
##   ..- attr(*, "unit")= int 8
##  $ legend.key.spacing.x            : NULL
##  $ legend.key.spacing.y            : NULL
##  $ legend.frame                    : NULL
##  $ legend.ticks                    : NULL
##  $ legend.ticks.length             : 'rel' num 0.2
##  $ legend.axis.line                : NULL
##  $ legend.text                     :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : 'rel' num 0.8
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ legend.text.position            : NULL
##  $ legend.title                    :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ legend.title.position           : NULL
##  $ legend.position                 : chr "right"
##  $ legend.position.inside          : NULL
##  $ legend.direction                : NULL
##  $ legend.byrow                    : NULL
##  $ legend.justification            : chr "center"
##  $ legend.justification.top        : NULL
##  $ legend.justification.bottom     : NULL
##  $ legend.justification.left       : NULL
##  $ legend.justification.right      : NULL
##  $ legend.justification.inside     : NULL
##  $ legend.location                 : NULL
##  $ legend.box                      : NULL
##  $ legend.box.just                 : NULL
##  $ legend.box.margin               : 'margin' num [1:4] 0cm 0cm 0cm 0cm
##   ..- attr(*, "unit")= int 1
##  $ legend.box.background           : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.box.spacing              : 'simpleUnit' num 10points
##   ..- attr(*, "unit")= int 8
##   [list output truncated]
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi TRUE
##  - attr(*, "validate")= logi TRUE
```

```
ggraph(mydata, layout = "stress") +
  geom_edge_link(aes(width = frequency), color = "grey", alpha = 0.7) +
  scale_edge_width(range = c(0.2, 2)) +
  geom_node_point(aes(color = department), size = 4, alpha = 0.8) +
  scale_color_manual(values = c("maroon", "blue", "green", "yellow",
"purple", "violet")) +
  geom_node_text(aes(label = department), repel = TRUE, size = 3,
               fontface = "bold", color = "black") +
  theme_void()
```



Including Plots

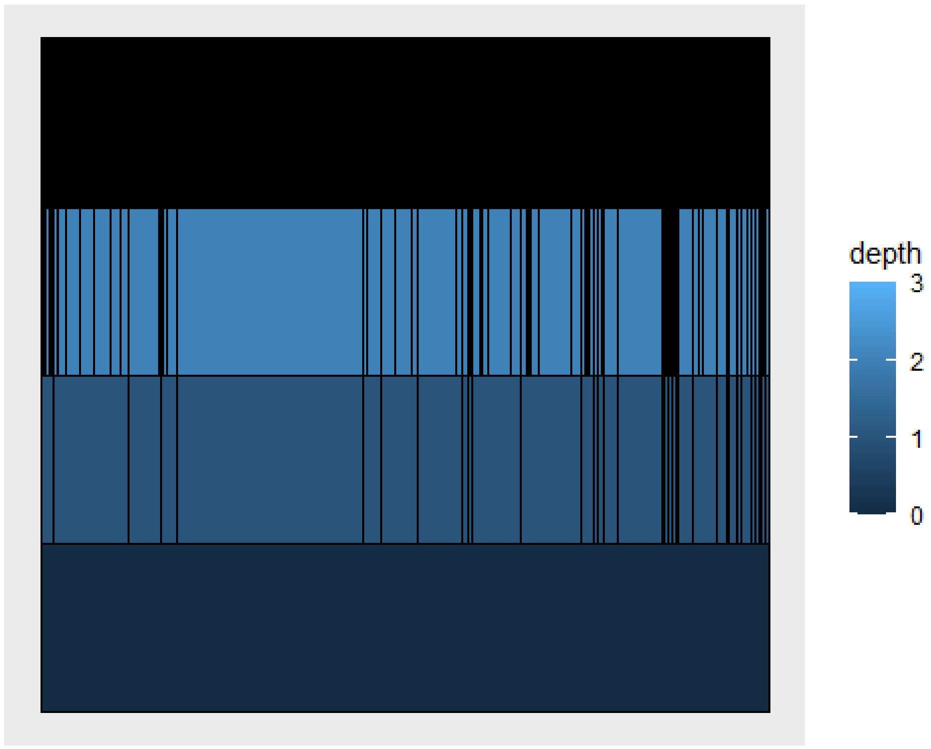
You can also embed plots, for example:

```r
mydata1 <- read.csv("C:/Users/vamsitha/Downloads/COPlants_Magnoliopsida.csv",
stringsAsFactors=FALSE)
mydata2 <- select(mydata1, Class, Order, Family, Genus)
#view(status)

data2.edges <- map_df(2:ncol(mydata2), ~select(mydata2, all_of(.x-1):.x) %>%
  setNames(c("from", "to"))) %>%
  distinct()
```

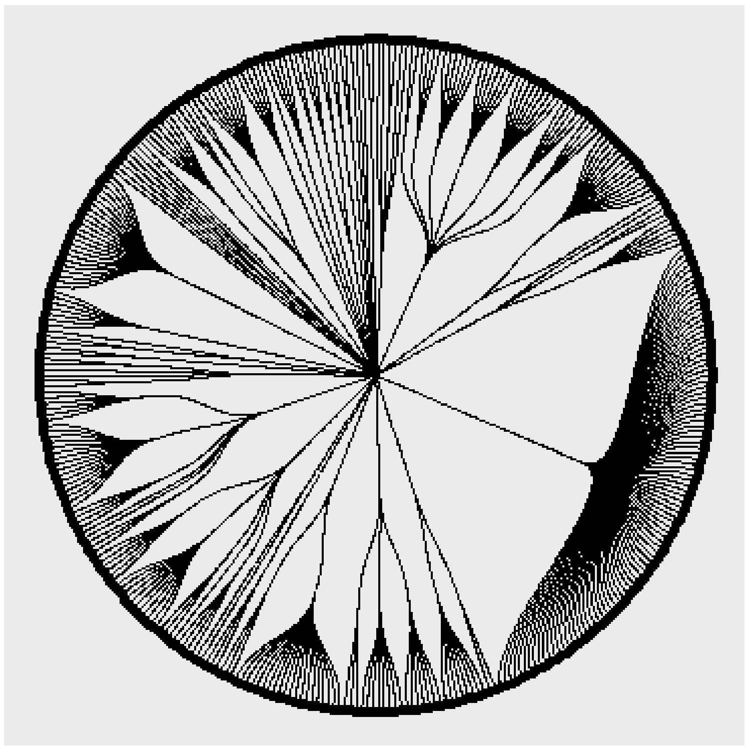
```
## Warning: Using an external vector in selections was deprecated in
tidyselect 1.1.0.
## ℹ Please use `all_of()` or `any_of()` instead.
##   # Was:
##   data %>% select(.x)
##
##   # Now:
##   data %>% select(all_of(.x))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
mydata2graph <- as_tbl_graph(data2.edges)
mydata2graph
```

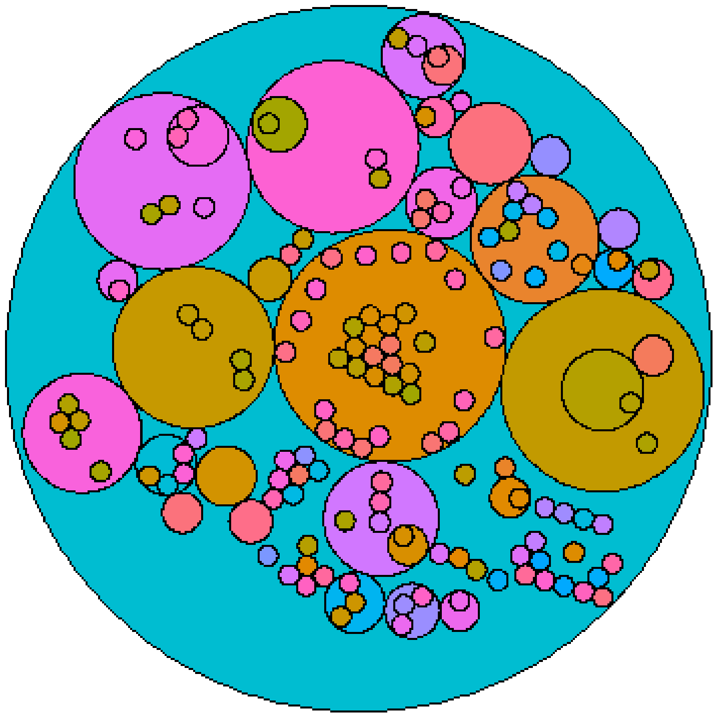
```
## # A tbl_graph: 477 nodes and 476 edges
## #
## # A rooted tree
## #
## # Node Data: 477 × 1 (active)
##    name
##    <chr>
##  1 Magnoliopsida
##  2 Sapindales
##  3 Caryophyllales
##  4 Apiales
##  5 Gentianales
##  6 Asterales
##  7 Ranunculales
##  8 Lamiales
##  9 Capparales
## 10 Campanulales
## # ℹ 467 more rows
## #
## # Edge Data: 476 × 2
##    from    to
##   <int> <int>
## 1     1     2
## 2     1     3
## 3      1      4
## # ℹ 473 more rows

mydata2 <- as_tbl_graph(mydata2) %>%
  activate(nodes) %>%
  left_join(mydata2, by = c("name" = "Family"))
ggraph(mydata2graph, layout = 'partition') +
  geom_node_tile(aes(fill=depth))
```



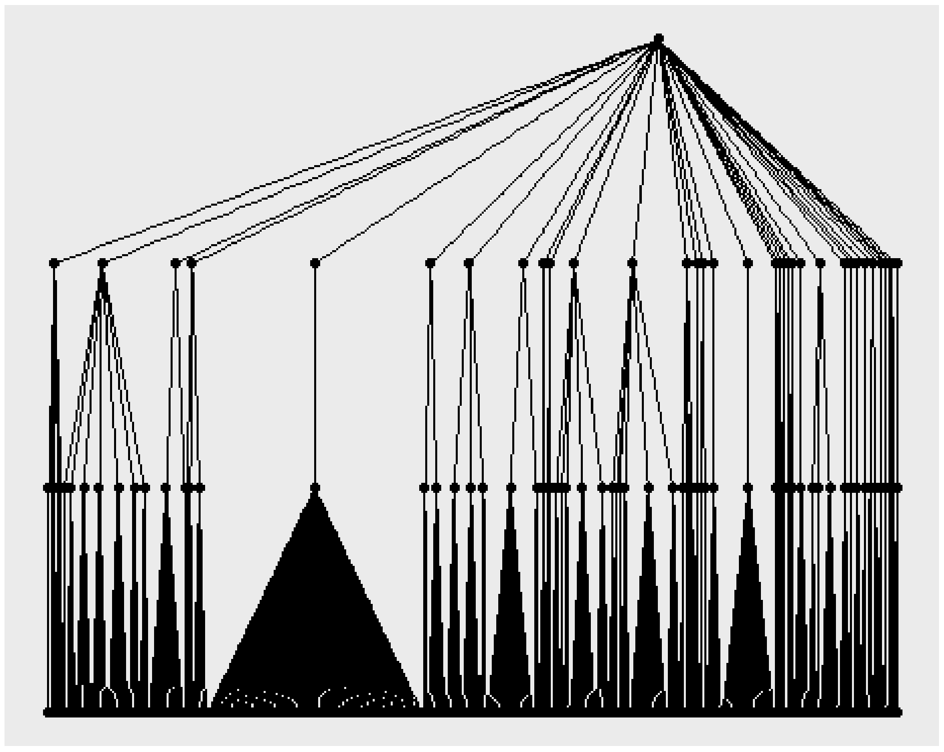
```
ggraph(mydata2graph, layout = 'dendrogram', circular = TRUE) +
  geom_edge_diagonal() +
  geom_node_point(aes(filter = leaf)) +
  coord_fixed()
```



```
mydata2.circlepack <- create_layout(mydata2graph, layout = 'circlepack')

ggraph(mydata2.circlepack) +
  geom_node_circle(aes(fill = name)) +
  theme_void() +
  coord_equal() +
  theme(legend.position = "none")
```



```r
ggraph(mydata2graph, layout = 'tree') +
    geom_edge_parallel() +
    geom_node_point()
```



Steps to Create the Visualizations:

1. Data Loading: First, load the necessary data from Excel files (or other data sources) using the `read_excel` function from the `readxl` package. The data typically consists of nodes (such as departments or employees) and edges (such as emails or interactions).

2. Data Preparation: Format the data into a suitable structure for graph processing. This involves specifying which columns represent nodes and edges, and whether the graph is directed.

3. Graph Creation: Use the `tbl_graph` function from the `tidygraph` package to create a graph object from the prepared data.

4. Visualization: Apply the `ggraph` function to the graph object, specify the layout and aesthetics (like colors, sizes, and labels), and add graphical elements (nodes, edges, texts) to visualize the graph.

5. Customization: Customize the appearance of the graph using themes, color gradients, and scales. Set the graph layout based on the type of visualization desired (e.g., stress, dendrogram, tree).

6. Output: Render the visualization and save or display the output as needed.

Description of Visualizations:

1. Network Graph:

- Purpose: Shows interactions between entities (e.g., departments or employees).

- Elements: Nodes represent entities, colored by attributes like department. Edges represent interactions, with thickness and color varying by frequency of interactions.

- Aesthetics: Uses colors to differentiate departments, edge thickness to indicate interaction frequency, and labels to identify nodes.

2. Stress Layout Visualization:

- Purpose: Emphasizes the placement of nodes to reduce stress in the network, helping to highlight the structure of interactions.

- Elements: Similar to the network graph but laid out to minimize visual stress and overlap of nodes and edges.

3. Dendrogram (Circular and Linear):

   - Purpose: Visualizes hierarchical relationships between entities, useful for understanding grouping and hierarchy.

   - Elements: Nodes are arranged in hierarchical levels with connecting lines. The circular dendrogram wraps this layout around a circle to save space and improve readability.

4. Tree Layout:

   - Purpose: Displays hierarchical data with a clear parent-child relationship, typically used to show lineage or decision paths.

   - Elements: Each node is connected to its subordinates by lines, clearly showing the flow from top to bottom or vice versa.

5. Circle Pack Layout:

   - Purpose: Visually groups nodes tightly within circles to reflect nested structures, useful for representing hierarchical data where each level of the hierarchy is enclosed within a circle.

   - Elements: Nodes are represented as circles, where each circle's size can be indicative of a metric like frequency, and nesting shows the hierarchy.