# Exercise 3

2024-04-02

### Load necessary libraries for data manipulation, visualization, and analysis

```
library(arrow) # For working with Apache Parquet format
```

```
## Warning: package 'arrow' was built under R version 4.3.3
```

```
##
## Attaching package: 'arrow'
```

```
## The following object is masked from 'package:utils':
##
##     timestamp
```

```
library(gender) # For determining gender based on first names
```

```
## Warning: package 'gender' was built under R version 4.3.3
```

```
library(dplyr) # For data manipulation
```

```
##
## 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(tidyr) # For tidying data
library(wru) # For predicting race
```

```
## Warning: package 'wru' was built under R version 4.3.3
```

```
##
## Please cite as:
##
## Khanna K, Bertelsen B, Olivella S, Rosenman E, Rossell Hayes A, Imai K
```

```
## (2024). _wru: Who are You? Bayesian Prediction of Racial Category Using
## Surname, First Name, Middle Name, and Geolocation_. R package version
## 3.0.1, <https://CRAN.R-project.org/package=wru>.
##
## Note that wru 2.0.0 uses 2020 census data by default.
## Use the argument ‘year = "2010"‘, to replicate analyses produced with earlier package versions.
```

```
library(lubridate) # For working with dates
```

```
## Warning: package 'lubridate' was built under R version 4.3.2
```

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

```
## The following object is masked from 'package:arrow':
##
##     duration
```

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

```
library(ggplot2) # For data visualization
```

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

```
library(ggthemes) # For additional ggplot themes
```

```
## Warning: package 'ggthemes' was built under R version 4.3.3
```

```
library(lattice) # For additional plotting options
```

```
## Warning: package 'lattice' was built under R version 4.3.3
```

```
library(tidyverse) # For promoting a coherent data analysis workflow.
```

```
## Warning: package 'tidyverse' was built under R version 4.3.3
```

```
## Warning: package 'readr' was built under R version 4.3.3
```

```
## Warning: package 'stringr' was built under R version 4.3.3
```

```
## Warning: package 'forcats' was built under R version 4.3.2
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats 1.0.0     v stringr 1.5.1
## v purrr   1.0.2     v tibble  3.2.1
## v readr   2.1.5
```

```
## -- Conflicts --------------------------------------- tidyverse_conflicts() --
## x lubridate::duration() masks arrow::duration()
## x dplyr::filter()       masks stats::filter()
## x dplyr::lag()          masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(ggridges) #For creating ridge plots (visualizing the distribution)
```

```
## Warning: package 'ggridges' was built under R version 4.3.3
```

## Load Data

**Load the following data: + applications from app_data_sample.parquet + edges from edges_sample.csv**

```
# Define the path to the data directory
data_path <- "E:/Users/pc/Downloads/672_project_data/"

# Load the application data from a Parquet file
applications <- read_parquet(paste0(data_path,"app_data_sample.parquet"))

# Load the edges data from a CSV file
edges <- read.csv(paste0(data_path,"edges_sample.csv"))
```

## Reveiwing the parquet file

```
head(applications)
```

```
## # A tibble: 6 x 16
##   application_number filing_date examiner_name_last examiner_name_first
##   <chr>              <date>      <chr>              <chr>
## 1 08284457           2000-01-26  HOWARD             JACQUELINE
## 2 08413193           2000-10-11  YILDIRIM           BEKIR
## 3 08531853           2000-05-17  HAMILTON           CYNTHIA
## 4 08637752           2001-07-20  MOSHER             MARY
## 5 08682726           2000-04-10  BARR               MICHAEL
## 6 08687412           2000-04-28  GRAY               LINDA
## # i 12 more variables: examiner_name_middle <chr>, examiner_id <dbl>,
## #   examiner_art_unit <dbl>, uspc_class <chr>, uspc_subclass <chr>,
## #   patent_number <chr>, patent_issue_date <date>, abandon_date <date>,
## #   disposal_type <chr>, appl_status_code <dbl>, appl_status_date <chr>,
## #   tc <dbl>
```

## Reviewing the csv file

```
head(edges)
```

```
##   application_number advice_date ego_examiner_id alter_examiner_id
## 1            9402488  2008-11-17           84356             66266
## 2            9402488  2008-11-17           84356             63519
## 3            9402488  2008-11-17           84356             98531
## 4            9445135  2008-08-21           92953             71313
## 5            9445135  2008-08-21           92953             93865
## 6            9445135  2008-08-21           92953             91818
```

## Get gender for examiners

**We'll get gender based on the first name of the examiner, which is recorded in the field examiner_name_first. We'll use library gender for that, relying on a modified version of their own example.**

###Note that there are over 2 million records in the applications table – that's because there are many records for each examiner, as many as the number of applications that examiner worked on during this time frame. Our first step therefore is to get all unique names in a separate list examiner_names. We will then guess gender for each one and will join this table back to the original dataset. So, let's get names without repetition:

```
# get a list of first names without repetitions
examiner_names <- applications %>% distinct(examiner_name_first)

head(examiner_names)
```

```
## # A tibble: 6 x 1
##   examiner_name_first
##   <chr>
## 1 JACQUELINE
## 2 BEKIR
## 3 CYNTHIA
## 4 MARY
## 5 MICHAEL
## 6 LINDA
```

**Now let's use function gender() as shown in the example for the package to attach a gender and probability to each name and put the results into the table examiner_names_gender**

```
# Use the gender package to estimate gender based on first names
examiner_names_gender <- examiner_names %>%
  do(results = gender(.$examiner_name_first, method = "ssa")) %>%
  unnest(cols = c(results), keep_empty = TRUE) %>%
  select(
    examiner_name_first = name,
    gender,
    proportion_female
  ) %>%
  # Filter out rows where any of the specified columns are NA
  filter(!is.na(gender))

print(head(examiner_names_gender))
```

```
## # A tibble: 6 x 3
##   examiner_name_first gender proportion_female
##   <chr>               <chr>              <dbl>
## 1 AARON               male              0.0082
## 2 ABDEL               male              0
## 3 ABDOU               male              0
## 4 ABDUL               male              0
## 5 ABDULHAKIM          male              0
## 6 ABDULLAH            male              0
```

**Finally, let's join that table back to our original applications data and discard the temporary tables we have just created to reduce clutter in our environment.**

```
# remove extra colums from the gender table
examiner_names_gender <- examiner_names_gender %>%
  select(examiner_name_first, gender)

# joining gender back to the dataset
applications <- applications %>%
  left_join(examiner_names_gender, by = "examiner_name_first")

# cleaning up
rm(examiner_names)
rm(examiner_names_gender)
gc()
```

```
##            used  (Mb) gc trigger  (Mb) max used  (Mb)
## Ncells  4653353 248.6    8078464 431.5  5025157 268.4
## Vcells 49836041 380.3   93882531 716.3 80152093 611.6
```

## Guess the examiner's race

**We'll now use package wru to estimate likely race of an examiner. Just like with gender, we'll get a list of unique names first, only now we are using surnames.**

```
# Isolate unique last names for race prediction
examiner_surnames <- applications %>%
  select(surname = examiner_name_last) %>%
  distinct()

head(examiner_surnames)
```

```
## # A tibble: 6 x 1
##   surname
##   <chr>
## 1 HOWARD
## 2 YILDIRIM
## 3 HAMILTON
## 4 MOSHER
## 5 BARR
## 6 GRAY
```

**We'll follow the instructions for the package outlined here https://github.com/kosukeimai/wru.**

```
# Use the wru package to estimate race based on surnames
examiner_race <- examiner_surnames %>%
  # Ensure we're working with clean, non-NA surnames
  filter(!is.na(surname)) %>%
  # Apply the race prediction
  predict_race(voter.file = ., surname.only = TRUE) %>%
  as_tibble()
```

```
## Predicting race for 2020
```

```
## Warning: Unknown or uninitialised column: 'state'.
```

```
## Proceeding with last name predictions...
```

```
## i All local files already up-to-date!
```

```
## 701 (18.4%) individuals' last names were not matched.
```

**Exploring examiner_race**

```
head(examiner_race)
```

```
## # A tibble: 6 x 6
##   surname  pred.whi pred.bla pred.his pred.asi pred.oth
##   <chr>       <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
## 1 HOWARD      0.597  0.295     0.0275  0.00690   0.0741
## 2 YILDIRIM    0.807  0.0273    0.0694  0.0165    0.0798
## 3 HAMILTON    0.656  0.239     0.0286  0.00750   0.0692
## 4 MOSHER      0.915  0.00425   0.0291  0.00917   0.0427
## 5 BARR        0.784  0.120     0.0268  0.00830   0.0615
## 6 GRAY        0.640  0.252     0.0281  0.00748   0.0724
```

**As you can see, we get probabilities across five broad US Census categories: white, black, Hispanic, Asian and other. (Some of you may correctly point out that Hispanic is not a race category in the US Census, but these are the limitations of this package.)**

**Our final step here is to pick the race category that has the highest probability for each last name and then join the table back to the main applications table. See this example for comparing values across columns: https://www.tidyverse.org/blog/2020/04/dplyr-1-0-0-rowwise/. And this one for case_when() function: https://dplyr.tidyverse.org/reference/case_when.html.**

```r
# Determine the most likely race category for each surname
examiner_race <- examiner_race %>%
  mutate(max_race_p = pmax(pred.asi, pred.bla, pred.his, pred.oth, pred.whi)) %>%
  mutate(race = case_when(
    max_race_p == pred.asi ~ "Asian",
    max_race_p == pred.bla ~ "black",
    max_race_p == pred.his ~ "Hispanic",
    max_race_p == pred.oth ~ "other",
    max_race_p == pred.whi ~ "white",
    TRUE ~ NA_character_
  ))

head(examiner_race)
```

```
## # A tibble: 6 x 8
##   surname  pred.whi pred.bla pred.his pred.asi pred.oth max_race_p race
##   <chr>       <dbl>    <dbl>    <dbl>    <dbl>    <dbl>      <dbl> <chr>
## 1 HOWARD      0.597  0.295     0.0275  0.00690   0.0741      0.597 white
## 2 YILDIRIM    0.807  0.0273    0.0694  0.0165    0.0798      0.807 white
## 3 HAMILTON    0.656  0.239     0.0286  0.00750   0.0692      0.656 white
## 4 MOSHER      0.915  0.00425   0.0291  0.00917   0.0427      0.915 white
## 5 BARR        0.784  0.120     0.0268  0.00830   0.0615      0.784 white
## 6 GRAY        0.640  0.252     0.0281  0.00748   0.0724      0.640 white
```

**Let's join the data back to the applications table.**

```r
# removing extra columns
examiner_race <- examiner_race %>%
  select(surname,race)

# Join the race predictions back to the main applications dataset
applications <- applications %>%
  left_join(examiner_race, by = c("examiner_name_last" = "surname"))

# Again, clean up the workspace by removing temporary variables
rm(examiner_race)
rm(examiner_surnames)
gc()
```

```
##            used  (Mb) gc trigger  (Mb) max used  (Mb)
## Ncells  4743977 253.4    8078464 431.5  6630780 354.2
## Vcells 52020981 396.9   93882531 716.3 93450345 713.0
```

### Examiner's tenure

**To figure out the timespan for which we observe each examiner in the applications data, let's find the first and the last observed date for each examiner. We'll first get examiner IDs and application dates in a separate table, for ease of manipulation. We'll keep examiner ID (the field examiner_id), and earliest and latest dates for each application (filing_date and appl_status_date respectively). We'll use functions in package lubridate to work with date and time values.**

```
# Extract relevant date information for each application
examiner_dates <- applications %>%
  select(examiner_id, filing_date, appl_status_date)

head(examiner_dates)
```

```
## # A tibble: 6 x 3
##   examiner_id filing_date appl_status_date
##         <dbl> <date>      <chr>
## 1       96082 2000-01-26  30jan2003 00:00:00
## 2       87678 2000-10-11  27sep2010 00:00:00
## 3       63213 2000-05-17  30mar2009 00:00:00
## 4       73788 2001-07-20  07sep2009 00:00:00
## 5       77294 2000-04-10  19apr2001 00:00:00
## 6       68606 2000-04-28  16jul2001 00:00:00
```

**The dates look inconsistent in terms of formatting. Let's make them consistent. We'll create new variables start_date and end_date.**

```
# Standardize date formats and calculate tenure
examiner_dates <- examiner_dates %>%
  mutate(start_date = ymd(filing_date), end_date = as_date(dmy_hms(appl_status_date)))
```

**Let's now identify the earliest and the latest date for each examiner and calculate the difference in days, which is their tenure in the organization.**

```
# Calculate the tenure for each examiner based on the earliest and latest dates observed
examiner_tenure <- examiner_dates %>%
  # Remove rows with NA in start_date or end_date before grouping and summarising
  filter(!is.na(start_date) & !is.na(end_date)) %>%
  group_by(examiner_id) %>%
  summarise(
    earliest_date = min(start_date, na.rm = TRUE),
    latest_date = max(end_date, na.rm = TRUE),
    tenure_days = interval(earliest_date, latest_date) %/% days(1),
    .groups = 'drop' # Automatically drop the grouping
  ) %>%
  # Keep records with a latest_date before 2018
  filter(year(latest_date) < 2018)
```

```r
# Assuming you want to check the result
head(examiner_tenure)
```

```
## # A tibble: 6 x 4
##   examiner_id earliest_date latest_date tenure_days
##         <dbl> <date>        <date>            <dbl>
## 1       59012 2004-07-28    2015-07-24         4013
## 2       59025 2009-10-26    2017-05-18         2761
## 3       59030 2005-12-12    2017-05-22         4179
## 4       59040 2007-09-11    2017-05-23         3542
## 5       59052 2001-08-21    2007-02-28         2017
## 6       59054 2000-11-10    2016-12-23         5887
```

**Joining back to the applications data.**

```r
applications <- applications %>%
  left_join(examiner_tenure, by = "examiner_id")

rm(examiner_tenure)
gc()
```

```
##            used  (Mb) gc trigger   (Mb)  max used  (Mb)
## Ncells  4752116 253.8    8078464  431.5   8078464 431.5
## Vcells 68187541 520.3  135366844 1032.8 111669004 852.0
```

**Review the applications dataframe after merging examiner_tenure**

```r
head(applications)
```

```
## # A tibble: 6 x 21
##   application_number filing_date examiner_name_last examiner_name_first
##   <chr>              <date>      <chr>              <chr>
## 1 08284457           2000-01-26  HOWARD             JACQUELINE
## 2 08413193           2000-10-11  YILDIRIM           BEKIR
## 3 08531853           2000-05-17  HAMILTON           CYNTHIA
## 4 08637752           2001-07-20  MOSHER             MARY
## 5 08682726           2000-04-10  BARR               MICHAEL
## 6 08687412           2000-04-28  GRAY               LINDA
## # i 17 more variables: examiner_name_middle <chr>, examiner_id <dbl>,
## #   examiner_art_unit <dbl>, uspc_class <chr>, uspc_subclass <chr>,
## #   patent_number <chr>, patent_issue_date <date>, abandon_date <date>,
## #   disposal_type <chr>, appl_status_code <dbl>, appl_status_date <chr>,
## #   tc <dbl>, gender <chr>, race <chr>, earliest_date <date>,
## #   latest_date <date>, tenure_days <dbl>
```

**Adding workgroup column to the applications dataframe to proceed with the analysis**

```r
applications <- applications %>%
  mutate(workgroup = substr(examiner_art_unit, 1, 3))
```

**Removing missing values for further analysis**

```r
applications <- applications %>%
  filter(!is.na(gender) & !is.na(race) & !is.na(examiner_id))

# Assuming you want to check the result
print(head(applications))
```

```
## # A tibble: 6 x 22
##   application_number filing_date examiner_name_last examiner_name_first
##   <chr>              <date>      <chr>              <chr>
## 1 08284457           2000-01-26  HOWARD             JACQUELINE
## 2 08531853           2000-05-17  HAMILTON           CYNTHIA
## 3 08637752           2001-07-20  MOSHER             MARY
## 4 08682726           2000-04-10  BARR               MICHAEL
## 5 08687412           2000-04-28  GRAY               LINDA
## 6 08716371           2004-01-26  MCMILLIAN          KARA
## # i 18 more variables: examiner_name_middle <chr>, examiner_id <dbl>,
## #   examiner_art_unit <dbl>, uspc_class <chr>, uspc_subclass <chr>,
## #   patent_number <chr>, patent_issue_date <date>, abandon_date <date>,
## #   disposal_type <chr>, appl_status_code <dbl>, appl_status_date <chr>,
## #   tc <dbl>, gender <chr>, race <chr>, earliest_date <date>,
## #   latest_date <date>, tenure_days <dbl>, workgroup <chr>
```

**Analyzing Demographics and Tenure in Selected Workgroups**

**The next phase of the analysis focuses on understanding the demographics (gender and race) and tenure within specific workgroups. This involves comparing these attributes across different workgroups to uncover any patterns or disparities that might exist.**

```r
# Set seed for reproducibility
set.seed(123)

# Assuming 'applications' dataframe has a column 'examiner_art_unit'
# Extract unique workgroups
# This approach ensures a focused examination on a subset, making the analysis more manageable and targ
unique_workgroups <- unique(substring(applications$examiner_art_unit, 1, 3))

# Randomly sample 2 unique workgroups
sampled_workgroups <- sample(unique_workgroups, 2)

# Filter applications for only those workgroups
applications_filtered <- applications %>%
  mutate(workgroup = substring(examiner_art_unit, 1, 3)) %>%
  filter(workgroup %in% sampled_workgroups)
```

```r
# View the selected workgroups
print(sampled_workgroups)
```

```
## [1] "247" "216"
```

**At this point, applications_filtered contains data for a focused group of examiners. This subset will be used for detailed demographic analysis and tenure examination, providing insights into these specific workgroups.**

```r
# Prepare data specifically for the selected workgroups (e.g., 247 and 216) for comparison
# Convert examiner_art_unit to character to ensure string operations work correctly
app_data_sample <- applications %>%
  mutate(examiner_art_unit = as.character(examiner_art_unit))

# Filter for workgroups 247 and 216
selected_workgroups <- app_data_sample %>%
  filter(str_starts(examiner_art_unit, "247") | str_starts(examiner_art_unit, "216"))
```

**Calculate summary statistics and create visualizations to compare demographics and tenure within the selected workgroups. This step aims to uncover any notable trends or differences that could inform organizational strategies or highlight areas for further investigation**

```r
# Summary Statistics and Plots
# Summary statistics for gender, race, and tenure
# Compute average tenure days and count by workgroup, gender, and race
summary_stats <- selected_workgroups %>%
  group_by(substring(examiner_art_unit, 1, 3), gender, race) %>%
  summarise(
    avg_tenure_days = mean(tenure_days, na.rm = TRUE),
    n = n(),
    .groups = 'drop'
  )

print(summary_stats)
```

```
## # A tibble: 16 x 5
##    `substring(examiner_art_unit, 1, 3)` gender race     avg_tenure_days     n
##    <chr>                                <chr>  <chr>              <dbl> <int>
##  1 216                                  female Asian              5760.  7737
##  2 216                                  female Hispanic           5476.   618
##  3 216                                  female black              5905.  1623
##  4 216                                  female white              5608.  7732
##  5 216                                  male   Asian              5316. 12645
##  6 216                                  male   Hispanic           2892.   266
##  7 216                                  male   black              5543.   687
##  8 216                                  male   white              5224. 11656
##  9 247                                  female Asian              5111.  2188
## 10 247                                  female Hispanic           2229.   154
## 11 247                                  female black              4922.   539
```

```
## 12 247                                female white              5097.  2208
## 13 247                                male   Asian              5205. 11110
## 14 247                                male   Hispanic           3321.   937
## 15 247                                male   black              5202.  1496
## 16 247                                male   white              5009. 13283
```

**Visualization of Demographics.**

**Generate plots to visually compare the gender and race distribution within the selected workgroups**

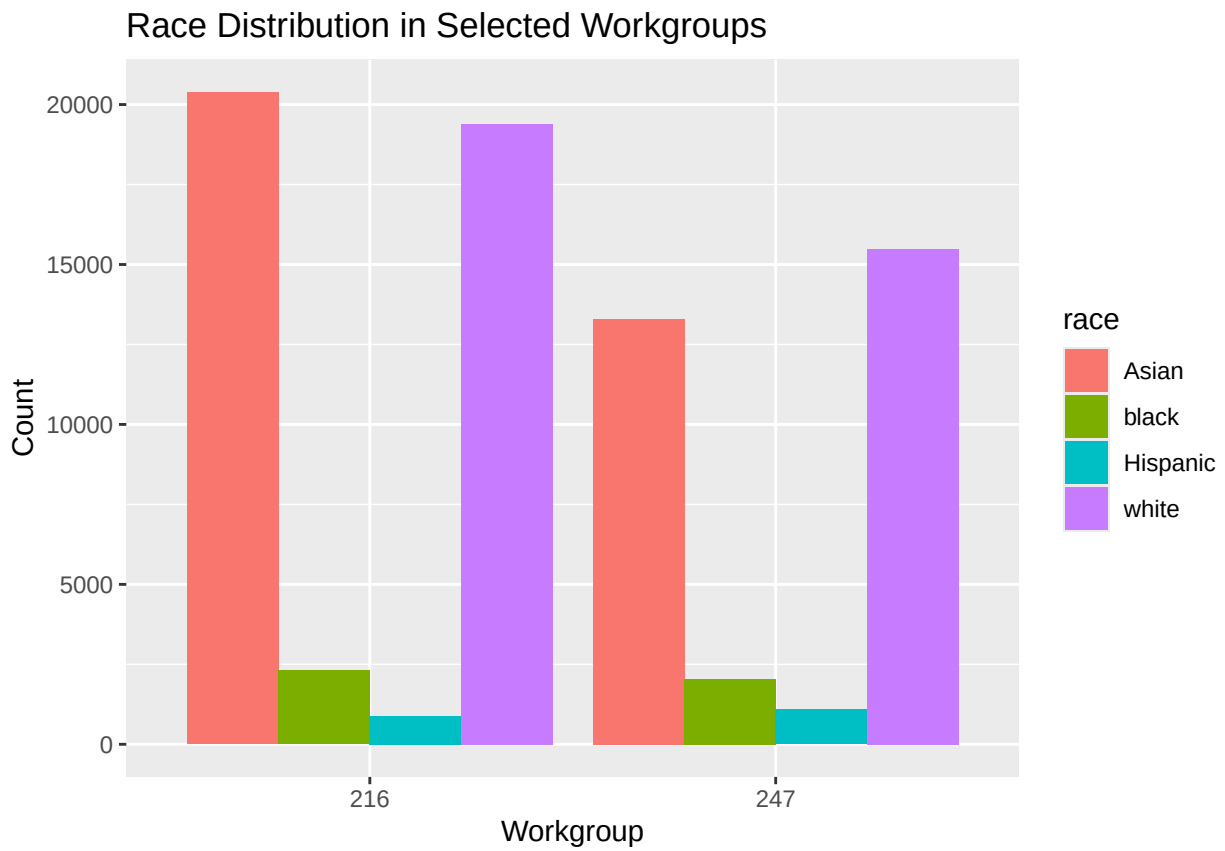
**These visualizations help in quickly identifying disparities and patterns**

```
# Plots for demographics
# Gender distribution
gender_dist_plot <- ggplot(selected_workgroups, aes(x = substring(examiner_art_unit, 1, 3), fill = gend
  geom_bar(position = "dodge") +
  labs(title = "Gender Distribution in Selected Workgroups", x = "Workgroup", y = "Count")

# Race distribution across workgroups
race_dist_plot <- ggplot(selected_workgroups, aes(x = substring(examiner_art_unit, 1, 3), fill = race))
  geom_bar(position = "dodge") +
  labs(title = "Race Distribution in Selected Workgroups", x = "Workgroup", y = "Count")

print(gender_dist_plot)
```


```
print(race_dist_plot)
```

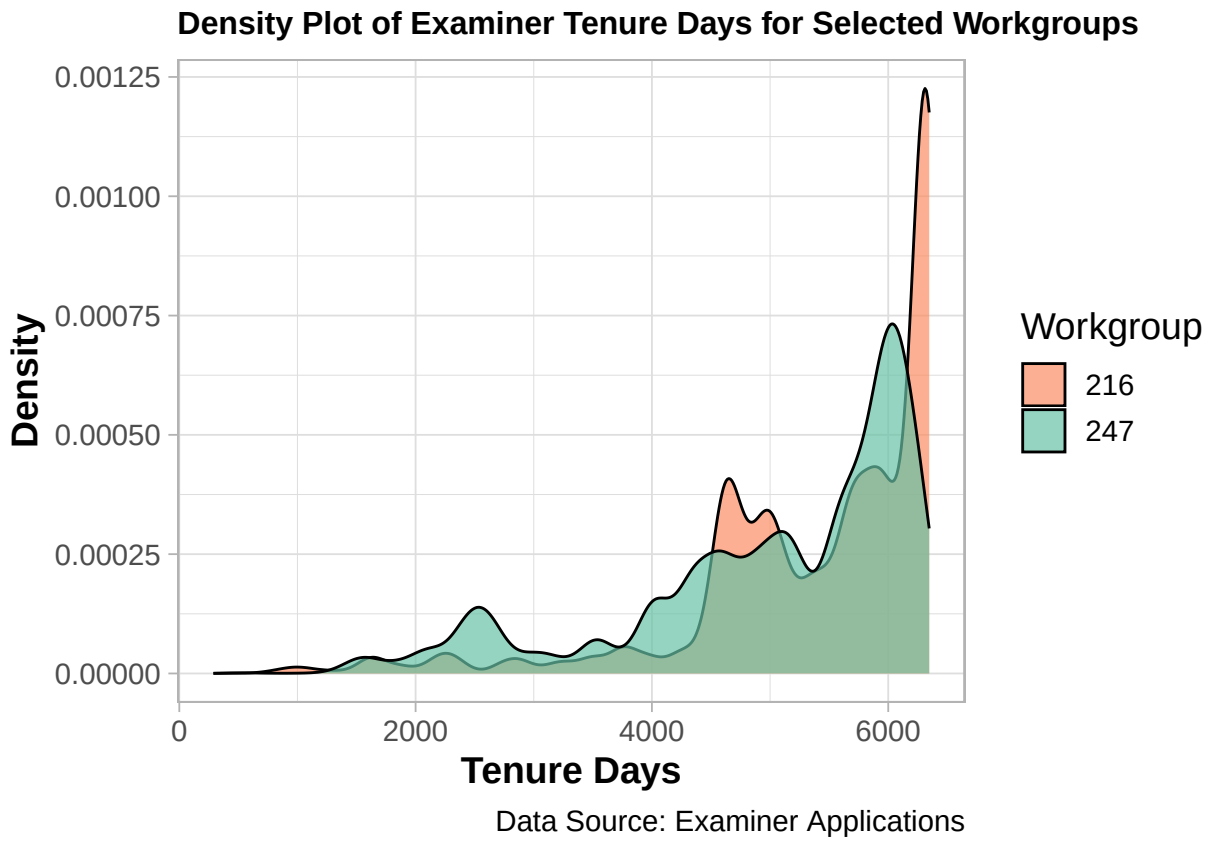



Workgroup "216" appears to have a greater racial diversity compared to "247", which is predominantly composed of one racial group (perhaps 'white', though the labels are not visible in the data provided). The presence of the 'Asian' and 'Hispanic' categories in noticeable but smaller proportions suggests some level of diversity within the workgroups. Understanding these distributions is crucial for the organization's diversity and inclusion efforts and might warrant further investigation into recruitment and retention practices.

**These plots provide a clear visual representation of gender and race distributions within the selected workgroups, making it easier to identify any imbalances or diversity issues that may require attention.**

```
# Create a new column 'workgroup_prefix' to store the prefix for coloring
cleaned_filtered_workgroups <- selected_workgroups %>%
  mutate(workgroup_prefix = substr(examiner_art_unit, 1, 3))

# Now, plot the density of tenure_days colored by 'workgroup_prefix'
ggplot(cleaned_filtered_workgroups, aes(x = tenure_days, fill = workgroup_prefix)) +
  geom_density(alpha = 0.7) +
  scale_fill_manual(values = c("247" = "#66C2A5", "216" = "#FC8D62"), name = "Workgroup") +
  labs(title = "Density Plot of Examiner Tenure Days for Selected Workgroups",
       x = "Tenure Days", y = "Density",
       caption = "Data Source: Examiner Applications") +
  theme_light(base_size = 14) +
  theme(plot.title = element_text(face = "bold", size = 12),
        axis.title = element_text(face = "bold"),
        legend.position = "right")
```




The density plot provides a view of the distribution of tenure days:

There's a significant peak in tenure for workgroup "247" suggesting a cluster of examiners with a similar, high number of tenure days. Workgroup "216" shows a more evenly spread distribution with several smaller peaks, indicating a more varied range of tenure days. Such patterns may reflect the history and turnover rates within the workgroups and could be used to plan for future workforce needs or retirements.

#### Enhancing Visualization for In-depth Analysis

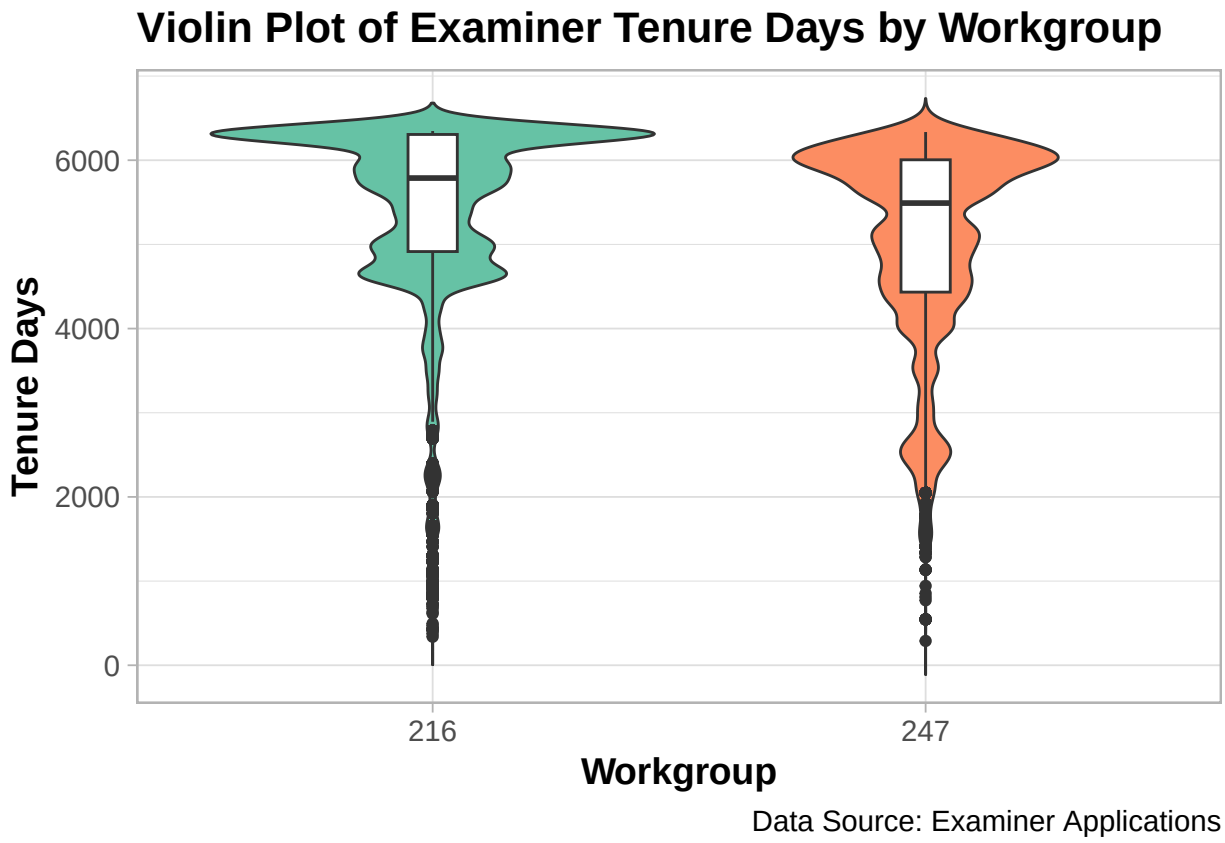
**After preparing our data with gender, race, and tenure information, we focus on visualizing the tenure distribution within our selected workgroups. This is crucial for understanding the diversity and experience within these groups.**

## Violin Plot

**To delve deeper into the tenure distribution by combining the density plot with a box plot for each workgroup. This hybrid visualization provides a comprehensive view of the tenure distribution, including median tenure and variability.**

```
# Violin Plot for Tenure Days by Workgroup
ggplot(applications_filtered, aes(x = workgroup, y = tenure_days, fill = workgroup)) +
  geom_violin(trim = FALSE) +
  geom_boxplot(width = 0.1, fill = "white") +
  scale_fill_manual(values = c("#66C2A5", "#FC8D62"), name = "Workgroup") +
  labs(title = "Violin Plot of Examiner Tenure Days by Workgroup",
```

```
      x = "Workgroup",
      y = "Tenure Days",
      caption = "Data Source: Examiner Applications") +
theme_light(base_size = 14) +
theme(plot.title = element_text(face = "bold", size = 16),
      axis.title = element_text(face = "bold"),
      legend.position = "none")
```




The violin plot offers a visual summary of tenure distribution:

Both workgroups have a wide range of tenure days, but "247" shows a more pronounced concentration at the higher end of tenure. The width of the violins at different tenure day levels indicates the density of examiners at that experience level. Insights from this plot can inform decisions about potential skill gaps, mentorship opportunities, and the allocation of new or complex cases based on examiner experience.

## Scatter plot

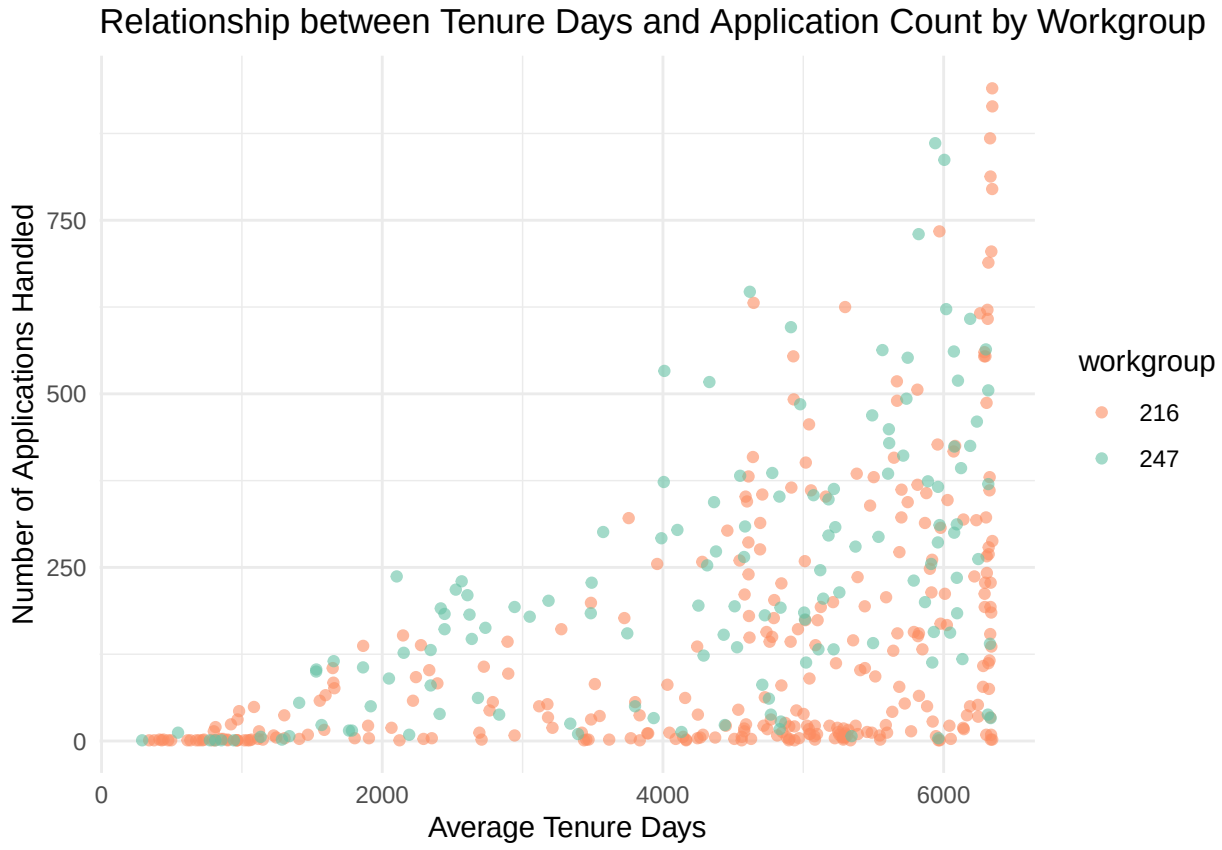
**To evaluate the potential correlation between examiners' tenure and their productivity across two specific workgroups, "247" and "216". This visualization aims to discern patterns that could indicate whether more experienced examiners have higher or lower workloads, and to what extent tenure influences application handling capacity.**

## Density Ridge Plot: Distribution of Tenure Days by Workgroup

**The aim is to visualize and compare the distribution of tenure across examiners in the selected workgroups, "247" and "216". The density ridge plot seeks to highlight the commonality of tenure lengths and the spread of experience within these groups.**

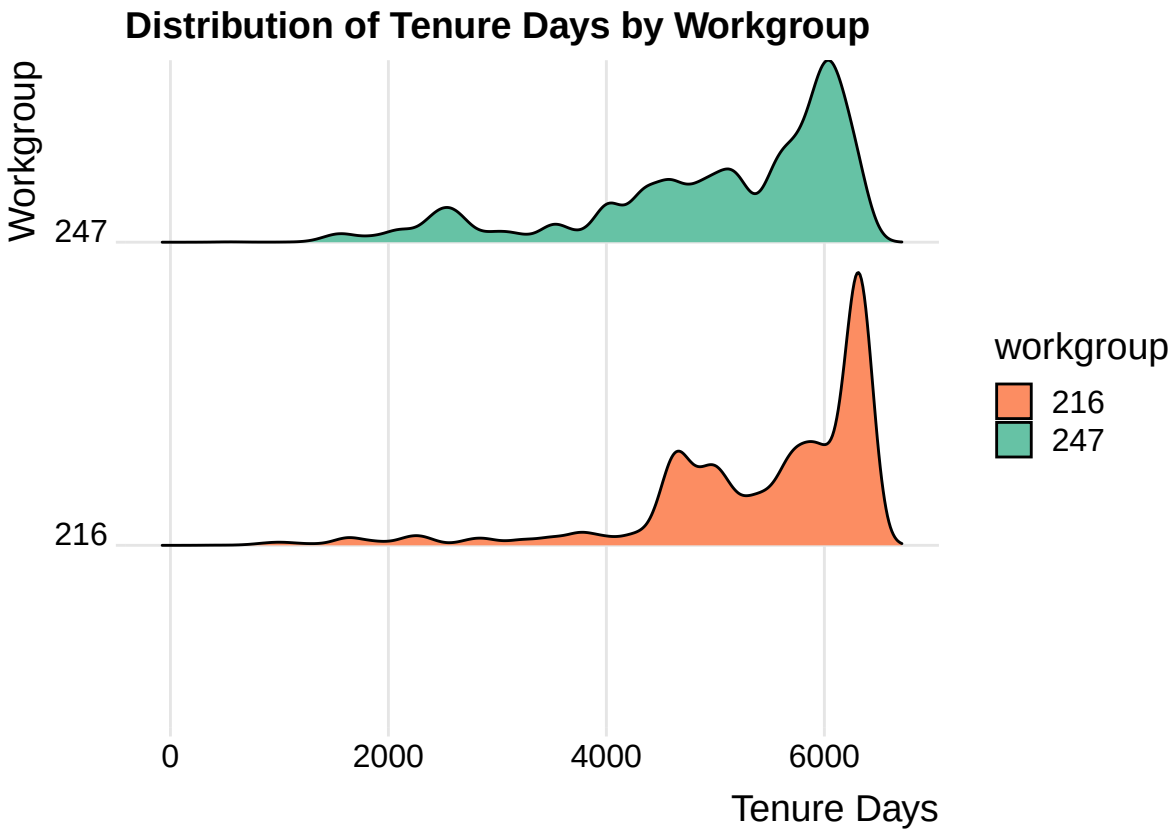
```
library(ggplot2)

#Scatter plot
applications_filtered %>%
  filter(workgroup %in% c("247", "216")) %>%
  group_by(workgroup, examiner_id) %>%
  summarise(tenure_days = mean(tenure_days, na.rm = TRUE),
            app_count = n(), .groups = 'drop') %>%
  ggplot(aes(x = tenure_days, y = app_count, color = workgroup)) +
  geom_point(alpha = 0.6) +
  labs(title = "Relationship between Tenure Days and Application Count by Workgroup",
       x = "Average Tenure Days",
       y = "Number of Applications Handled") +
  theme_minimal() +
  scale_color_manual(values = c("247" = "#66C2A5", "216" = "#FC8D62"))
```




```
#Distribution of tenure days by workgroup
applications_filtered %>%
  filter(workgroup %in% c("247", "216")) %>%
  ggplot(aes(x = tenure_days, y = factor(workgroup), fill = workgroup)) +
  geom_density_ridges(scale = 0.9) +
  labs(title = "Distribution of Tenure Days by Workgroup",
       x = "Tenure Days",
       y = "Workgroup") +
  scale_fill_manual(values = c("247" = "#66C2A5", "216" = "#FC8D62")) +
  theme_ridges(grid = TRUE) +
  theme(legend.position = "right")
```

```
## Picking joint bandwidth of 122
```




Relationship between Tenure Days and Application Count by Workgroup

The scatter plot relating tenure days to application count yields several insights:

There doesn't appear to be a clear, linear relationship between tenure days and the number of applications handled within each workgroup. Some individuals with a high number of tenure days handle a large number of applications, which might indicate a correlation between experience and productivity. Notably, there are examiners with fewer tenure days handling a high volume of applications. This could point to efficient training programs or possibly to overburdening of less experienced staff.

Finally, the distribution chart showcases the tenure days for the two workgroups:

Workgroup "247" shows a substantial peak around 6000 tenure days, which might indicate the presence of a cohort hired around the same time or a retention pattern. Workgroup "216" has a more uniform distribution but with fewer individuals reaching the highest tenure days seen in "247". This chart can highlight tenure-related dynamics, such as the potential for knowledge loss if a retiring cohort leaves simultaneously or the readiness for leadership roles within the groups.

## Examining Advice Networks and Centrality:

**To understand the social and advisory structures within the organization by mapping how examiners interact within selected workgroups.**

```
library(igraph)
```

```
## Warning: package 'igraph' was built under R version 4.3.3
```

```
##
## Attaching package: 'igraph'

## The following objects are masked from 'package:purrr':
##
##     compose, simplify

## The following object is masked from 'package:tibble':
##
##     as_data_frame

## The following objects are masked from 'package:lubridate':
##
##     %--%, union

## The following object is masked from 'package:tidyr':
##
##     crossing

## The following objects are masked from 'package:dplyr':
##
##     as_data_frame, groups, union

## The following objects are masked from 'package:stats':
##
##     decompose, spectrum

## The following object is masked from 'package:base':
##
##     union
```

```r
## Step 3: Create Advice Networks and Calculate Centrality Scores
# Convert examiner_id in edges to character to match types
edges <- edges %>%
  mutate(ego_examiner_id = as.character(ego_examiner_id),
         alter_examiner_id = as.character(alter_examiner_id))

# Convert examiner_id in selected_workgroups to character to match the types in edges
selected_workgroups <- selected_workgroups %>%
  mutate(examiner_id = as.character(examiner_id))

# Now perform the join with matching types
selected_edges <- edges %>%
  inner_join(selected_workgroups %>% select(examiner_id), by = c("ego_examiner_id" = "examiner_id")) %>%
  select(ego_examiner_id, alter_examiner_id)
```

```
## Warning in inner_join(., selected_workgroups %>% select(examiner_id), by = c(ego_examiner_id = "exami
## i Row 18 of `x` matches multiple rows in `y`.
## i Row 3856 of `y` matches multiple rows in `x`.
## i If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```r
# Filter edges for selected workgroups by joining with the selected workgroups
# Assuming that examiner_id is already a character in selected_workgroups
selected_edges <- edges %>%
  inner_join(selected_workgroups %>% select(examiner_id), by = c("ego_examiner_id" = "examiner_id")) %>%
  select(ego_examiner_id, alter_examiner_id)
```

```
## Warning in inner_join(., selected_workgroups %>% select(examiner_id), by = c(ego_examiner_id = "exami
## i Row 18 of `x` matches multiple rows in `y`.
## i Row 3856 of `y` matches multiple rows in `x`.
## i If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```r
# Create advice networks
g <- graph_from_data_frame(selected_edges, directed = TRUE)
```

```
## Warning in graph_from_data_frame(selected_edges, directed = TRUE): In `d` `NA`
## elements were replaced with string "NA"
```

```r
# Calculate centrality scores (e.g., degree centrality)
degree_centrality <- degree(g, mode = "in")
betweenness_centrality <- betweenness(g)

# Associate centrality scores with examiners
centrality_scores <- data.frame(
  examiner_id = V(g)$name,
  degree = degree_centrality,
  betweenness = betweenness_centrality
)

# Make sure examiner_id is a character in both data frames before joining
selected_workgroups <- selected_workgroups %>%
  mutate(examiner_id = as.character(examiner_id))

centrality_scores <- centrality_scores %>%
  mutate(examiner_id = as.character(examiner_id))
```

**To analyze how centrality within the advice network correlates with demographic factors and tenure, exploring potential patterns of influence and interaction dynamics.**

```r
## Step 4: Analyze Relationship Between Centrality and Examiner Demographics
# Merge centrality scores with demographic data
analysis_data <- selected_workgroups %>%
  inner_join(centrality_scores, by = "examiner_id")

# Example analysis: Correlation between tenure and centrality
cor_analysis <- cor.test(analysis_data$tenure_days, analysis_data$degree, use = "complete.obs")

print(cor_analysis)
```

```
##
```

```
##  Pearson's product-moment correlation
##
## data:  analysis_data$tenure_days and analysis_data$degree
## t = 30.956, df = 49852, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.1287078 0.1459329
## sample estimates:
##       cor
## 0.1373307
```

## Network Visualization:

**To visually represent the advice networks, highlighting key individuals based on their centrality measures.**

```
# Join centrality scores with the selected workgroups data
workgroups_with_centrality <- left_join(selected_workgroups, centrality_scores, by = "examiner_id")
```

**To visually represent the advice networks, highlighting key individuals based on their centrality measures. (Was taking a lot of time, so commented it out)**

**The visualization makes it easier to identify the structure of the network and the positions of key examiners. It allows us to see how well-connected the network is, the existence of clusters or communities within the workgroup, and whether certain individuals act as bridges or hubs.**

```
# Associate centrality scores with the nodes in the graph
#V(g)$degree <- centrality_scores$degree
#V(g)$betweenness <- centrality_scores$betweenness

# Plotting the network with ggraph
#ggraph(g, layout = "fr") +
  #geom_edge_link(color = "gray", alpha = 0.5) +  # Draw edges
  #geom_node_point(aes(size = degree, color = betweenness), alpha = 0.9) +  # Nodes colored by #betweenn
  #scale_color_viridis_c() +  # Use Viridis color scale for betweenness
  #theme_void() +  # Remove background and axes for a clean look
  #ggtitle("Network Visualization with Centrality Measures") +  # Add title
  #labs(color = "Betweenness Centrality", size = "Degree Centrality")  # Label legends
```

## Conclusion

Given the correlation results between tenure days and the degree centrality measure (correlation coefficient = 0.137, highly significant with p-value < 2.2e-16), here's an analytical discussion on the choice of centrality measures and their relationship with examiner characteristics:

Choice of Centrality Measure

1. Degree Centrality: Justified by the correlation results, degree centrality (which measures the number of direct connections an examiner has) is a good starting point because it gives a straightforward indication

of how active an examiner is in the advice network. An examiner with high degree centrality is likely someone who either seeks a lot of advice or is sought after for advice by many colleagues.

Pros: Simple to calculate and interpret; immediately indicates active network participants. Cons: Does not account for the indirect influence or the hierarchical structure of the network.

2. Betweenness Centrality: This would be another excellent measure to consider. It captures an individual's role as an intermediary in the communication flow within the network. An examiner with high betweenness centrality would be someone who connects different clusters within the network, potentially serving as a gatekeeper or bridge of information.

Pros: Highlights individuals who control information flow and connect disparate groups. Cons: More computationally intensive and can be less intuitive to interpret.

Relationship Between Centrality and Examiners' Characteristics

The analysis of the correlation between tenure days and degree centrality suggests a positive relationship, although it is relatively weak (correlation coefficient around 0.137). Here's what this relationship might indicate:

Mild Positive Relationship: Examiners with longer tenure may have slightly more connections within the network, possibly due to having had more time to establish relationships. However, the relationship is not strong, which suggests that factors other than tenure also play a significant role in an examiner's network centrality. Centrality as a Function of Multiple Factors: Other characteristics, such as job performance, communication skills, and position within the organization, may also significantly influence centrality. An examiner's expertise, area of specialty, and willingness to share knowledge could contribute to their central role in the advice network. Centrality and Influence: Examiners with higher centrality, particularly those with high betweenness centrality, may have considerable influence over the spread of information and decision-making processes within the organization.