INDIVIDUAL ASSIGNMENT

Programming For Data Analysis

APD2F2206CS(CYB)

ASSIGNED DATE : Week 2

HAND IN DATE : Week 9

WEIGHTAGE : 50%

LECTURER : MINNU HELEN JOSEPH

STUDENT NAME : CHANG SHIAU HUEI

TP NUMBER : TP060322

TABLE OF CONTENTS

1.0 INTRODUCTION

A grocery store chain in Vancouver noticed that their employees were being fired or resigned at an alarming rate in all of their locations. As a result, the human resource department manager assigned me to conduct an analysis on the supplied dataset to determine the reasons why their employees were being terminated quickly in order to provide them with helpful information for decision-making.

1.1 - Assumptions

- Since there are no names of the employees available to indicate their unique identity, it is assumed that each Employee ID represents one employee.
- The skipped Employee ID values indicate employees that were terminated prior to the creation of this dataset.
- “Layoff” termination type indicates that the employees were terminated involuntarily, whereas “Resignation” and “Retirement” termination types imply that the employees were terminated voluntarily.
- All employees remain at 1 store only, indicating that if an employee was transferred from one store to another, only the latest store in which he or she works will be regarded in the “data” dataset.
- All employees that has the status as “Active” indicates that they are not yet terminated. Since active employees still have a termination date assigned to them, this date will be filtered out later in the analysis.

2.0 IMPORT DATA

2.1 - Import Dataset

The initial step in any data analysis is to import the provided data set. The data set provided (employee_attrition.csv) is in a Comma-Separated Values (CSV) file, which is a delimited text file in which values are separated by a comma. In order for the data to be used, the data set is first loaded into the program as the attribute “raw_data”. To make sure that the contents of the data would not be changed accidentally, it is also assigned into the “data” attribute.

```
raw_data = read.csv("C:\\Users\\shiau\\OneDrive - Asia Pacific University\\DEGREE\\Y2\\PFDA\\ASSIGNMENT\\employee_attrition.csv", header=TRUE)
View(raw_data)

data <- raw_data
```

Code: Importing “employee_attrition.csv”

	EmployeeID	recorddate_key	birthdate_key	orighiredate_key	terminationdate_key	age	length_of_service	city_name	department_name	job_title
1	1318	12/31/2006 0:00	1/3/1954	8/28/1989	1/1/1900	52	17	Vancouver	Executive	CEO
2	1318	12/31/2007 0:00	1/3/1954	8/28/1989	1/1/1900	53	18	Vancouver	Executive	CEO
3	1318	12/31/2008 0:00	1/3/1954	8/28/1989	1/1/1900	54	19	Vancouver	Executive	CEO
4	1318	12/31/2009 0:00	1/3/1954	8/28/1989	1/1/1900	55	20	Vancouver	Executive	CEO
5	1318	12/31/2010 0:00	1/3/1954	8/28/1989	1/1/1900	56	21	Vancouver	Executive	CEO
6	1318	12/31/2011 0:00	1/3/1954	8/28/1989	1/1/1900	57	22	Vancouver	Executive	CEO
7	1318	12/31/2012 0:00	1/3/1954	8/28/1989	1/1/1900	58	23	Vancouver	Executive	CEO
8	1318	12/31/2013 0:00	1/3/1954	8/28/1989	1/1/1900	59	24	Vancouver	Executive	CEO
9	1318	12/31/2014 0:00	1/3/1954	8/28/1989	1/1/1900	60	25	Vancouver	Executive	CEO
10	1318	12/31/2015 0:00	1/3/1954	8/28/1989	1/1/1900	61	26	Vancouver	Executive	CEO
11	1319	12/31/2006 0:00	1/3/1957	8/28/1989	1/1/1900	49	17	Vancouver	Executive	VP Stores

store_name	gender_short	gender_full	termreason_desc	termtype_desc	STATUS_YEAR	STATUS	BUSINESS_UNIT
35	M	Male	Not Applicable	Not Applicable	2006	ACTIVE	HEADOFFICE
35	M	Male	Not Applicable	Not Applicable	2007	ACTIVE	HEADOFFICE
35	M	Male	Not Applicable	Not Applicable	2008	ACTIVE	HEADOFFICE
35	M	Male	Not Applicable	Not Applicable	2009	ACTIVE	HEADOFFICE
35	M	Male	Not Applicable	Not Applicable	2010	ACTIVE	HEADOFFICE
35	M	Male	Not Applicable	Not Applicable	2011	ACTIVE	HEADOFFICE
35	M	Male	Not Applicable	Not Applicable	2012	ACTIVE	HEADOFFICE
35	M	Male	Not Applicable	Not Applicable	2013	ACTIVE	HEADOFFICE
35	M	Male	Not Applicable	Not Applicable	2014	ACTIVE	HEADOFFICE
35	M	Male	Not Applicable	Not Applicable	2015	ACTIVE	HEADOFFICE
35	F	Female	Not Applicable	Not Applicable	2006	ACTIVE	HEADOFFICE

Output: Data in “employee_attrition.csv”

2.2 - Import Packages

Necessary packages are installed and loaded into the program.

```
# 2.2 Import packages
# To check exactly what packages have I used throughout the code
install.packages("NCmisc")
library(NCmisc)
list.functions.in.file("C:\\Users\\shiau\\OneDrive - Asia Pacific University\\DEGREE\\Y2\\PFDA\\ASSIGNMENT\\CHANG SHIAU HUEI_TP060322.R", alphabetic = TRUE)

install.packages("janitor")
library(janitor)

library(dplyr)

install.packages("ggplot2")
library(ggplot2)

install.packages("viridis")
library("viridis")

install.packages("plotrix")
library(plotrix)

install.packages("plotly")
library(plotly)

install.packages("ggcorrplot")
library(ggcorrplot)
```

Code: Imported Packages

NCmisc package stands for Miscellaneous Functions for Creating Adaptive Functions and Scripts. I used it to check exactly what packages have I used throughout the code after I have finished with all the analysis.

Janitor package is mainly used for data cleaning, data exploration, and also helps arranging data into tables.

Dplyr package is mainly used for data exploration, data manipulation and also data transformation.

Ggplot2 package, Plotrix package and Plotly package are mainly used for data visualization. They provide a lot of plots, labeling, axis and color scaling functions.

Viridis package consists of color maps for data visualisation to improve graph readability for readers who suffer from common forms of colour blindness or colour vision deficiency. (Garnier, 2021)

Ggcorrplot package provides a reordering solution for the correlation matrix and displays the significance level on the correlogram. It also includes a function for generating a matrix of correlation p-values. (STHDA, 2022)

3.0 DATA CLEANING

Data cleaning is the process of the detection, the modification and the removal of inaccurate or incomplete raw data. For this dataset, we will remove redundant columns, check for duplicate records, examine the data for empty values, as well as filter the data to acquire the latest records.

3.1 - Remove Redundant Columns

gender_short	gender_full
M	Male
M	Male
M	Male
M	Male
M	Male
M	Male
M	Male
M	Male
M	Male
M	Male
F	Female

Figure: "gender_short" and "gender_full" Columns

In Section 2.1, we could see that the data had two columns that represent gender. Hence, the code below was used to remove one of the redundant columns which is "gender_short".

```
data$gender_short = NULL
```

Code: Removing "gender_short" Column

3.2 - Check for Duplicate Records

Since it is assumed that every Employee ID is unique and that each Record Date should be different, duplicate records are detected by using get_dupes( ) to see whether any rows have the exact same values for every field.

```
get_dupes(data)
```

Code: Check for Duplicate Records

```
> get_dupes(data)
No variable names specified - using all columns.

No duplicate combinations found of: EmployeeID, recorddate_key, birthdate_key, orighiredate_key, terminationdate_key,
 age, length_of_service, city_name, department_name, ... and 8 other variables
 [1] EmployeeID          recorddate_key      birthdate_key       orighiredate_key
 [5] terminationdate_key age                 length_of_service   city_name
 [9] department_name     job_title           store_name          gender_full
[13] termreason_desc     termtype_desc       STATUS_YEAR         STATUS
[17] BUSINESS_UNIT       dupe_count
<0 rows> (or 0-length row.names)
```

Output: No Duplicate Combinations Found

The output implies that no duplicates combinations were found, hence we do not have to remove any rows.

3.3 - Check for Empty Columns or Rows (Excluding "Not Applicable")

The code below is used to check for empty data by calculating the sums of blank (“ “) columns. However, we excluded the “Not Applicable" data in the search since we would need it for the analysis later.

```
colSums(data == "")
```

Code: Check for Empty Columns or Rows

```
> colSums(data == "")
         EmployeeID      recorddate_key       birthdate_key    orighiredate_key terminationdate_key
                  0                   0                   0                   0                   0
                age   length_of_service         city_name     department_name           job_title
                  0                   0                   0                   0                   0
         store_name         gender_full     termreason_desc       termtype_desc         STATUS_YEAR
                  0                   0                   0                   0                   0
             STATUS       BUSINESS_UNIT
                  0                   0
```

Output: No Empty Columns or Rows

From the output, we can conclude that no empty columns or rows were found.

3.4 - Filter the Latest Record of Employees

NOTE: This section is done after Section 4.0 to avoid redundancy in code.

As previously stated, it is assumed that every Employee ID is unique. Since it is evident from Section 2.1 that most Employee IDs had more than 1 record, this indicates that the data consisted of both old and new entries. Before filtering the latest record entries, the current data is assigned to the "data_full" variable for subsequent analysis. The most recent data is then filtered so that each Employee ID only has 1 occurrence.

```
# Save full data into another variable
data_full <- data

# 3.4 - Only taking the latest record of employees
data = data %>%
  group_by(EMPLOYEE_ID) %>%
  dplyr::arrange(desc(RECORD_DATE),.by_group = TRUE) %>%
  distinct(EMPLOYEE_ID,.keep_all = TRUE)
## desc() -> Latest records come first

## Making sure it is a data frame to avoid data being 4 different classes
## grouped_df" "tbl_df"     "tbl"        "data.frame"
data <- data.frame(data)
```

Code: Filter the Latest Record of Employees

	EMPLOYEE_ID
1	1318
2	1319
3	1320
4	1321
5	1322
6	1323
7	1325
8	1328
9	1329
10	1330

Output: Every EmployeeID Only has 1 Occurrence in "data"

From the output, we can see that every EmployeeID only has one occurrence in "data". After this step, we would now have 3 data sets:

DATA SET	DETAILS
raw_data	Original data that has not been pre-processed nor cleaned
data_full	Cleaned and pre-processed data that consists of both old and latest records
data	Cleaned and pre-processed data that consists of only the latest records

4.0 PRE-PROCESSING DATA

Data pre-processing is the process of transforming data in order to prepare them for subsequent analysis. For this dataset, we will modify the column headings, correcting some data values, and change the datatype of the data.

4.1 - Change Column Names

4.1.1 - Clean Column Names

```
> colnames(data) # View column headings
 [1] "EmployeeID"          "recorddate_key"      "birthdate_key"       "orighiredate_key"    "terminationdate_key" "age"                 "length_of_service"
 [8] "city_name"           "department_name"     "job_title"           "store_name"          "gender_full"         "termreason_desc"     "termtype_desc"
[15] "STATUS_YEAR"         "STATUS"              "BUSINESS_UNIT"
```

Output: All Column Headers of the Dataset

The current column names are inconsistent, with some in uppercase and others in lowercase. clean_names() is used to ensure that all column names have the same casing, which is all capital letters. The dataframe with the new column headers would be then assigned to the "data" variable.

```
colnames(data) # View column headings
data <- clean_names(data, case="all_caps")
```

Code: Clean Column Names

```
> colnames(data)
 [1] "EMPLOYEE_ID"         "RECORDDATE_KEY"      "BIRTHDATE_KEY"       "ORIGHIREDATE_KEY"    "TERMINATIONDATE_KEY"
 [6] "AGE"                 "LENGTH_OF_SERVICE"   "CITY_NAME"           "DEPARTMENT_NAME"     "JOB_TITLE"
[11] "STORE_NAME"          "GENDER_FULL"         "TERMREASON_DESC"     "TERMTYPE_DESC"       "STATUS_YEAR"
[16] "STATUS"              "BUSINESS_UNIT"
```

Output: All Column Headers of the Dataset After clean_names( ) is Used

From the output, we can see that all column headings are now consistent, with all being written in uppercase letters only.

4.1.2 - Change Column Names

Some column names are adjusted further to make them more understandable and to facilitate further data analysis.

```
colnames(data)[2] <- "RECORD_DATE"
colnames(data)[3] <- "BIRTH_DATE"
colnames(data)[4] <- "ORI_HIRE_DATE"
colnames(data)[5] <- "TERMINATION_DATE"
colnames(data)[8] <- "CITY"
colnames(data)[9] <- "DEPARTMENT"
colnames(data)[11] <- "STORE"
colnames(data)[12] <- "GENDER"
colnames(data)[13] <- "TERMINATION_REASON"
colnames(data)[14] <- "TERMINATION_TYPE"
colnames(data)[15] <- "RECORD_YEAR"
colnames(data)
```

Code: Changing Certain Column Names

```
> colnames(data)[2] <- "RECORD_DATE"
> colnames(data)[3] <- "BIRTH_DATE"
> colnames(data)[4] <- "ORI_HIRE_DATE"
> colnames(data)[5] <- "TERMINATION_DATE"
> colnames(data)[8] <- "CITY"
> colnames(data)[9] <- "DEPARTMENT"
> colnames(data)[11] <- "STORE"
> colnames(data)[12] <- "GENDER"
> colnames(data)[13] <- "TERMINATION_REASON"
> colnames(data)[14] <- "TERMINATION_TYPE"
> colnames(data)[15] <- "RECORD_YEAR"
> colnames(data)
 [1] "EMPLOYEE_ID"        "RECORD_DATE"        "BIRTH_DATE"         "ORI_HIRE_DATE"      "TERMINATION_DATE"   "AGE"
 [7] "LENGTH_OF_SERVICE"  "CITY"               "DEPARTMENT"         "JOB_TITLE"          "STORE"              "GENDER"
[13] "TERMINATION_REASON" "TERMINATION_TYPE"   "RECORD_YEAR"        "STATUS"             "BUSINESS_UNIT"
```

Output: Certain Column Names are changed

4.2 - Correcting Data Values

Because certain values were misspelt, the code below is used to rectify the errors.

```
data = data %>%
  mutate(CITY = ifelse(CITY=="New Westminister","New Westminster", CITY))

data = data %>%
  mutate(TERMINATION_REASON = ifelse(TERMINATION_REASON=="Resignaton","Resignation", TERMINATION_REASON))

data = data %>%
  mutate(JOB_TITLE = ifelse(JOB_TITLE=="CHief Information Officer","Chief Information Officer", JOB_TITLE))

data = data %>%
  mutate(JOB_TITLE = ifelse(JOB_TITLE=="Accounts Receiveable","Accounts Receivable", JOB_TITLE))
```

Code : Correcting Data Values

4.3 - Change Datatypes

4.3.1 - Check Datatypes

This is to ensure that the data in each column is of the right datatype before it is utilised for further analysis.

```
str(data)
```

Code: Checking Datatype of the Columns

```
> str(data)
'data.frame':   49653 obs. of  17 variables:
 $ EMPLOYEE_ID       : int  1318 1318 1318 1318 1318 1318 1318 1318 1318 1318 ...
 $ RECORD_DATE       : chr  "12/31/2006 0:00" "12/31/2007 0:00" "12/31/2008 0:00" "12/31/2009 0:00" ...
 $ BIRTH_DATE        : chr  "1/3/1954" "1/3/1954" "1/3/1954" "1/3/1954" ...
 $ ORI_HIRE_DATE     : chr  "8/28/1989" "8/28/1989" "8/28/1989" "8/28/1989" ...
 $ TERMINATION_DATE  : chr  "1/1/1900" "1/1/1900" "1/1/1900" "1/1/1900" ...
 $ AGE               : int  52 53 54 55 56 57 58 59 60 61 ...
 $ LENGTH_OF_SERVICE : int  17 18 19 20 21 22 23 24 25 26 ...
 $ CITY              : chr  "Vancouver" "Vancouver" "Vancouver" "Vancouver" ...
 $ DEPARTMENT        : chr  "Executive" "Executive" "Executive" "Executive" ...
 $ JOB_TITLE         : chr  "CEO" "CEO" "CEO" "CEO" ...
 $ STORE             : int  35 35 35 35 35 35 35 35 35 35 ...
 $ GENDER            : chr  "Male" "Male" "Male" "Male" ...
 $ TERMINATION_REASON: chr  "Not Applicable" "Not Applicable" "Not Applicable" "Not Applicable" ...
 $ TERMINATION_TYPE  : chr  "Not Applicable" "Not Applicable" "Not Applicable" "Not Applicable" ...
 $ RECORD_YEAR       : int  2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 ...
 $ STATUS            : chr  "ACTIVE" "ACTIVE" "ACTIVE" "ACTIVE" ...
 $ BUSINESS_UNIT     : chr  "HEADOFFICE" "HEADOFFICE" "HEADOFFICE" "HEADOFFICE" ...
```

Output: Datatype of the Columns

The current datatypes only consists of integer and characters, which is not ideal. The values in their respective columns should be of the following datatypes:

Column Number	Column	Datatype
1	EMPLOYEE_ID	Factor
2	RECORD_DATE	POSIXlt
3	BIRTH_DATE	Date
4	ORI_HIRE_DATE	Date
5	TERMINATION_DATE	Date
6	AGE	int
7	LENGTH_OF_SERVICE	int

8	CITY	Factor
9	DEPARTMENT	Factor
10	JOB_TITLE	Factor
11	STORE	Factor
12	GENDER	Factor
13	TERMINATION_REASON	Factor
14	TERMINATION_TYPE	Factor
15	RECORD_YEAR	int
16	STATUS	Factor
17	BUSINESS_UNIT	Factor

4.3.2 - Character to POSIXlt Datatype

The POSIXlt class stores date or time values as a list of components which makes it easy to extract certain components like hours and minutes separately. To change character datatype to POSIXlt datatype, we need to make sure that the character string is in a standard ambiguous format.

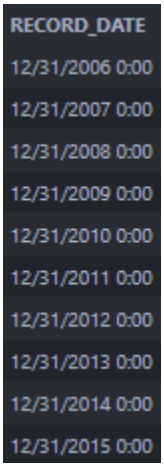


Figure: Data in "RECORD_DATE" Column

From our current data, it is shown that the date and time format for the "RECORD_DATE" column is "m/d/Y H:M". Hence, we will inform the program how the current date and time are formatted, so that it could convert the values into a POSIXlt datatype.

```
data$RECORD_DATE <- strptime(data$RECORD_DATE,format="%m/%d/%Y %H:%M")
str(data$RECORD_DATE)
```

Code: Converting into POSIXlt Datatype

From the figure, the datatype for the "RECORD_DATE" column has been successfully changed to POSIXlt datatype (stat.berkeley.edu., 2006a). However, due to all hours and minutes recorded being set to 0, POSIXlt did not display the time even though it was previously stated.

```
> data$RECORD_DATE <- strptime(data$RECORD_DATE,format="%m/%d/%Y %H:%M")
> str(data$RECORD_DATE)
 POSIXlt[1:49653], format: "2006-12-31" "2007-12-31" "2008-12-31" "2009-12-31" "2010-12-31" "2011-12
-31" ...
```

Output: Changed Format of "RECORD_DATE" Column

4.3.3 - Character to Date Datatype

To change character datatype to Date datatype, we need to make sure that character string is in a standard ambiguous format.

BIRTH_DATE	ORI_HIRE_DATE	TERMINATION_DATE
1/3/1954	8/28/1989	1/1/1900
1/3/1954	8/28/1989	1/1/1900
1/3/1954	8/28/1989	1/1/1900
1/3/1954	8/28/1989	1/1/1900
1/3/1954	8/28/1989	1/1/1900
1/3/1954	8/28/1989	1/1/1900
1/3/1954	8/28/1989	1/1/1900
1/3/1954	8/28/1989	1/1/1900
1/3/1954	8/28/1989	1/1/1900
1/3/1954	8/28/1989	1/1/1900

Figure: Data in "BIRTH_DATE", "ORI_HIRE_DATE", and "TERMINATION_DATE" Column

From our current data, it is shown that the date and time format for column 3 to 5 in the data set is "m/d/Y". Hence, we will inform the program how the current date is formatted, so that it could convert the values into a Date datatype.

```
for(i in 3:5) {
  data[,i] <- as.Date(data[,i], "%m/%d/%Y")
}
str(data[3:5])
```

Code: Converting into Date Datatype

```
> for(i in 3:5) {
+   data[,i] <- as.Date(data[,i], "%m/%d/%Y")
+ }
> str(data[3:5])
'data.frame':   49653 obs. of  3 variables:
 $ BIRTH_DATE      : Date, format: "1954-01-03" "1954-01-03" "1954-01-03" ...
 $ ORI_HIRE_DATE   : Date, format: "1989-08-28" "1989-08-28" "1989-08-28" ...
 $ TERMINATION_DATE: Date, format: "1900-01-01" "1900-01-01" "1900-01-01" ...
```

Output: Changed Format of "BIRTH_DATE", "ORI_HIRE_DATE", and "TERMINATION_DATE" Columns

From the figure, the datatype of column 3 to 5 has been successfully changed to Date datatype.

4.3.4 - Character to Factor Datatype

Because Factors are an exceptionally efficient means of storing character values, certain character datatypes will be converted to Factor datatypes. This is due to the fact that distinct character values are only recorded once and the data is kept as a vector of integers. (stat.berkeley.edu., 2006b)

CITY	DEPARTMENT	JOB_TITLE	STORE	GENDER	TERMINATION_REASON	TERMINATION_TYPE
Vancouver	Executive	CEO	35	Male	Not Applicable	Not Applicable
Vancouver	Executive	CEO	35	Male	Not Applicable	Not Applicable
Vancouver	Executive	CEO	35	Male	Not Applicable	Not Applicable
Vancouver	Executive	CEO	35	Male	Not Applicable	Not Applicable
Vancouver	Executive	CEO	35	Male	Not Applicable	Not Applicable
Vancouver	Executive	CEO	35	Male	Not Applicable	Not Applicable
Vancouver	Executive	CEO	35	Male	Not Applicable	Not Applicable
Vancouver	Executive	CEO	35	Male	Not Applicable	Not Applicable
Vancouver	Executive	CEO	35	Male	Not Applicable	Not Applicable
Vancouver	Executive	CEO	35	Male	Not Applicable	Not Applicable
Vancouver	Executive	VP Stores	35	Female	Not Applicable	Not Applicable

STATUS	BUSINESS_UNIT
ACTIVE	HEADOFFICE
ACTIVE	HEADOFFICE
ACTIVE	HEADOFFICE
ACTIVE	HEADOFFICE
ACTIVE	HEADOFFICE
ACTIVE	HEADOFFICE
ACTIVE	HEADOFFICE
ACTIVE	HEADOFFICE
ACTIVE	HEADOFFICE
ACTIVE	HEADOFFICE
ACTIVE	HEADOFFICE

Figure: Data in "CITY", "DEPARTMENT", "JOB_TITLE", "STORE", "GENDER", "TERMINATION_REASON", "TERMINATION_TYPE", "STATUS" and "BUSINESS_UNIT" Columns

```
data$EMPLOYEE_ID <- factor(data$EMPLOYEE_ID)
for(i in 8:14) {
  data[,i] <- factor(data[,i])
}
data$STATUS <- factor(data$STATUS)
data$BUSINESS_UNIT <- factor(data$BUSINESS_UNIT)
str(data)
```

Code: Converting into Factor Datatype

```
> data$EMPLOYEE_ID <- factor(data$EMPLOYEE_ID)
> for(i in 8:14) {
+   data[,i] <- factor(data[,i])
+ }
> data$STATUS <- factor(data$STATUS)
> data$BUSINESS_UNIT <- factor(data$BUSINESS_UNIT)
> str(data)
'data.frame':   49653 obs. of  17 variables:
 $ EMPLOYEE_ID       : Factor w/ 6284 levels "1318","1319",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ RECORD_DATE       : POSIXlt, format: "2006-12-31" "2007-12-31" "2008-12-31" "2009-12-31" ...
 $ BIRTH_DATE        : Date, format: "1954-01-03" "1954-01-03" "1954-01-03" "1954-01-03" ...
 $ ORI_HIRE_DATE     : Date, format: "1989-08-28" "1989-08-28" "1989-08-28" "1989-08-28" ...
 $ TERMINATION_DATE  : Date, format: "1900-01-01" "1900-01-01" "1900-01-01" "1900-01-01" ...
 $ AGE               : int  52 53 54 55 56 57 58 59 60 61 ...
 $ LENGTH_OF_SERVICE : int  17 18 19 20 21 22 23 24 25 26 ...
 $ CITY              : Factor w/ 39 levels "Abbotsford","Aldergrove",..: 34 34 34 34 34 34 34 34 34 34 ...
 $ DEPARTMENT        : Factor w/ 21 levels "Accounting","Accounts Payable",..: 10 10 10 10 10 10 10 10 10 10 ...
 $ JOB_TITLE         : Factor w/ 47 levels "Accounting Clerk",..: 9 9 9 9 9 9 9 9 9 9 ...
 $ STORE             : Factor w/ 46 levels "1","2","3","4",..: 35 35 35 35 35 35 35 35 35 35 ...
 $ GENDER            : Factor w/ 2 levels "Female","Male": 2 2 2 2 2 2 2 2 2 2 ...
 $ TERMINATION_REASON: Factor w/ 4 levels "Layoff","Not Applicable",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ TERMINATION_TYPE  : Factor w/ 3 levels "Involuntary",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ RECORD_YEAR       : int  2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 ...
 $ STATUS            : Factor w/ 2 levels "ACTIVE","TERMINATED": 1 1 1 1 1 1 1 1 1 1 ...
 $ BUSINESS_UNIT     : Factor w/ 2 levels "HEADOFFICE","STORES": 1 1 1 1 1 1 1 1 1 1 ...
```

Output: Changed Format of "CITY", "DEPARTMENT", "JOB_TITLE", "STORE", "GENDER", "TERMINATION_REASON", "TERMINATION_TYPE", "STATUS" and "BUSINESS_UNIT" Columns

From the figure, the datatype of column 8 to 14, "STATUS" and "BUSINESS_UNIT" columns has been successfully changed to Factor datatype.

5.0 DATA EXPLORATION

Exploratory Data Analysis (EDA) employs visual exploration to comprehend the contents and characteristics of a dataset.

5.1 - Viewing Dataset

5.1.1 -Viewing "data" Dataset

```
## View the first few values of the data set
glimpse(data)

## How the data is stored (Chosen by compiler)
class(data)

## Summary of the data
summary(data)
```

Code: Viewing the "data" Dataset Information

The first few values of the "data" dataset provided is viewed in the console along with the datatype of the columns. It also displays the dataset's column headers, with a total of 17 columns, as well as the number of rows in this dataset, which is 6284.

```
> glimpse(data)
Rows: 6,284
Columns: 17
$ EMPLOYEE_ID         <fct> 1318, 1319, 1320, 1321, 1322, 1323, 1325, 1328, 1329, 1330, 1331, 1332, 1334, 1335, 1338, 1339, 1340, 1341…
$ RECORD_DATE         <dttm> 2015-12-31, 2015-12-31, 2015-12-31, 2015-12-31, 2015-12-31, 2015-12-31, 2015-12-31, 2015-12-31, 2015-12-3…
$ BIRTH_DATE          <date> 1954-01-03, 1957-01-03, 1955-01-02, 1959-01-02, 1958-01-09, 1962-01-09, 1964-01-13, 1956-01-17, 1967-01-2…
$ ORI_HIRE_DATE       <date> 1989-08-28, 1989-08-28, 1989-08-28, 1989-08-28, 1989-08-31, 1989-08-31, 1989-09-02, 1989-09-05, 1989-09-0…
$ TERMINATION_DATE    <date> 1900-01-01, 1900-01-01, 1900-01-01, 1900-01-01, 1900-01-01, 1900-01-01, 1900-01-01, 1900-01-01, 1900-01-0…
$ AGE                 <int> 61, 58, 60, 56, 57, 53, 51, 59, 48, 48, 50, 60, 54, 53, 60, 65, 60, 60, 65, 65, 65, 60, 60, 65, 60, 60, 60…
$ LENGTH_OF_SERVICE   <int> 26, 26, 26, 26, 26, 26, 26, 26, 26, 26, 26, 26, 26, 26, 26, 19, 24, 19, 19, 24, 24, 24, 19, 19, 24, 19, 19, 19…
$ CITY                <fct> Vancouver, Vancouver, Vancouver, Vancouver, Vancouver, Vancouver, Vancouver, Vancouver, Terrace, Terrace, …
$ DEPARTMENT          <fct> Executive, Executive, Executive, Executive, Executive, Executive, Executive, Executive, Store Management, …
$ JOB_TITLE           <fct> "CEO", "VP Stores", "Legal Counsel", "VP Human Resources", "VP Finance", "Exec Assistant, VP Stores", "Exe…
$ STORE               <fct> 35, 35, 35, 35, 35, 35, 35, 35, 32, 32, 18, 35, 35, 35, 35, 19, 16, 35, 37, 35, 35, 37, 35, 35, 15, 35, 35…
$ GENDER              <fct> Male, Female, Female, Male, Male, Male, Female, Female, Female, Female, Female, Female, Male, Female, Male…
$ TERMINATION_REASON  <fct> Not Applicable, Not Applicable, Not Applicable, Not Applicable, Not Applicable, Not Applicable, Not Applic…
$ TERMINATION_TYPE    <fct> Not Applicable, Not Applicable, Not Applicable, Not Applicable, Not Applicable, Not Applicable, Not Applic…
$ RECORD_YEAR         <int> 2015, 2015, 2015, 2015, 2015, 2015, 2015, 2015, 2015, 2015, 2015, 2015, 2015, 2015, 2009, 2014, 2009, 2009…
$ STATUS              <fct> ACTIVE, ACTIVE, ACTIVE, ACTIVE, ACTIVE, ACTIVE, ACTIVE, ACTIVE, ACTIVE, ACTIVE, ACTIVE, ACTIVE, ACTIVE, AC…
$ BUSINESS_UNIT       <fct> HEADOFFICE, HEADOFFICE, HEADOFFICE, HEADOFFICE, HEADOFFICE, HEADOFFICE, HEADOFFICE, HEADOFFICE, STORES, ST…
```

Output: Data View in "data" Dataset

The dataset is stored in R as a data frame.

```
> class(data)
[1] "data.frame"
```

Output: Class of the "data" Dataset

The summary of all data is shown below. Each column will be discussed in further detail in the following sections.

```
> summary(data)
  EMPLOYEE_ID     RECORD_DATE                    BIRTH_DATE         ORI_HIRE_DATE        TERMINATION_DATE          AGE
 1318   :   1   Min.   :2006-01-01 00:00:00.0   Min.   :1941-01-15   Min.   :1989-08-28   Min.   :1900-01-01   Min.   :19.00
 1319   :   1   1st Qu.:2015-12-31 00:00:00.0   1st Qu.:1956-12-27   1st Qu.:1995-09-30   1st Qu.:1900-01-01   1st Qu.:32.00
 1320   :   1   Median :2015-12-31 00:00:00.0   Median :1970-01-13   Median :2000-10-05   Median :1900-01-01   Median :45.00
 1321   :   1   Mean   :2014-11-13 20:36:58.2   Mean   :1969-09-17   Mean   :2001-07-23   Mean   :1926-04-18   Mean   :44.74
 1322   :   1   3rd Qu.:2015-12-31 00:00:00.0   3rd Qu.:1982-12-06   3rd Qu.:2007-07-28   3rd Qu.:1900-01-01   3rd Qu.:58.00
 1323   :   1   Max.   :2015-12-31 00:00:00.0   Max.   :1994-12-31   Max.   :2013-12-11   Max.   :2015-12-30   Max.   :65.00
 (Other):6278
 LENGTH_OF_SERVICE             CITY                 DEPARTMENT          JOB_TITLE        STORE        GENDER
 Min.   : 0.00     Vancouver      :1392   Meats           :1252   Meat Cutter  :1218   46     : 522   Female:3278
 1st Qu.: 7.00     Victoria       : 624   Customer Service:1190   Cashier      :1158   18     : 481   Male  :3006
 Median :13.00     Nanaimo        : 481   Produce         :1060   Dairy Person :1032   21     : 403
 Mean   :12.84     New Westminster: 447   Dairy           :1033   Produce Clerk:1027   42     : 392
 3rd Qu.:19.00     Kelowna        : 305   Bakery          : 898   Baker        : 865   43     : 311
 Max.   :26.00     Kamloops       : 267   Processed Foods : 736   Shelf Stocker: 704   16     : 305
                   (Other)        :2768   (Other)         : 115   (Other)      : 280   (Other):3870
      TERMINATION_REASON       TERMINATION_TYPE  RECORD_YEAR          STATUS        BUSINESS_UNIT
 Layoff        : 215     Involuntary   : 215    Min.   :2006   ACTIVE    :4804   HEADOFFICE:  80
 Not Applicable:4804     Not Applicable:4804    1st Qu.:2015   TERMINATED:1480   STORES    :6204
 Resignation   : 382     Voluntary     :1265    Median :2015
 Retirement    : 883                            Mean   :2014
                                                3rd Qu.:2015
                                                Max.   :2015
```

Output: Summary of the "data" Dataset

5.1.2 -Viewing "data_full" Dataset

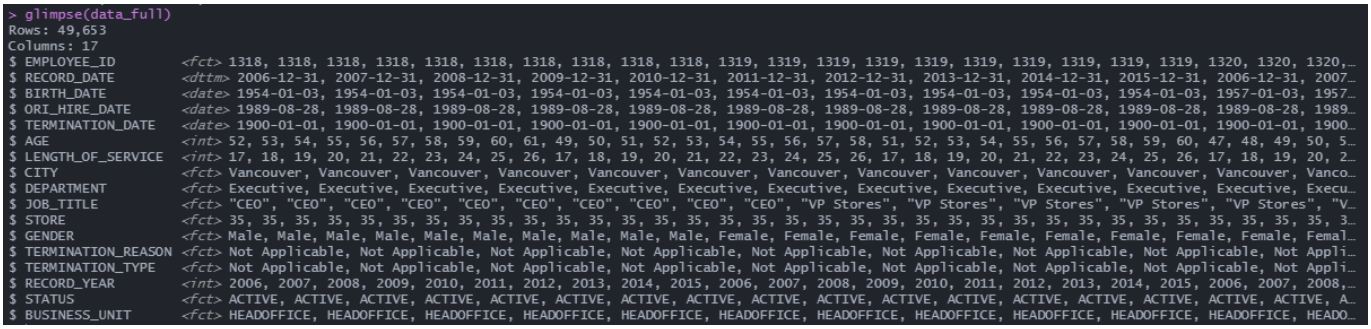
```
## View the first few values of the data set
glimpse(data_full)

## How the data is stored (Chosen by compiler)
class(data_full)

## Summary of the data
summary(data_full)
```

Code: Viewing the "data_full" Dataset Information

The first few values of the "data_full" dataset provided is viewed in the console along with the datatype of the columns. You can see the difference between this dataset and "data" dataset is that there would be repeating EMPLOYEE_ID values. It also shows us the column headings of the dataset, with a total of 17 columns, as well as the number of rows in this dataset, which is 49653.



Output: Data View in "data_full" Dataset

The dataset is stored in R as a data frame.

```
> class(data_full)
[1] "data.frame"
```

Output: Class of the "data_full" Dataset

The summary of all data is shown below. Because of the increased data, all columns deviate somewhat from the "data" dataset. However, because most columns include duplicate data, as we can see certain data records are recorded 10 times according to the "EMPLOYEE_ID" column, we will only focus on the relevant columns, namely the "EMPLOYEE_ID", "RECORD_DATE", as well as "RECORD_YEAR" columns in the following sections.

```
> summary(data_full)
  EMPLOYEE_ID     RECORD_DATE                    BIRTH_DATE         ORI_HIRE_DATE        TERMINATION_DATE          AGE
 1318   :   10   Min.   :2006-01-01 00:00:00.0   Min.   :1941-01-15   Min.   :1989-08-28   Min.   :1900-01-01   Min.   :19.00
 1319   :   10   1st Qu.:2008-12-31 00:00:00.0   1st Qu.:1958-05-28   1st Qu.:1995-06-02   1st Qu.:1900-01-01   1st Qu.:31.00
 1320   :   10   Median :2011-12-31 00:00:00.0   Median :1968-12-04   Median :2000-03-31   Median :1900-01-01   Median :42.00
 1321   :   10   Mean   :2011-08-06 19:32:10.4   Mean   :1969-01-09   Mean   :2000-09-04   Mean   :1916-05-10   Mean   :42.08
 1322   :   10   3rd Qu.:2013-12-31 00:00:00.0   3rd Qu.:1979-07-18   3rd Qu.:2005-10-13   3rd Qu.:1900-01-01   3rd Qu.:53.00
 1323   :   10   Max.   :2015-12-31 00:00:00.0   Max.   :1994-12-31   Max.   :2013-12-11   Max.   :2015-12-30   Max.   :65.00
 (Other):49593
 LENGTH_OF_SERVICE             CITY                   DEPARTMENT           JOB_TITLE        STORE         GENDER
 Min.   : 0.00     Vancouver      :11211   Meats           :10269   Meat Cutter  :9984   46     : 4422   Female:25898
 1st Qu.: 5.00     Victoria       : 4885   Dairy           : 8599   Dairy Person :8590   18     : 3876   Male  :23755
 Median :10.00     Nanaimo        : 3876   Produce         : 8515   Produce Clerk:8237   42     : 3827
 Mean   :10.43     New Westminster: 3465   Bakery          : 8381   Baker        :8096   21     : 3211
 3rd Qu.:15.00     Kelowna        : 2513   Customer Service: 7122   Cashier      :6816   43     : 2896
 Max.   :26.00     Burnaby        : 2067   Processed Foods : 5911   Shelf Stocker:5622   16     : 2513
                   (Other)        :21636   (Other)         :  856   (Other)      :2308   (Other):28908
     TERMINATION_REASON      TERMINATION_TYPE  RECORD_YEAR       STATUS          BUSINESS_UNIT
 Layoff        :  215    Involuntary    :  215   Min.   :2006   ACTIVE    :48168   HEADOFFICE:  585
 Not Applicable:48168    Not Applicable:48168   1st Qu.:2008   TERMINATED: 1485   STORES    :49068
 Resignation   :  385    Voluntary      : 1270   Median :2011
 Retirement    :  885                           Mean   :2011
                                                3rd Qu.:2013
                                                Max.   :2015
```

Output: Summary of the "data_full" Dataset

5.2 - “EMPLOYEE_ID” Column

This column contains all employees’ identity code.

```
## Unique data
unique(data$EMPLOYEE_ID)

## Counting the amount of distinct values
nlevels(data$EMPLOYEE_ID)

## Determining the datatype
class(data$EMPLOYEE_ID)
```

Code: Checking Unique data, the Number of Distinct Values and Datatype

The unique data and the amount of distinct values are checked for the “EMPLOYEE_ID” Column. This code would only need to be executed once because the unique information for the "data" and "data_full" datasets would be the same. As we can see, the majority of Employee ID values are continuous, with some values being skipped occasionally. We will assume that the skipped Employee IDs are employees that were terminated prior to the creation of this dataset. We can also observe that there are 6284 unique Employee IDs in this dataset.

```
> unique(data$EMPLOYEE_ID)
   [1] 1318 1319 1320 1321 1322 1323 1325 1328 1329 1330 1331 1332 1334 1335 1338 1339 1340 1341 1343 1344
  [21] 1346 1347 1351 1352 1353 1355 1357 1358 1359 1360 1362 1363 1365 1366 1367 1370 1372 1373 1374 1376
  [41] 1377 1380 1381 1382 1383 1385 1386 1388 1389 1390 1391 1392 1394 1395 1396 1397 1399 1400 1401 1402
  [61] 1403 1404 1405 1406 1408 1409 1411 1412 1413 1414 1417 1419 1421 1422 1425 1427 1428 1430 1431 1433
  [81] 1434 1435 1436 1438 1439 1440 1441 1442 1444 1445 1446 1447 1448 1449 1453 1456 1458 1460 1461 1463
 [101] 1465 1466 1469 1470 1474 1475 1476 1480 1481 1482 1483 1484 1485 1486 1487 1488 1489 1490 1491 1492
 [121] 1494 1495 1497 1498 1499 1500 1502 1512 1513 1516 1517 1522 1523 1524 1527 1528 1529 1530 1532 1533
 [141] 1534 1535 1536 1538 1541 1542 1543 1544 1545 1546 1547 1548 1549 1550 1552 1553 1554 1556 1558 1559
 [161] 1560 1562 1566 1567 1568 1569 1571 1573 1574 1575 1578 1579 1580 1581 1585 1586 1587 1588 1589 1590
 [181] 1592 1593 1594 1595 1596 1597 1598 1599 1600 1603 1610 1611 1612 1613 1615 1616 1617 1618 1619 1621
 [201] 1624 1627 1628 1631 1632 1633 1634 1636 1638 1640 1642 1643 1644 1645 1647 1650 1651 1652 1653 1655
 [221] 1656 1658 1662 1664 1668 1670 1671 1672 1673 1674 1677 1680 1683 1684 1687 1690 1691 1693 1695 1700
 [241] 1703 1705 1706 1710 1711 1713 1714 1715 1716 1717 1718 1719 1720 1721 1722 1726 1728 1730 1732 1733
 [261] 1734 1735 1736 1739 1740 1741 1742 1743 1744 1745 1750 1754 1755 1757 1758 1762 1763 1767 1768 1769
 [281] 1770 1771 1773 1774 1775 1776 1777 1779 1780 1781 1783 1784 1786 1787 1788 1789 1790 1791 1792 1793
 [301] 1794 1795 1799 1800 1801 1802 1803 1804 1806 1808 1810 1811 1812 1814 1816 1817 1820 1823 1825 1827
 [321] 1829 1830 1831 1834 1835 1836 1839 1840 1841 1844 1845 1846 1848 1849 1850 1851 1852 1853 1855 1856
 [341] 1857 1858 1859 1860 1862 1864 1865 1866 1868 1869 1870 1871 1873 1875 1876 1877 1878 1879 1880 1881
 [361] 1882 1884 1885 1888 1892 1894 1895 1896 1898 1899 1900 1901 1902 1905 1906 1907 1908 1909 1910 1911
 [381] 1915 1916 1919 1920 1921 1922 1923 1924 1926 1927 1928 1929 1931 1932 1934 1937 1938 1942 1943 1944
 [401] 1945 1946 1948 1951 1953 1954 1955 1956 1959 1960 1961 1963 1964 1965 1970 1971 1972 1973 1975 1977
 [421] 1978 1979 1980 1985 1987 1988 1990 1992 1993 1994 1995 1996 1998 2002 2003 2004 2005 2008 2009 2011
 [441] 2014 2015 2016 2018 2021 2022 2024 2030 2034 2036 2039 2040 2042 2044 2046 2047 2048 2049 2053 2055
 [461] 2056 2058 2060 2061 2063 2065 2067 2068 2070 2071 2073 2074 2075 2076 2077 2080 2081 2082 2084 2085
 [481] 2086 2088 2089 2090 2091 2093 2094 2095 2096 2100 2101 2103 2105 2108 2110 2111 2112 2113 2114 2115
 [501] 2116 2117 2118 2119 2120 2122 2123 2125 2126 2127 2130 2131 2132 2134 2142 2143 2145 2146 2148 2149
 [521] 2152 2153 2154 2155 2156 2159 2161 2162 2164 2165 2166 2169 2170 2172 2173 2177 2178 2179 2180 2181
 [541] 2183 2187 2188 2189 2190 2192 2194 2195 2201 2202 2203 2204 2205 2206 2207 2210 2211 2213 2214 2215
 [561] 2217 2218 2219 2221 2222 2223 2225 2226 2229 2230 2231 2232 2235 2236 2237 2238 2240 2241 2242 2243
 [581] 2245 2246 2247 2248 2250 2251 2252 2253 2254 2255 2256 2257 2259 2260 2261 2262 2264 2265 2266 2269
 [601] 2271 2272 2273 2274 2275 2276 2277 2280 2282 2283 2286 2290 2293 2294 2295 2296 2297 2298 2299 2302
 [621] 2303 2304 2305 2308 2309 2310 2311 2312 2314 2315 2317 2318 2319 2320 2321 2322 2323 2324 2325 2326
 [641] 2329 2330 2331 2332 2333 2334 2335 2337 2339 2341 2342 2343 2344 2347 2348 2349 2350 2351 2352 2354
 [661] 2355 2356 2357 2358 2359 2360 2362 2363 2364 2365 2366 2367 2368 2370 2371 2372 2373 2374 2376 2377
 [681] 2378 2379 2380 2381 2382 2384 2385 2386 2387 2388 2390 2391 2392 2393 2394 2395 2396 2397 2399 2400
 [701] 2401 2403 2406 2407 2408 2409 2410 2411 2412 2413 2415 2416 2417 2418 2419 2420 2421 2422 2423 2424
 [721] 2425 2427 2428 2430 2431 2432 2433 2434 2436 2438 2440 2441 2442 2443 2444 2446 2447 2449 2450 2451
 [741] 2452 2453 2454 2455 2456 2457 2458 2461 2462 2464 2466 2467 2468 2469 2470 2471 2472 2473 2474 2475
 [761] 2476 2477 2478 2479 2481 2482 2483 2484 2485 2486 2487 2488 2489 2491 2492 2494 2495 2496 2497 2499
 [781] 2500 2501 2504 2505 2506 2507 2508 2509 2510 2511 2514 2515 2516 2517 2518 2520 2522 2523 2524 2525
 [801] 2526 2527 2528 2529 2530 2531 2532 2533 2534 2535 2536 2537 2539 2540 2541 2542 2543 2544 2545 2546
 [821] 2548 2550 2552 2553 2554 2555 2556 2558 2559 2561 2562 2563 2564 2565 2567 2568 2569 2570 2571 2572
 [841] 2574 2575 2576 2577 2578 2579 2580 2581 2582 2583 2584 2585 2586 2588 2589 2590 2591 2592 2593 2595
 [861] 2597 2598 2599 2601 2602 2604 2605 2606 2608 2609 2611 2612 2613 2614 2615 2616 2617 2618 2619 2620
 [881] 2622 2623 2624 2625 2626 2627 2630 2631 2632 2633 2634 2635 2636 2637 2638 2639 2640 2641 2643 2644
 [901] 2645 2646 2647 2648 2650 2651 2652 2653 2654 2657 2660 2663 2664 2665 2666 2667 2668 2669 2670 2671
 [921] 2672 2673 2674 2675 2676 2678 2679 2680 2681 2683 2685 2687 2689 2690 2691 2692 2693 2694 2696 2697
 [941] 2698 2699 2700 2701 2702 2703 2704 2705 2706 2708 2709 2710 2711 2712 2714 2715 2716 2718 2719 2720
 [961] 2721 2722 2723 2724 2726 2727 2728 2730 2731 2732 2733 2734 2735 2736 2737 2738 2739 2740 2741 2742
 [981] 2743 2744 2746 2747 2748 2749 2750 2751 2752 2753 2755 2756 2757 2758 2759 2760 2761 2762 2764 2766
 [ reached getOption("max.print") -- omitted 5284 entries ]
6284 Levels: 1318 1319 1320 1321 1322 1323 1325 1328 1329 1330 1331 1332 1334 1335 1338 1339 1340 ... 8336
```

Output: Unique Data in “EMPLOYEE_ID” Column

```
> nlevels(data$EMPLOYEE_ID)
[1] 6284
```

Output: Amount of distinct values in "EMPLOYEE_ID" Column

```
> class(data$EMPLOYEE_ID)
[1] "factor"
```

Output: Datatype of "EMPLOYEE_ID" Column

As converted earlier, the datatype of the column is Factor.

5.2.1 - In "data_full" dataset

```
## Data summary
summary(data_full$EMPLOYEE_ID)
```

Code: Summary of the "EMPLOYEE_ID" Column

By looking at the summary of the "EMPLOYEE_ID" Column, we can see that most Employee IDs occurred 10 times in the "data_full" dataset, which indicates that there were 10 records of the same Employee IDs in the dataset.

```
> summary(data_full$EMPLOYEE_ID)
   1318    1319    1320    1321    1322    1323    1325    1328    1329    1330    1331    1332    1334
     10      10      10      10      10      10      10      10      10      10      10      10      10
   1335    1513    1516    1522    1524    1527    1530    1533    1534    1538    1541    1542    1543
     10      10      10      10      10      10      10      10      10      10      10      10      10
   1547    1548    1552    1558    1560    1562    1567    1568    1571    1578    1579    1580    1587
     10      10      10      10      10      10      10      10      10      10      10      10      10
   1589    1592    1594    1595    1598    1603    1612    1615    1616    1617    1618    1619    1621
     10      10      10      10      10      10      10      10      10      10      10      10      10
   1627    1628    1631    1632    1633    1634    1638    1643    1650    1651    1655    1656    1664
     10      10      10      10      10      10      10      10      10      10      10      10      10
   1668    1672    1673    1674    1677    1680    1684    1687    1690    1691    1693    1695    1700
     10      10      10      10      10      10      10      10      10      10      10      10      10
   1703    1705    1706    1710    1711    1713    1715    1716    1717    1718    1719    1720    1721
     10      10      10      10      10      10      10      10      10      10      10      10      10
   1722    1726    1728    1730    1732    1733    1734    1735 (Other)
     10      10      10      10      10      10      10      10   48663
```

Output: Summary of the "EMPLOYEE_ID" Column

```
## List of the frequency of each values
exp_emp_id <- tabyl(data_full, EMPLOYEE_ID) %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE)
exp_emp_id
```

Code: Generating a Table of the Frequency of Each "EMPLOYEE_ID"

To this, a table that lists the frequency of each Employee ID is created to see if there were any patterns.

```
> exp_emp_id
 EMPLOYEE_ID  n percent
        1318 10   0.02%
        1319 10   0.02%
        1320 10   0.02%
        1321 10   0.02%
        1322 10   0.02%
        1323 10   0.02%
        1325 10   0.02%
        1328 10   0.02%
        1329 10   0.02%
        1330 10   0.02%
        1331 10   0.02%
        1332 10   0.02%
        1334 10   0.02%
        1335 10   0.02%
        1338  4   0.01%
        1339  9   0.02%
        1340  4   0.01%
        1341  4   0.01%
        1343  9   0.02%
        1344  9   0.02%
        1346  9   0.02%
        1347  4   0.01%
        1351  4   0.01%
        1352  9   0.02%
        1353  4   0.01%
        1355  4   0.01%
        1357  4   0.01%
        1358  4   0.01%
        1359  9   0.02%
        1360  9   0.02%
```

Output: First Few Rows of the Table of the Frequency of Each "EMPLOYEE_ID"

By looking further, we can notice that not all Employee IDs occurred 10 times in the "data_full" dataset. This also indicates that the records for certain Employee IDs were not updated as often.

```
## The summary of the frequency of each values
summary(exp_emp_id$n)
```

Code: Summary of the Frequency of Each "EMPLOYEE_ID"

```
> summary(exp_emp_id$n)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   6.000  10.000   7.901  10.000  10.000
```

Output: Summary of the Frequency of Each "EMPLOYEE_ID"

We may deduce from the results that certain Employee IDs have only been recorded once in this dataset, and that the maximum value of the occurrence of each Employee ID is 10.

```r
## Create a table that displays the frequency of the number of occurrence for each employee ID
exp_emp_id <- tabyl(data_full, EMPLOYEE_ID) %>%
  dplyr::arrange( n ) %>%
  split( .[,"n"] ) %>%
  purrr::map_df(., janitor::adorn_totals) %>%
  filter(EMPLOYEE_ID == "Total")

## Appending the frequency number column to the data since the min & max frequency is known
Times_Repeated = factor(c(1:10))
exp_emp_id = cbind(exp_emp_id, Times_Repeated)

## Deleting unecessary columns
exp_emp_id["EMPLOYEE_ID"] = NULL

## Reorder column
exp1_emp_id <- exp_emp_id[, c(3,1,2)]

## TABLE: Frequency of the number of occurrence for EMPLOYEE_ID
exp1_emp_id %>%
    adorn_totals() %>%
    adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
```

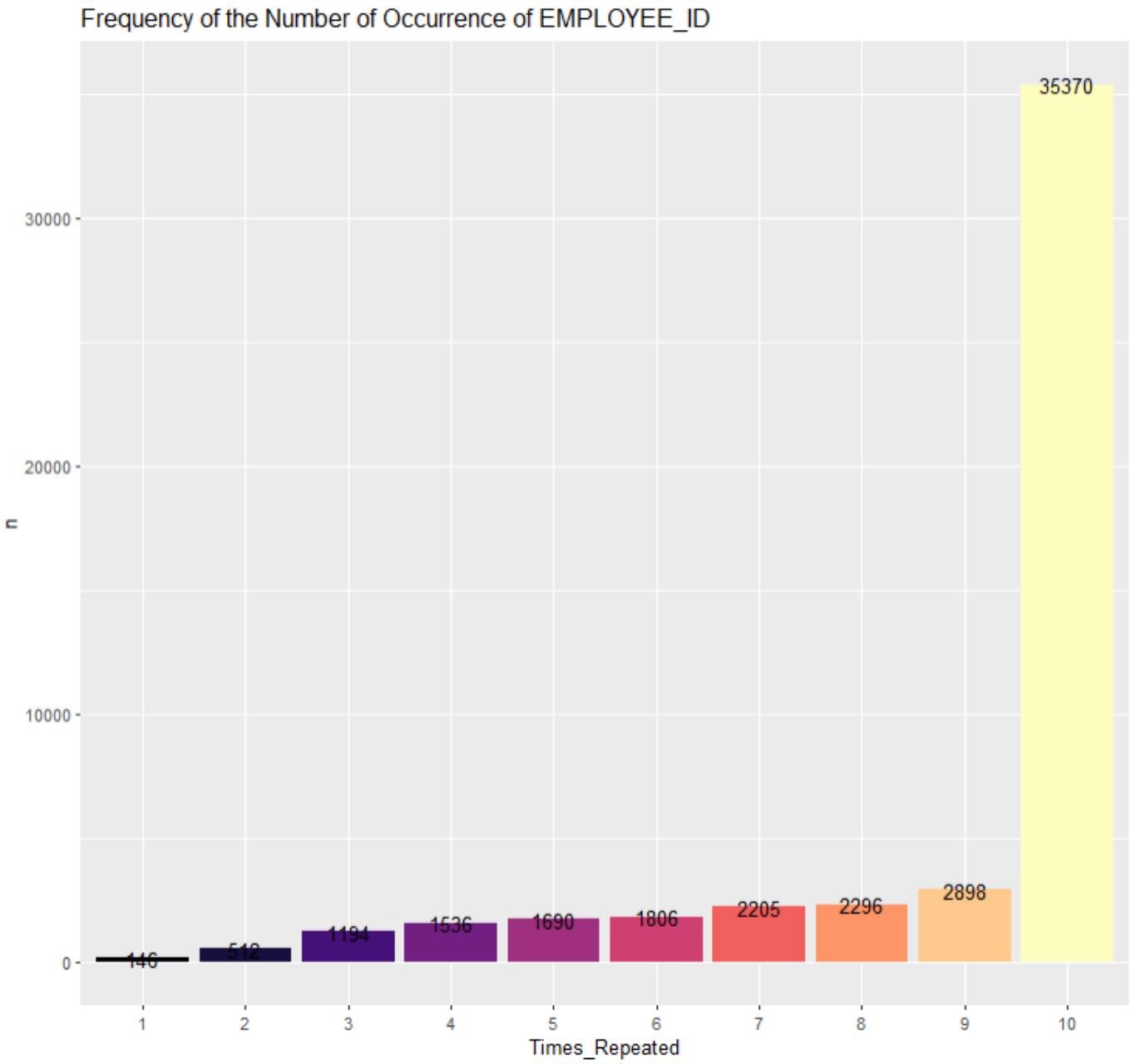
Code: Generating a Table of the Frequency of the Number of Occurrence for EMPLOYEE_ID

```
> exp_emp_id <- tabyl(data_full, EMPLOYEE_ID) %>%
+   dplyr::arrange( n ) %>%
+   split( .[,"n"] ) %>%
+   purrr::map_df(., janitor::adorn_totals) %>%
+   filter(EMPLOYEE_ID == "Total")
> Times_Repeated = factor(c(1:10))
> exp_emp_id = cbind(exp_emp_id, Times_Repeated)
> exp_emp_id["EMPLOYEE_ID"] = NULL
> exp1_emp_id <- exp_emp_id[, c(3,1,2)]
> exp1_emp_id %>%
+     adorn_totals() %>%
+     adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
 Times_Repeated     n percent
              1   146   0.29%
              2   512   1.03%
              3  1194   2.40%
              4  1536   3.09%
              5  1690   3.40%
              6  1806   3.64%
              7  2205   4.44%
              8  2296   4.62%
              9  2898   5.84%
             10 35370  71.23%
          Total 49653 100.00%
```

Output: Table of the Frequency of the Number of Occurrence for EMPLOYEE_ID

```r
ggplot(exp_emp_id, aes(x=Times_Repeated, y=n)) +
  geom_bar(stat="identity", fill=magma(10)) +
  geom_text(aes(label=n)) +
  ggtitle("Frequency of the Number of Occurrence of EMPLOYEE_ID")
```

Code: Generating a Bar Graph of the Frequency of the Number of Occurrence for EMPLOYEE_ID



Graph: Bar Graph of the Frequency of the Number of Occurrence for EMPLOYEE_ID

The output reveal that majority (71.23%) of Employee IDs appeared 10 times in the "data_full" dataset, while just 146 out of all data (0.29%) had only 1 entry in the dataset.

5.2.2 - In "data" dataset

```
summary(data$EMPLOYEE_ID)
```

Code: Summary of the "EMPLOYEE_ID" Column

```
> summary(data$EMPLOYEE_ID)
   1318    1319    1320    1321    1322    1323    1325    1328
      1       1       1       1       1       1       1       1
   1329    1330    1331    1332    1334    1335    1338    1339
      1       1       1       1       1       1       1       1
   1340    1341    1343    1344    1346    1347    1351    1352
      1       1       1       1       1       1       1       1
   1353    1355    1357    1358    1359    1360    1362    1363
      1       1       1       1       1       1       1       1
   1365    1366    1367    1370    1372    1373    1374    1376
      1       1       1       1       1       1       1       1
   1377    1380    1381    1382    1383    1385    1386    1388
      1       1       1       1       1       1       1       1
   1389    1390    1391    1392    1394    1395    1396    1397
      1       1       1       1       1       1       1       1
   1399    1400    1401    1402    1403    1404    1405    1406
      1       1       1       1       1       1       1       1
   1408    1409    1411    1412    1413    1414    1417    1419
      1       1       1       1       1       1       1       1
   1421    1422    1425    1427    1428    1430    1431    1433
      1       1       1       1       1       1       1       1
   1434    1435    1436    1438    1439    1440    1441    1442
      1       1       1       1       1       1       1       1
   1444    1445    1446    1447    1448    1449    1453    1456
      1       1       1       1       1       1       1       1
   1458    1460    1461 (Other)
      1       1       1    6185
```

Output: Summary of the "EMPLOYEE_ID" Column

We can see from the summary of the "EMPLOYEE ID" Column that all Employee IDs happened just once in the "data" dataset since this dataset only consists of the latest records of each Employee ID.

5.3 - "RECORD_DATE" Column

This column indicates the dates on which the employees' records were created.

```
class(data$RECORD_DATE)
```

Code: Datatype of "RECORD_DATE" Column

```
> class(data$RECORD_DATE)
[1] "POSIXlt" "POSIXt"
```

Output: Datatype of "RECORD_DATE" Column

As converted earlier, the datatype of the column is POSIXlt. This code would only need to be executed once because the datatype for the "data" and "data_full" datasets would be the same.

5.3.1 - In "data_full" dataset

```
## Data summary
summary(data_full$RECORD_DATE)

## Unique data
unique(data_full$RECORD_DATE)
```

Code: Summary and Unique Data of the "RECORD_DATE" Column

```
> summary(data_full$RECORD_DATE)
                  Min.                   1st Qu.                    Median                      Mean
"2006-01-01 00:00:00.0000" "2008-12-31 00:00:00.0000" "2011-12-31 00:00:00.0000" "2011-08-06 19:32:10.3969"
               3rd Qu.                      Max.
"2013-12-31 00:00:00.0000" "2015-12-31 00:00:00.0000"
```

Output: Summary of the "RECORD_DATE" Column

According to the summary, the earliest data record is on the 1st of January 2006, while the most recent data record is on the 31st of December 2015.

```
> unique(data_full$RECORD_DATE)
  [1] "2006-12-31 +08" "2007-12-31 +08" "2008-12-31 +08" "2009-12-31 +08" "2010-12-31 +08" "2011-12-31 +08" "2012-12-31 +08"
  [8] "2013-12-31 +08" "2014-12-31 +08" "2015-12-31 +08" "2009-02-01 +08" "2014-02-01 +08" "2014-03-01 +08" "2009-03-01 +08"
 [15] "2012-09-01 +08" "2014-04-01 +08" "2009-04-01 +08" "2009-05-01 +08" "2014-05-01 +08" "2014-12-01 +08" "2009-06-01 +08"
 [22] "2014-06-01 +08" "2014-07-01 +08" "2009-07-01 +08" "2009-08-01 +08" "2014-08-01 +08" "2009-12-01 +08" "2014-09-01 +08"
 [29] "2009-09-01 +08" "2009-10-01 +08" "2014-10-01 +08" "2014-11-01 +08" "2009-11-01 +08" "2007-07-01 +08" "2015-01-01 +08"
 [36] "2010-01-01 +08" "2010-12-01 +08" "2015-02-01 +08" "2010-02-01 +08" "2015-03-01 +08" "2010-03-01 +08" "2010-04-01 +08"
 [43] "2015-04-01 +08" "2015-05-01 +08" "2010-05-01 +08" "2010-06-01 +08" "2015-06-01 +08" "2010-07-01 +08" "2015-07-01 +08"
 [50] "2015-12-01 +08" "2011-03-01 +08" "2015-08-01 +08" "2010-09-01 +08" "2015-09-01 +08" "2015-10-01 +08" "2010-10-01 +08"
 [57] "2011-10-01 +08" "2015-11-01 +08" "2010-11-01 +08" "2007-10-01 +08" "2007-04-01 +08" "2012-01-01 +08" "2006-09-01 +08"
 [64] "2011-09-01 +08" "2006-01-01 +08" "2008-03-01 +08" "2006-02-01 +08" "2006-03-01 +08" "2006-04-01 +08" "2006-05-01 +08"
 [71] "2007-03-01 +08" "2006-06-01 +08" "2006-07-01 +08" "2006-08-01 +08" "2006-10-01 +08" "2006-11-01 +08" "2011-04-01 +08"
 [78] "2006-12-01 +08" "2007-01-01 +08" "2007-02-01 +08" "2009-01-01 +08" "2007-05-01 +08" "2008-11-01 +08" "2007-12-01 +08"
 [85] "2007-06-01 +08" "2011-12-01 +08" "2007-08-01 +08" "2008-06-01 +08" "2007-09-01 +08" "2007-11-01 +08" "2012-12-01 +08"
 [92] "2008-01-01 +08" "2008-02-01 +08" "2008-04-01 +08" "2008-05-01 +08" "2011-07-01 +08" "2008-07-01 +08" "2008-08-01 +08"
 [99] "2008-09-01 +08" "2008-10-01 +08" "2008-12-01 +08" "2010-08-01 +08" "2012-03-01 +08" "2011-01-01 +08" "2011-02-01 +08"
[106] "2011-05-01 +08" "2011-06-01 +08" "2013-10-01 +08" "2011-08-01 +08" "2013-02-01 +08" "2011-11-01 +08" "2012-02-01 +08"
[113] "2012-04-01 +08" "2012-05-01 +08" "2012-06-01 +08" "2012-07-01 +08" "2012-08-01 +08" "2012-10-01 +08" "2012-11-01 +08"
[120] "2013-01-01 +08" "2013-03-01 +08" "2013-04-01 +08" "2013-05-01 +08" "2013-06-01 +08" "2013-07-01 +08" "2013-08-01 +08"
[127] "2013-09-01 +08" "2013-11-01 +08" "2013-12-01 +08" "2014-01-01 +08"
```

Output: Checking Unique Data of the "RECORD_DATE" Column

The "RECORD_DATE" column is checked for unique values. There are 130 unique dates, with all dates being either on the 1st day of any month, or the last day of December from the year 2006 to 2015. All Record Dates are also in the UTC +8 timezone, with each record being taken at precisely 0000 hours.

```
exp_recorddate <- tabyl(data_full, RECORD_DATE) %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE)
exp_recorddate
```

Code: List of the Frequency of Each Date of the "RECORD_DATE" Column

RECORD_DATE	n	percent
2006-01-01	8	0.02%
2006-02-01	12	0.02%
2006-03-01	12	0.02%
2006-04-01	9	0.02%
2006-05-01	11	0.02%
2006-06-01	9	0.02%
2006-07-01	18	0.04%
2006-08-01	5	0.01%
2006-09-01	13	0.03%
2006-10-01	13	0.03%
2006-11-01	16	0.03%
2006-12-01	8	0.02%
2006-12-31	4445	8.95%
2007-01-01	12	0.02%
2007-02-01	15	0.03%
2007-03-01	16	0.03%
2007-04-01	17	0.03%
2007-05-01	8	0.02%
2007-06-01	12	0.02%
2007-07-01	9	0.02%
2007-08-01	11	0.02%

Output: First Few Rows of the List of the Frequency of Each Date of the "RECORD_DATE" Column

From the output, we can see that most data are recorded during the last day of December for each year, while some data are updated in a small amount occasionally.

```
summary(exp_recorddate$n)
```

Code: Summary of the Frequency of Each "RECORD_DATE"

```
> summary(exp_recorddate$n)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    2.0     8.0    11.0   381.9    15.0  5215.0
```

Output: Summary of the Frequency of Each "RECORD_DATE"

The findings show that a record has been uploaded at least twice on the same day for every date, with the greatest value of the recurrence of a date being 5215.

```r
## Extracting just the month value of the dates
exp_recorddate_month = format(as.Date(data_full$RECORD_DATE, format="%Y/%m/%d"),"%m")

## Counting the amount of distinct months -> Records are done in all 12 months
n_distinct(exp_recorddate_month)
```

Code: Extracting Month Value from "RECORD_DATE" and Counting the Distinct Months

```
> ## Extracting just the month value of the dates
> exp_recorddate_month = format(as.Date(data_full$RECORD_DATE, format="%Y/%m/%d"),"%m")
>
> ## Counting the amount of distinct months -> Records are done in all 12 months
> n_distinct(exp_recorddate_month)
[1] 12
```

Output: Counting the Distinct Months of "RECORD_DATE" Column

The results show that there are 12 distinct months in the "RECORD_DATE" Column, meaning a data record has been done in every month throughout all the years.

```r
## Arranging the months and their frequency
exp_recorddate_month <- tabyl(exp_recorddate_month) %>%
  dplyr::arrange( n ) %>%
  dplyr::arrange(exp_recorddate_month)
names(exp_recorddate_month)[1] <- "Month"

## TABLE: Frequency of RECORD_DATE by Month (Including All Records)
exp1_recorddate <- exp_recorddate_month %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
exp1_recorddate
```

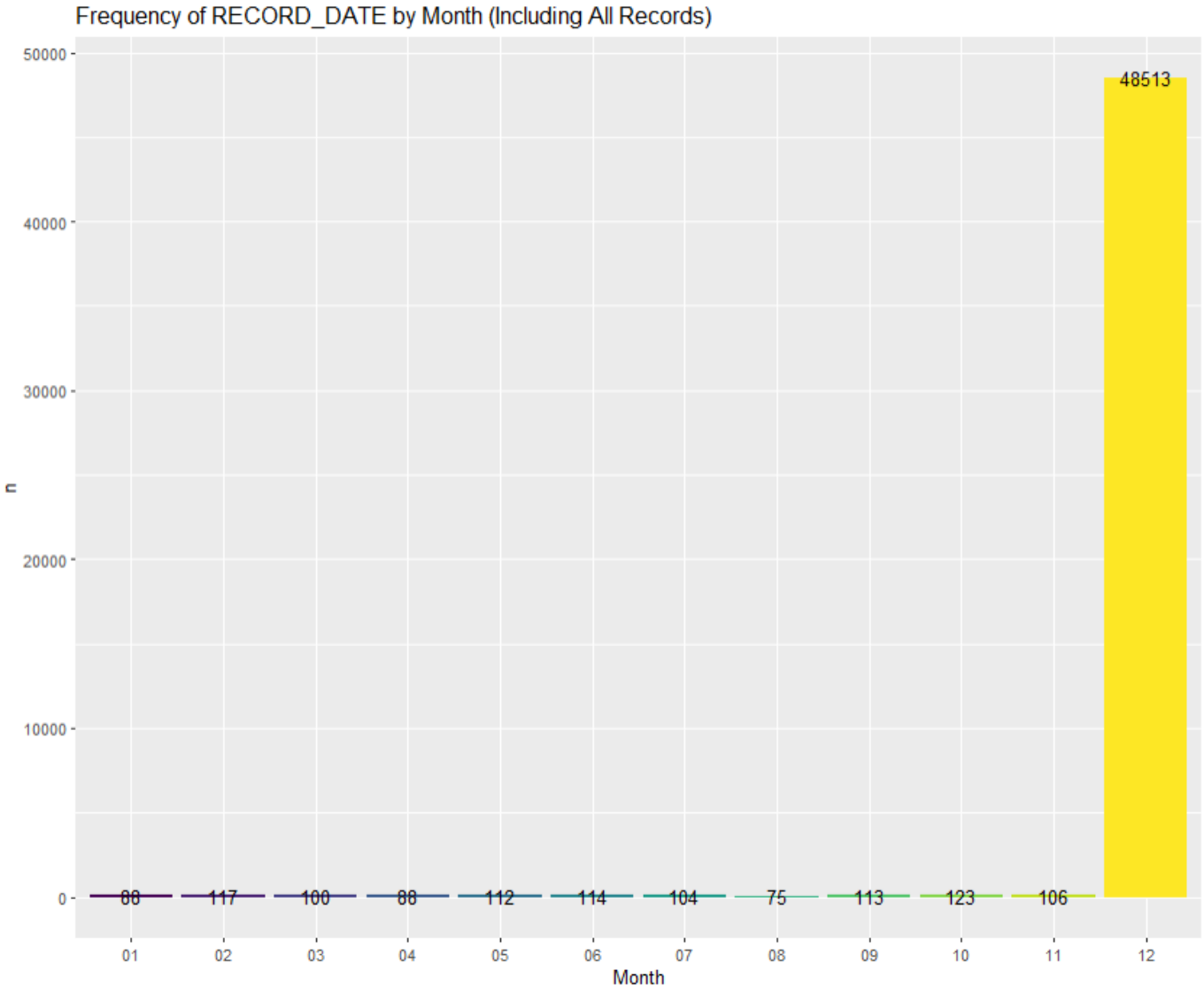
Code: Generating a Table of the Frequency of "RECORD_DATE" by Month

```
> exp_recorddate_month <- tabyl(exp_recorddate_month) %>%
+   dplyr::arrange( n ) %>%
+   dplyr::arrange(exp_recorddate_month)
> names(exp_recorddate_month)[1] <- "Month"
>
> ## TABLE: Frequency of RECORD_DATE by Month (Including All Records)
> exp1_recorddate <- exp_recorddate_month %>%
+   adorn_totals() %>%
+   adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
> exp1_recorddate
 Month     n percent
    01    88   0.18%
    02   117   0.24%
    03   100   0.20%
    04    88   0.18%
    05   112   0.23%
    06   114   0.23%
    07   104   0.21%
    08    75   0.15%
    09   113   0.23%
    10   123   0.25%
    11   106   0.21%
    12 48513  97.70%
 Total 49653 100.00%
```

Output: Table of the Frequency of "RECORD_DATE" by Month

```r
## CHART: Frequency of RECORD_DATE by Month (Including All Records)
exp_recorddate_vis <- exp_recorddate_month %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
ggplot(exp_recorddate_vis, aes(x=Month, y=n)) +
  geom_bar(stat="identity",
           fill=viridis(n_distinct(exp_recorddate_month))) +
  geom_text(aes(label=n)) +
  ggtitle("Frequency of RECORD_DATE by Month (Including All Records)")
```

Code: Generating a Bar Graph of the Frequency of "RECORD_DATE" by Month



Graph: Bar Graph of the Frequency of "RECORD_DATE" by Month

In the data, we can see that most records were entered during December, with a total of 48513 records (97.70%). I believe the reason so many entries were made in December was to update the number of employees that were not terminated yet during the year. For other months, a consistent number of records are done, with 11 months having less than 124 data recorded by month in total.

5.3.2 - In “data” dataset

```
## Data summary
summary(data$RECORD_DATE)

## Unique data
unique(data$RECORD_DATE)
```

Code: Summary and Unique Data of the “RECORD_DATE” Column

```
> summary(data$RECORD_DATE)
                     Min.                   1st Qu.
"2006-01-01 00:00:00.0000" "2015-12-31 00:00:00.0000"
                   Median                      Mean
"2015-12-31 00:00:00.0000" "2014-11-13 20:36:58.2049"
                  3rd Qu.                      Max.
"2015-12-31 00:00:00.0000" "2015-12-31 00:00:00.0000"
```

Output: Summary of the “RECORD_DATE” Column

According to the summary, the earliest data record is on the 1st of January 2006, while the most recent data record is on the 31st of December 2015. This is the same with the “data_full” dataset, with only slight changes in the first quartile, median, mean and 3rd quartile since the latest records are done in the latest year which is 2015.

```
> unique(data$RECORD_DATE)
  [1] "2015-12-31 +08" "2009-02-01 +08" "2014-02-01 +08" "2014-03-01 +08" "2009-03-01 +08" "2012-09-01 +08"
  [7] "2014-04-01 +08" "2009-04-01 +08" "2009-05-01 +08" "2014-05-01 +08" "2014-12-01 +08" "2009-06-01 +08"
 [13] "2014-06-01 +08" "2014-07-01 +08" "2009-07-01 +08" "2009-08-01 +08" "2014-08-01 +08" "2009-12-01 +08"
 [19] "2014-09-01 +08" "2009-09-01 +08" "2009-10-01 +08" "2014-10-01 +08" "2014-11-01 +08" "2009-11-01 +08"
 [25] "2007-07-01 +08" "2015-01-01 +08" "2010-01-01 +08" "2010-12-01 +08" "2015-02-01 +08" "2010-02-01 +08"
 [31] "2015-03-01 +08" "2010-03-01 +08" "2010-04-01 +08" "2015-04-01 +08" "2015-05-01 +08" "2010-05-01 +08"
 [37] "2010-06-01 +08" "2015-06-01 +08" "2010-07-01 +08" "2015-07-01 +08" "2015-12-01 +08" "2011-03-01 +08"
 [43] "2015-08-01 +08" "2010-09-01 +08" "2015-09-01 +08" "2015-10-01 +08" "2010-10-01 +08" "2011-10-01 +08"
 [49] "2015-11-01 +08" "2010-11-01 +08" "2007-10-01 +08" "2007-04-01 +08" "2012-01-01 +08" "2006-09-01 +08"
 [55] "2011-09-01 +08" "2006-01-01 +08" "2008-03-01 +08" "2006-02-01 +08" "2006-03-01 +08" "2006-04-01 +08"
 [61] "2006-05-01 +08" "2007-03-01 +08" "2006-06-01 +08" "2006-07-01 +08" "2006-08-01 +08" "2006-10-01 +08"
 [67] "2006-11-01 +08" "2011-04-01 +08" "2006-12-01 +08" "2007-01-01 +08" "2007-02-01 +08" "2009-01-01 +08"
 [73] "2007-05-01 +08" "2008-11-01 +08" "2007-12-01 +08" "2007-06-01 +08" "2011-12-01 +08" "2007-08-01 +08"
 [79] "2008-06-01 +08" "2007-09-01 +08" "2007-11-01 +08" "2012-12-01 +08" "2007-12-31 +08" "2008-01-01 +08"
 [85] "2008-02-01 +08" "2008-04-01 +08" "2008-05-01 +08" "2011-07-01 +08" "2008-07-01 +08" "2008-08-01 +08"
 [91] "2008-09-01 +08" "2008-10-01 +08" "2008-12-01 +08" "2008-12-31 +08" "2010-08-01 +08" "2012-03-01 +08"
 [97] "2011-01-01 +08" "2011-02-01 +08" "2011-05-01 +08" "2011-06-01 +08" "2013-10-01 +08" "2011-08-01 +08"
[103] "2013-02-01 +08" "2011-11-01 +08" "2012-02-01 +08" "2012-04-01 +08" "2012-05-01 +08" "2012-06-01 +08"
[109] "2012-07-01 +08" "2012-08-01 +08" "2012-10-01 +08" "2012-11-01 +08" "2013-01-01 +08" "2013-03-01 +08"
[115] "2013-04-01 +08" "2013-05-01 +08" "2013-06-01 +08" "2013-07-01 +08" "2013-08-01 +08" "2013-09-01 +08"
[121] "2013-11-01 +08" "2013-12-01 +08" "2009-12-31 +08" "2014-12-31 +08" "2014-01-01 +08" "2013-12-31 +08"
```

Output: Checking Unique Data of the “RECORD_DATE” Column

The unique data are checked for the “RECORD_DATE” Column which is quite similar to the “data_full” dataset. There are 126 unique dates, with all dates being either on the 1st day of any month, or the last day of December from the year 2006 to 2015. All Record Dates are also in the UTC +8 timezone, with each record being taken at precisely 0000 hours.

```
exp_recorddate <- tabyl(data, RECORD_DATE) %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE)
exp_recorddate
```

Code: List of the Frequency of Each Date of the "RECORD_DATE" Column

RECORD_DATE	n	percent
2006-01-01	8	0.13%
2006-02-01	12	0.19%
2006-03-01	12	0.19%
2006-04-01	9	0.14%
2006-05-01	11	0.18%
2006-06-01	9	0.14%
2006-07-01	18	0.29%
2006-08-01	5	0.08%
2006-09-01	13	0.21%
2006-10-01	13	0.21%
2006-11-01	16	0.25%
2006-12-01	8	0.13%
2007-01-01	12	0.19%
2007-02-01	15	0.24%
2007-03-01	16	0.25%
2007-04-01	17	0.27%
2007-05-01	8	0.13%
2007-06-01	12	0.19%
2007-07-01	9	0.14%
2007-08-01	11	0.18%
2007-09-01	13	0.21%
2007-10-01	15	0.24%
2007-11-01	14	0.22%

Output: First Few Rows of the List of the Frequency of Each Date of the "RECORD_DATE" Column

2015-03-01	4	0.06%
2015-04-01	3	0.05%
2015-05-01	10	0.16%
2015-06-01	9	0.14%
2015-07-01	5	0.08%
2015-08-01	6	0.10%
2015-09-01	6	0.10%
2015-10-01	10	0.16%
2015-11-01	4	0.06%
2015-12-01	85	1.35%
2015-12-31	4799	76.37%

Output: Last Few Rows of the List of the Frequency of Each Date of the "RECORD_DATE" Column

From the output, we can see that most data are recorded during 31st of December 2015, which is when the company updates their latest records every year.

```
summary(exp_recorddate$n)
```

Code: Summary of the Frequency of Each "RECORD_DATE"

```
> summary(exp_recorddate$n)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.00    7.00   10.50   49.87   14.00 4799.00
```

Output: Summary of the Frequency of Each "RECORD_DATE"

We may deduce from the results that the maximum value of the occurrence of a date is 4799.

```
## Extracting just the month value of the dates
exp_recorddate_month = format(as.Date(data$RECORD_DATE, format="%Y/%m/%d"),"%m")

## Counting the amount of distinct months -> Records are done in all 12 months
n_distinct(exp_recorddate_month)
```

Code: Extracting Month Value from "RECORD_DATE" and Counting the Distinct Months

```
> ## Extracting just the month value of the dates
> exp_recorddate_month = format(as.Date(data$RECORD_DATE, format="%Y/%m/%d"),"%m")
>
> ## Counting the amount of distinct months -> Records are done in all 12 months
> n_distinct(exp_recorddate_month)
[1] 12
```

Output: Counting the Distinct Months of "RECORD_DATE" Column

Similarly to the "data_full" dataset, the results show that there are 12 distinct months in the "RECORD_DATE" Column, meaning a data record has been done in every month throughout all the years.

```
## Arranging the months and their frequency
exp_recorddate_month <- tabyl(exp_recorddate_month) %>%
  dplyr::arrange( n ) %>%
  dplyr::arrange(exp_recorddate_month)
names(exp_recorddate_month)[1] <- "Month"

## TABLE: Frequency of RECORD_DATE by Month (Including Only Latest Records)
exp2_recorddate <- exp_recorddate_month %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
exp2_recorddate
```

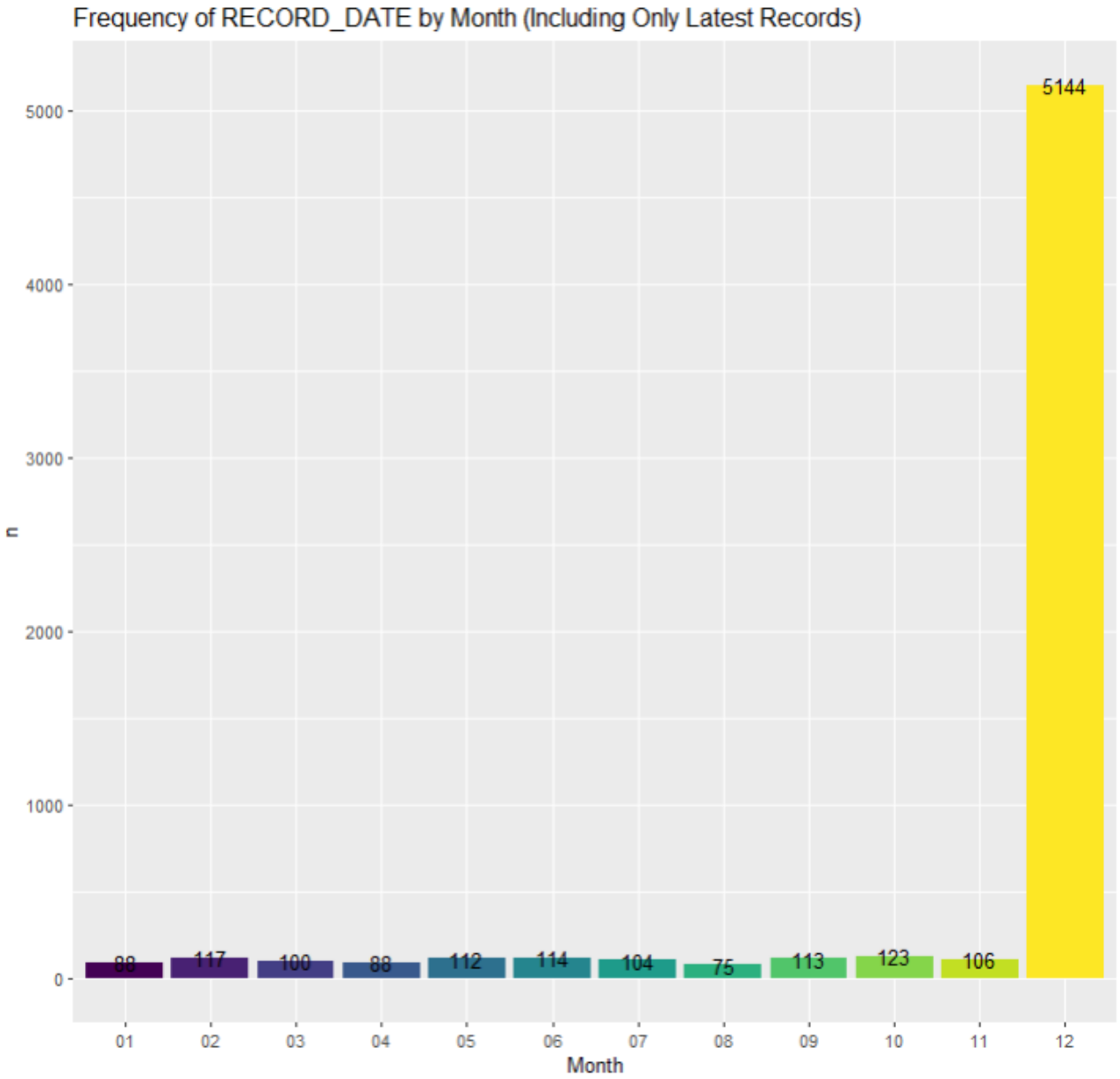
Code: Generating a Table of the Frequency of "RECORD_DATE" by Month

```
> exp_recorddate_month <- tabyl(exp_recorddate_month) %>%
+   dplyr::arrange( n ) %>%
+   dplyr::arrange(exp_recorddate_month)
> names(exp_recorddate_month)[1] <- "Month"
>
> ## TABLE: Frequency of RECORD_DATE by Month (Including Only Latest Records)
> exp2_recorddate <- exp_recorddate_month %>%
+   adorn_totals() %>%
+   adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
> exp2_recorddate
 Month    n percent
    01   88   1.40%
    02  117   1.86%
    03  100   1.59%
    04   88   1.40%
    05  112   1.78%
    06  114   1.81%
    07  104   1.65%
    08   75   1.19%
    09  113   1.80%
    10  123   1.96%
    11  106   1.69%
    12 5144  81.86%
 Total 6284 100.00%
```

Output: Table of the Frequency of "RECORD_DATE" by Month

```r
## CHART: Frequency of RECORD_DATE by Month (Including Only Latest Records)
exp_recorddate_vis2 <- exp_recorddate_month %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
ggplot(exp_recorddate_vis2, aes(x=Month, y=n)) +
  geom_bar(stat="identity",
           fill=viridis(n_distinct(exp_recorddate_month))) +
  geom_text(aes(label=n)) +
  ggtitle("Frequency of RECORD_DATE by Month (Including Only Latest Records)")
```

Code: Generating a Bar Graph of the Frequency of "RECORD_DATE" by Month



Graph: Bar Graph of the Frequency of "RECORD_DATE" by Month

According to the statistics, the majority of entries were entered in December, with a total of 5144 records (97.70%). We can observe that the frequency of Record Dates by Month is the same from January to November by comparing this chart to the same graph in the "data_full" dataset. This demonstrates that the

December data update is basically merely to update the number of employees who have not yet been terminated throughout the year.

5.4 - “BIRTH_DATE” COLUMN

This column indicates the birth dates of the employees.

```
## Data summary
summary(data$BIRTH_DATE)

## Unique data
unique(data$BIRTH_DATE)

## Counting the amount of distinct values
n_distinct(data$BIRTH_DATE)
```

Code: Summary, Unique Data, and Distinct Values of the “BIRTH_DATE” Column

```
> summary(data$BIRTH_DATE)
        Min.      1st Qu.       Median         Mean      3rd Qu.         Max.
"1941-01-15" "1956-12-27" "1970-01-13" "1969-09-17" "1982-12-06" "1994-12-31"
```

Output: Summary of the “BIRTH_DATE” Column

According to the summary, the earliest birth date is on the 15th of January 1941, while the latest birth date is on the 31st of December 1994.

```
> unique(data$BIRTH_DATE)
   [1] "1954-01-03" "1957-01-03" "1955-01-02" "1959-01-02" "1958-01-09" "1962-01-09"
   [7] "1964-01-13" "1956-01-17" "1967-01-23" "1967-01-25" "1965-01-28" "1955-02-05"
  [13] "1961-02-06" "1962-02-07" "1949-02-16" "1949-02-19" "1949-02-21" "1949-02-24"
  [19] "1949-02-25" "1949-03-04" "1949-03-09" "1949-03-11" "1949-03-13" "1949-03-15"
  [25] "1949-03-14" "1949-03-16" "1949-03-17" "1949-03-18" "1949-03-20" "1949-03-24"
  [31] "1949-03-26" "1949-03-27" "1949-04-01" "1949-04-05" "1949-04-04" "1949-04-06"
  [37] "1949-04-11" "1949-04-15" "1949-04-17" "1949-04-20" "1949-04-23" "1949-04-30"
  [43] "1949-05-06" "1949-05-07" "1949-05-08" "1949-05-10" "1949-05-11" "1949-05-17"
  [49] "1949-05-16" "1949-05-18" "1949-05-20" "1949-05-28" "1949-05-27" "1949-05-31"
  [55] "1949-06-02" "1949-06-12" "1949-06-13" "1949-06-16" "1949-06-18" "1949-06-30"
  [61] "1949-07-05" "1949-07-12" "1949-07-18" "1949-07-20" "1949-07-30" "1949-08-01"
  [67] "1949-08-04" "1949-08-10" "1949-08-14" "1949-08-15" "1949-08-18" "1949-08-23"
  [73] "1949-08-27" "1949-08-26" "1949-08-29" "1949-09-03" "1949-09-04" "1949-09-05"
  [79] "1949-09-06" "1949-09-14" "1949-10-01" "1949-10-04" "1949-10-11" "1949-10-12"
  [85] "1949-10-15" "1949-10-16" "1949-10-18" "1949-10-20" "1949-10-23" "1949-11-02"
```

```
> n_distinct(data$BIRTH_DATE)
[1] 5342
```

Output: Top Few Rows of Checking Unique Data and Number of Distinct Values of the “BIRTH_DATE” Column

The "BIRTH_DATE" column is checked for unique values. There are 5342 unique dates in total.

```
class(data$BIRTH_DATE)
```

Code: Datatype of “BIRTH_DATE” Column

```
> class(data$BIRTH_DATE)
[1] "Date"
```

Output: Datatype of "BIRTH_DATE" Column

As converted earlier, the datatype of the column is Date.

```
exp_birthdate <- tabyl(data, BIRTH_DATE) %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE)
exp_birthdate
```

Code: List of the Frequency of Each Date of the "BIRTH_DATE" Column

BIRTH_DATE	n	percent
1941-01-15	2	0.03%
1941-02-14	1	0.02%
1941-02-22	1	0.02%
1941-02-28	1	0.02%
1941-03-07	1	0.02%
1941-03-13	1	0.02%
1941-03-26	1	0.02%
1941-03-30	1	0.02%
1941-04-19	1	0.02%
1941-04-20	1	0.02%
1941-04-22	1	0.02%
1941-04-23	1	0.02%
1941-04-24	1	0.02%
1941-04-28	1	0.02%
1941-05-03	1	0.02%
1941-05-06	2	0.03%
1941-05-12	1	0.02%
1941-05-13	1	0.02%
1941-05-17	1	0.02%
1941-05-31	1	0.02%
1941-06-17	2	0.03%
1941-06-21	1	0.02%
1941-06-28	1	0.02%
1941-06-29	1	0.02%
1941-07-01	1	0.02%
1941-07-06	1	0.02%
1941-07-11	2	0.03%

Output: First Few Rows of the List of the Frequency of Each Date of the "BIRTH_DATE" Column

From the output, we can see that most birth dates are rarely repeated.

```
summary(exp_birthdate$n)
```

Code: Summary of the Frequency of Each "BIRTH_DATE"

```
> summary(exp_birthdate$n)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   1.000   1.000   1.176   1.000   4.000
```

Output: Summary of the Frequency of Each "BIRTH_DATE"

The findings show that the most people in the dataset do not have clashing birth dates, as the 3rd quartile of the data still shows 1. The most frequent birthday date was only repeated 4 times in the data.

```
## Extracting just the year value of the dates
exp_birthdate_year = format(as.Date(data$BIRTH_DATE, format="%Y/%m/%d"),"%Y")

## Counting the amount of distinct years
n_distinct(exp_birthdate_year)
```

Code: Extracting Year Value from "BIRTH_DATE" and Counting the Distinct Years

```
> ## Extracting just the year value of the dates
> exp_birthdate_year = format(as.Date(data$BIRTH_DATE, format="%Y/%m/%d"),"%Y")
>
> ## Counting the amount of distinct years
> n_distinct(exp_birthdate_year)
[1] 54
```

Output: Counting the Distinct Years of "BIRTH_DATE" Column

The results show that there are 54 distinct years in the "BIRTH_DATE" Column, meaning there would be 54 different ages as well.

```
## Arranging the years and their frequency
exp_birthdate_year <- tabyl(exp_birthdate_year) %>%
  dplyr::arrange( n ) %>%
  dplyr::arrange(exp_birthdate_year)
names(exp_birthdate_year)[1] <- "Year"

## TABLE: Frequency of BIRTH_DATE by Year
exp1_birthdate <- exp_birthdate_year %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
exp1_birthdate
```

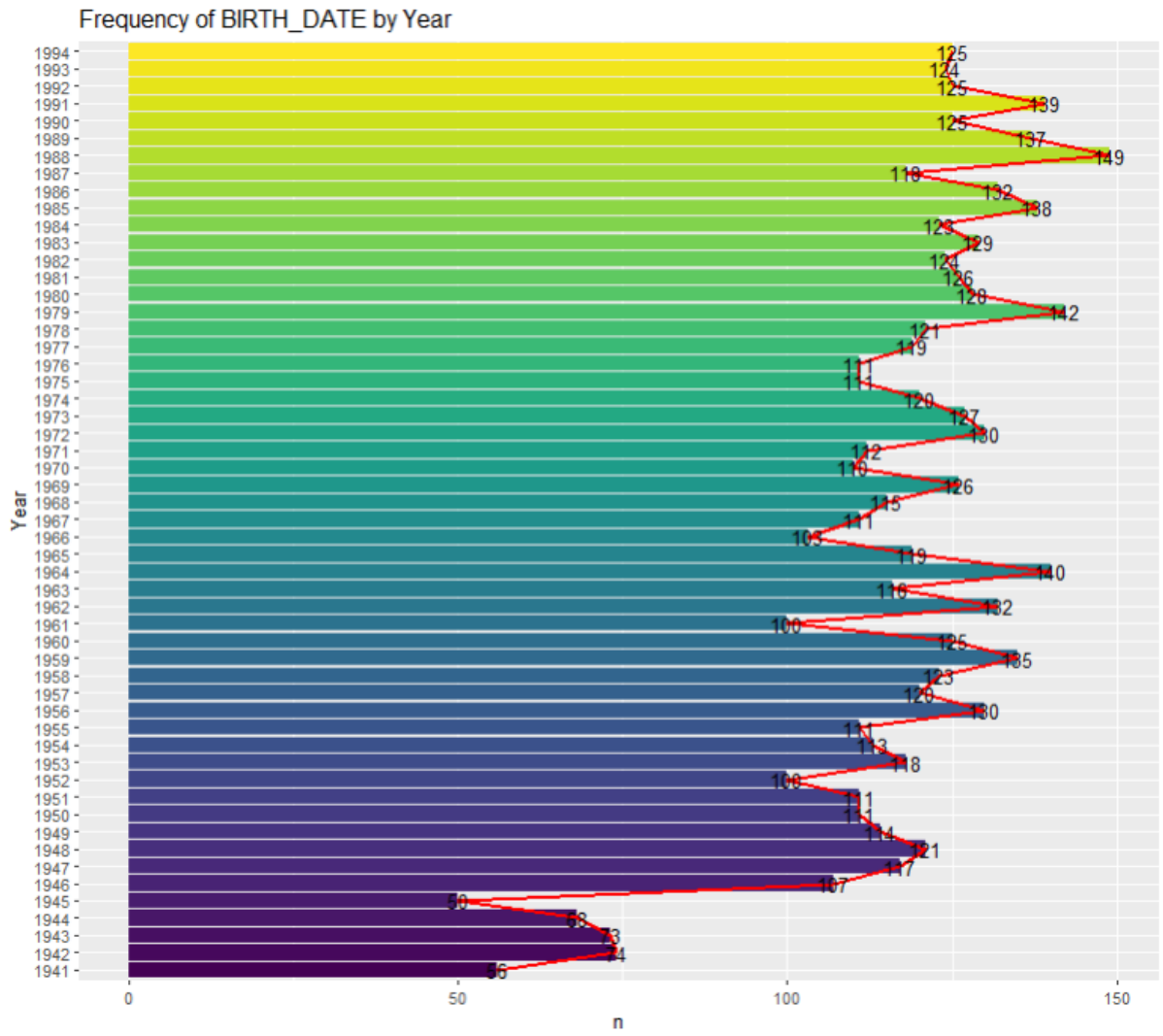
Code: Generating a Table of the Frequency of "BIRTH_DATE" by Year

```
> exp1_birthdate
  Year    n percent
  1941   56   0.89%
  1942   74   1.18%
  1943   73   1.16%
  1944   68   1.08%
  1945   50   0.80%
  1946  107   1.70%
  1947  117   1.86%
  1948  121   1.93%
  1949  114   1.81%
  1950  111   1.77%
  1951  111   1.77%
  1952  100   1.59%
  1953  118   1.88%
  1954  113   1.80%
  1955  111   1.77%
  1956  130   2.07%
  1957  120   1.91%
  1958  123   1.96%
  1959  135   2.15%
  1960  125   1.99%
  1961  100   1.59%
  1962  132   2.10%
  1963  116   1.85%
  1964  140   2.23%
  1965  119   1.89%
  1966  103   1.64%
  1967  111   1.77%
  1968  115   1.83%
  1969  126   2.01%
  1970  110   1.75%
  1971  112   1.78%
  1972  130   2.07%
  1973  127   2.02%
  1974  120   1.91%
  1975  111   1.77%
  1976  111   1.77%
  1977  119   1.89%
  1978  121   1.93%
  1979  142   2.26%
  1980  128   2.04%
  1981  126   2.01%
  1982  124   1.97%
  1983  129   2.05%
  1984  123   1.96%
  1985  138   2.20%
  1986  132   2.10%
  1987  118   1.88%
  1988  149   2.37%
  1989  137   2.18%
  1990  125   1.99%
  1991  139   2.21%
  1992  125   1.99%
  1993  124   1.97%
  1994  125   1.99%
 Total 6284 100.00%
```

Output: Table of the Frequency of "BIRTH_DATE" by Year

```
exp_birthdate_vis <- exp_birthdate_year
ggplot(exp_birthdate_vis, aes(x=n, y=Year, group=1)) +
  geom_bar(stat="identity",
           fill=viridis(n_distinct(exp_birthdate_year))) +
  geom_path(stat="identity", color="red", size=1) +
  geom_text(aes(label=n)) +
  ggtitle("Frequency of BIRTH_DATE by Year")
```

Code: Generating a Bar Graph and Path Graph of the Frequency of "BIRTH_DATE" by Year



Graph: Bar Graph and Path Graph of the Frequency of "BIRTH_DATE" by Year

In the data, we can see that most employees were born on 1988 (2.37%). There is also a sudden drop of employees born in 1946 (107 employees, 1.70%) and 1945 (50 employees, 0.80%).

By looking from a generational point of view, in 2015, the employees that are 51 years old and above are called Boomers (Born in 1946 to 1964), employees that are 39 years old to 50 years old are called Gen X (Born in 1965 to 1976), while the employees that are 20 years old to 38 years old are called millennials or Gen Y (Born in 1977 to 1995). Employers born in the year 1945 and below are called traditionalists (Lablogatory, 2018), while employees that are currently 19 years old are the first pioneers of Gen Z (Born in 1996 to 2015). (Dorsey, 2021) This information would be beneficial in analysing data later as we can get to know certain behaviours of certain generations.

5.5 - "ORI_HIRE_DATE" COLUMN

This column indicates the dates on which the employees were hired.

```
## Data summary
summary(data$ORI_HIRE_DATE)

## Unique data
unique(data$ORI_HIRE_DATE)

## Counting the amount of distinct values
n_distinct(data$ORI_HIRE_DATE)
```

Code: Summary, Unique Data, and Distinct Values of the "ORI_HIRE_DATE" Column

```
> summary(data$ORI_HIRE_DATE)
        Min.      1st Qu.       Median         Mean      3rd Qu.         Max.
"1989-08-28" "1995-09-30" "2000-10-05" "2001-07-23" "2007-07-28" "2013-12-11"
```

Output: Summary of the "ORI_HIRE_DATE" Column

According to the summary, the earliest hiring date is on the 28th of August 1989, while the latest hiring date is on the 11st of December 2013.

```
> unique(data$ORI_HIRE_DATE)
  [1] "1989-08-28" "1989-08-31" "1989-09-02" "1989-09-05" "1989-09-08" "1989-09-09"
  [7] "1989-09-10" "1989-09-15" "1989-09-16" "1989-09-20" "1989-09-22" "1989-09-23"
 [13] "1989-09-25" "1989-09-29" "1989-10-02" "1989-10-03" "1989-10-04" "1989-10-05"
 [19] "1989-10-06" "1989-10-07" "1989-10-08" "1989-10-10" "1989-10-11" "1989-10-14"
 [25] "1989-10-16" "1989-10-17" "1989-10-19" "1989-10-22" "1989-10-23" "1989-10-24"
 [31] "1989-10-26" "1989-10-29" "1989-11-02" "1989-11-03" "1989-11-04" "1989-11-07"
 [37] "1989-11-08" "1989-11-09" "1989-11-13" "1989-11-15" "1989-11-16" "1989-11-21"
 [43] "1989-11-22" "1989-11-23" "1989-11-24" "1989-12-01" "1989-12-03" "1989-12-07"
 [49] "1989-12-10" "1989-12-11" "1989-12-16" "1989-12-18" "1989-12-19" "1989-12-22"
 [55] "1989-12-24" "1989-12-25" "1989-12-27" "1989-12-29" "1989-12-31" "1990-01-01"
 [61] "1990-01-04" "1990-01-05" "1990-01-06" "1990-01-10" "1990-01-19" "1990-01-20"
 [67] "1990-01-24" "1990-01-25" "1990-01-26" "1990-01-27" "1990-01-28" "1990-01-29"
 [73] "1990-01-30" "1990-02-05" "1990-02-06" "1990-02-07" "1990-02-10" "1990-02-12"
 [79] "1990-02-15" "1990-02-16" "1990-02-17" "1990-02-18" "1990-02-19" "1990-02-20"
 [85] "1990-02-21" "1990-02-22" "1990-02-26" "1990-02-27" "1990-03-01" "1990-03-02"
```

```
> n_distinct(data$ORI_HIRE_DATE)
[1] 4415
```

Output: Top Few Rows of Checking Unique Data and Number of Distinct Values of the "ORI_HIRE_DATE" Column

The "ORI_HIRE_DATE" column is checked for unique values. There are 4415 unique dates in total.

```
class(data$ORI_HIRE_DATE)
```

Code: Datatype of "ORI_HIRE_DATE" Column

```
> class(data$ORI_HIRE_DATE)
[1] "Date"
```

Output: Datatype of "ORI_HIRE_DATE" Column

As converted earlier, the datatype of the column is Date.

```
exp_orihiredate <- tabyl(data, ORI_HIRE_DATE) %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE)
exp_orihiredate
```

Code: List of the Frequency of Each Date of the "ORI_HIRE_DATE" Column

```
> exp_orihiredate
 ORI_HIRE_DATE n percent
    1989-08-28 4   0.06%
    1989-08-31 2   0.03%
    1989-09-02 1   0.02%
    1989-09-05 1   0.02%
    1989-09-08 1   0.02%
    1989-09-09 1   0.02%
    1989-09-10 1   0.02%
    1989-09-15 2   0.03%
    1989-09-16 1   0.02%
    1989-09-20 1   0.02%
    1989-09-22 2   0.03%
    1989-09-23 1   0.02%
    1989-09-25 2   0.03%
    1989-09-29 2   0.03%
    1989-10-02 1   0.02%
    1989-10-03 1   0.02%
    1989-10-04 1   0.02%
    1989-10-05 3   0.05%
    1989-10-06 2   0.03%
    1989-10-07 1   0.02%
    1989-10-08 1   0.02%
    1989-10-10 1   0.02%
    1989-10-11 2   0.03%
    1989-10-14 1   0.02%
    1989-10-16 2   0.03%
    1989-10-17 1   0.02%
```

Output: First Few Rows of the List of the Frequency of Each Date of the "ORI_HIRE_DATE" Column

From the output, we can see that most hiring dates are rarely repeated.

```
summary(exp_orihiredate$n)
```

Code: Summary of the Frequency of Each "ORI_HIRE_DATE"

```
> summary(exp_orihiredate$n)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   1.000   1.000   1.423   2.000   5.000
```

Output: Summary of the Frequency of Each "ORI_HIRE_DATE"

The findings show that the most people in the dataset do not have clashing hiring dates, as the median of the data still shows 1. Only a maximum of 5 employees were hired on the same day.

```
## Extracting just the year value of the dates
exp_orihiredate_year = format(as.Date(data$ORI_HIRE_DATE, format="%Y/%m/%d"),"%Y")

## Counting the amount of distinct years
n_distinct(exp_orihiredate_year)
```

Code: Extracting Year Value from "ORI_HIRE_DATE" and Counting the Distinct Years

```
> ## Extracting just the year value of the dates
> exp_orihiredate_year = format(as.Date(data$ORI_HIRE_DATE, format="%Y/%m/%d"),"%Y")
>
> ## Counting the amount of distinct years
> n_distinct(exp_orihiredate_year)
[1] 25
```

Output: Counting the Distinct Years of "ORI_HIRE_DATE" Column

The results show that there are 25 distinct years in the "ORI_HIRE_DATE" Column.

```
## Arranging the years and their frequency
exp_orihiredate_year <- tabyl(exp_orihiredate_year) %>%
  dplyr::arrange( n ) %>%
  dplyr::arrange(exp_orihiredate_year)
names(exp_orihiredate_year)[1] <- "Year"

## TABLE: Frequency of ORI_HIRE_DATE by Year
exp1_orihiredate <- exp_orihiredate_year %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
exp1_orihiredate
```

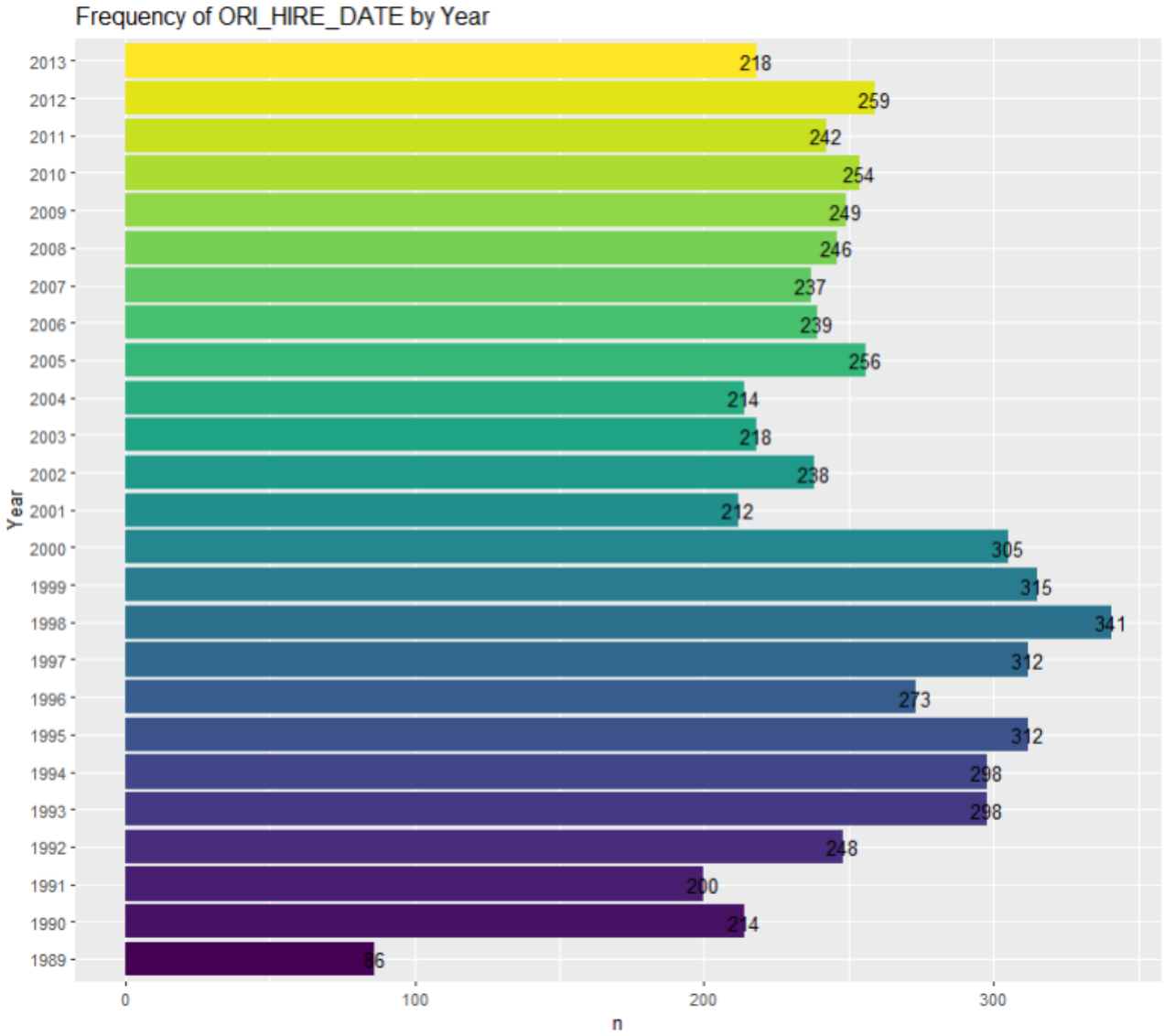
Code: Generating a Table of the Frequency of "ORI_HIRE_DATE" by Year

```
> exp1_orihiredate
   Year    n percent
   1989   86   1.37%
   1990  214   3.41%
   1991  200   3.18%
   1992  248   3.95%
   1993  298   4.74%
   1994  298   4.74%
   1995  312   4.96%
   1996  273   4.34%
   1997  312   4.96%
   1998  341   5.43%
   1999  315   5.01%
   2000  305   4.85%
   2001  212   3.37%
   2002  238   3.79%
   2003  218   3.47%
   2004  214   3.41%
   2005  256   4.07%
   2006  239   3.80%
   2007  237   3.77%
   2008  246   3.91%
   2009  249   3.96%
   2010  254   4.04%
   2011  242   3.85%
   2012  259   4.12%
   2013  218   3.47%
  Total 6284 100.00%
```

Output: Table of the Frequency of "ORI_HIRE_DATE" by Year

```r
## CHART: Frequency of ORI_HIRE_DATE by Year
exp_orihiredate_vis <- exp_orihiredate_year
ggplot(exp_orihiredate_vis, aes(x=n, y=Year)) +
  geom_bar(stat="identity",
           fill=viridis(n_distinct(exp_orihiredate_year))) +
  geom_text(aes(label=n)) +
  ggtitle("Frequency of ORI_HIRE_DATE by Year")
```

Code: Generating a Bar Graph of the Frequency of "ORI_HIRE_DATE" by Year



Graph: Bar Graph and Path Graph of the Frequency of "ORI_HIRE_DATE" by Year

In the data, we can see that most employees were hired on 1998 (341 employees, 5.43%). There were less people hired in 1989 (86 employees, 1.37%), compare to other years that hired at least 200 employees per year.

5.6 - "TERMINATION_DATE" COLUMN

This column indicates the dates on which the employees were terminated. Since active employees still have a termination date assigned to them, we will filter the data accordingly beforehand so that only terminated employee data are used in the analysis.

```
## To extract TERMINATED employees from the data
termdate <- filter(data, STATUS=="TERMINATED")

## Data summary
summary(termdate$TERMINATION_DATE)

## Unique data
unique(termdate$TERMINATION_DATE)

## Counting the amount of distinct values
n_distinct(termdate$TERMINATION_DATE)
```

Code: Summary, Unique Data, and Distinct Values of the "TERMINATION_DATE" Column

```
> summary(termdate$TERMINATION_DATE)
        Min.      1st Qu.       Median         Mean      3rd Qu.         Max.
"2006-01-01" "2008-06-05" "2011-02-14" "2011-04-08" "2014-08-10" "2015-12-30"
```

Output: Summary of the "TERMINATION_DATE" Column

According to the summary, the earliest termination date is on the 1st of January 2006, while the latest termination date is on the 30th of December 2015.

```
> unique(termdate$TERMINATION_DATE)
   [1] "2009-02-16" "2014-02-19" "2009-02-19" "2009-02-21" "2014-02-24" "2014-02-25" "2014-03-04" "2009-03-04" "2009-03-09"
  [10] "2014-03-11" "2009-03-13" "2009-03-15" "2009-03-14" "2014-03-16" "2014-03-17" "2014-03-18" "2009-03-20" "2012-09-26"
  [19] "2009-03-26" "2009-03-27" "2014-04-01" "2009-04-05" "2014-04-04" "2009-04-06" "2009-04-11" "2014-04-15" "2014-04-17"
  [28] "2014-04-20" "2009-04-20" "2009-04-23" "2009-04-30" "2009-05-06" "2014-05-07" "2009-05-07" "2009-05-08" "2014-05-10"
  [37] "2014-05-11" "2009-05-10" "2014-05-17" "2014-05-16" "2014-12-30" "2014-05-20" "2009-05-28" "2014-05-27" "2009-05-31"
  [46] "2009-06-02" "2014-06-12" "2009-06-12" "2009-06-16" "2009-06-18" "2014-06-30" "2014-07-05" "2009-07-12" "2009-07-18"
  [55] "2009-07-20" "2014-07-30" "2009-08-01" "2009-08-04" "2014-08-10" "2009-08-14" "2009-08-15" "2014-08-18" "2009-12-30"
  [64] "2009-08-27" "2009-08-26" "2014-08-29" "2014-09-03" "2009-09-04" "2014-09-05" "2009-09-05" "2009-09-06" "2014-09-14"
  [73] "2009-10-01" "2014-10-04" "2009-10-11" "2009-10-12" "2014-10-12" "2014-10-15" "2014-10-16" "2009-10-18" "2009-10-20"
  [82] "2014-10-23" "2014-11-02" "2014-11-04" "2009-11-07" "2014-11-13" "2014-11-12" "2009-11-16" "2014-11-22" "2007-07-25"
  [91] "2009-11-24" "2009-11-25" "2014-11-27" "2014-11-30" "2014-12-02" "2009-12-03" "2014-12-04" "2014-12-13" "2014-12-14"
 [100] "2014-12-15" "2014-12-19" "2014-12-20" "2015-01-01" "2015-01-05" "2010-01-05" "2015-01-15" "2010-01-19" "2015-01-21"
 [109] "2015-01-31" "2010-12-30" "2010-01-30" "2015-02-01" "2010-02-02" "2015-02-04" "2010-02-06" "2015-02-06" "2015-02-11"
```

```
> n_distinct(termdate$TERMINATION_DATE)
[1] 1049
```

Output: Top Few Rows of Checking Unique Data and Number of Distinct Values of the "TERMINATION_DATE" Column

The "TERMINATION_DATE" column is checked for unique values. There are 1049 unique dates in total.

```
class(termdate$TERMINATION_DATE)
```

Code: Datatype of "TERMINATION_DATE" Column

```
> class(termdate$TERMINATION_DATE)
[1] "Date"
```

Output: Datatype of "TERMINATION_DATE" Column

As converted earlier, the datatype of the column is Date.

```
exp_termdate <- tabyl(termdate, TERMINATION_DATE) %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE)
tail(exp_termdate,10)
```

Code: List of the Frequency of Each Date of the "TERMINATION_DATE" Column

```
> tail(exp_termdate,10)
 TERMINATION_DATE  n percent
       2015-12-01  1   0.07%
       2015-12-06  1   0.07%
       2015-12-11  1   0.07%
       2015-12-17  1   0.07%
       2015-12-21  1   0.07%
       2015-12-24  1   0.07%
       2015-12-26  1   0.07%
       2015-12-28  1   0.07%
       2015-12-29  1   0.07%
       2015-12-30 76   5.14%
```

Output: Last 10 Rows of the List of the Frequency of Each Date of the "TERMINATION_DATE" Column

From the output, we can see that most termination dates are rarely repeated, except for 30th December 2015 as there were 76 terminations (5.14%).

```
summary(exp_termdate$n)
```

Code: Summary of the Frequency of Each "TERMINATION_DATE"

```
> summary(exp_termdate$n)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   1.000   1.000   1.411   1.000 144.000
```

Output: Summary of the Frequency of Each "TERMINATION_DATE"

The findings show that the most people in the dataset do not have clashing termination dates, as the 3rd quartile of the data still shows 1. However, a huge amount of employees (144 employees) were terminated all in one day.

```
## Extracting just the year value of the dates
exp_termdate_year = format(as.Date(termdate$TERMINATION_DATE, format="%Y/%m/%d"),"%Y")

## Counting the amount of distinct years
n_distinct(exp_termdate_year)
```

Code: Extracting Year Value from "TERMINATION_DATE" and Counting the Distinct Years

```
> ## Extracting just the year value of the dates
> exp_termdate_year = format(as.Date(termdate$TERMINATION_DATE, format="%Y/%m/%d"),"%Y")
>
> ## Counting the amount of distinct years
> n_distinct(exp_termdate_year)
[1] 10
```

Output: Counting the Distinct Years of "TERMINATION_DATE" Column

The results show that there are 10 distinct years in the "TERMINATION_DATE" Column.

```
## Arranging the years and their frequency
exp_termdate_year <- tabyl(exp_termdate_year) %>%
  dplyr::arrange( n ) %>%
  dplyr::arrange(exp_termdate_year)
names(exp_termdate_year)[1] <- "Year"

## TABLE: Frequency of TERMINATION_DATE by Year
exp1_termdate <- exp_termdate_year %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
exp1_termdate
```

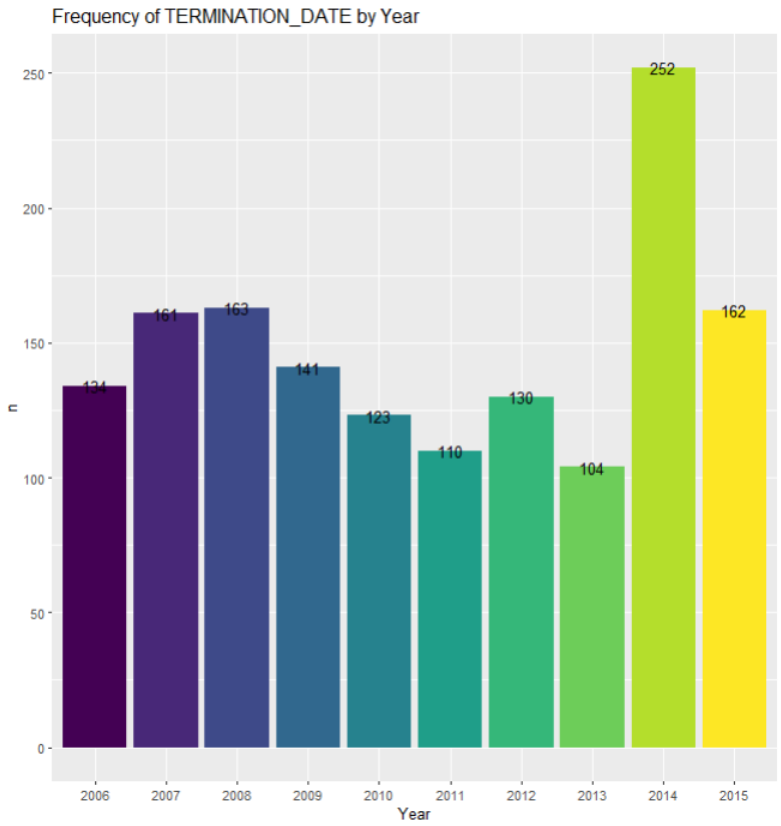
Code: Generating a Table of the Frequency of "TERMINATION_DATE" by Year

```
> exp1_termdate
   Year    n percent
   2006  134   9.05%
   2007  161  10.88%
   2008  163  11.01%
   2009  141   9.53%
   2010  123   8.31%
   2011  110   7.43%
   2012  130   8.78%
   2013  104   7.03%
   2014  252  17.03%
   2015  162  10.95%
  Total 1480 100.00%
```

Output: Table of the Frequency of "TERMINATION_DATE" by Year

```
## CHART: Frequency of TERMINATION_DATE by Year
exp_termdate_vis <- exp_termdate_year
ggplot(exp_termdate_vis, aes(y=n, x=Year)) +
  geom_bar(stat="identity",
           fill=viridis(n_distinct(exp_termdate_year))) +
  geom_text(aes(label=n)) +
  ggtitle("Frequency of TERMINATION_DATE by Year")
```

Code: Generating a Bar Graph of the Frequency of "TERMINATION_DATE" by Year



Graph: Bar Graph and Path Graph of the Frequency of "TERMINATION_DATE" by Year

In the data, we can see that most employees were terminated in 2014 (252 employees, 17.03%). In contrast, 2013 had the least amount of employees terminated (104 employees, 7.03%). After the peak of 252 employees terminated in 2014, the amount of terminated employees in the following year greatly decreased to 162 (10.95%).

5.7 - "AGE" COLUMN

This column indicates the age of the employees.

```
## Data summary
summary(data$AGE)

## Unique data
unique(data$AGE)

## Counting the amount of distinct values
n_distinct(data$AGE)
```

Code: Summary, Unique Data, and Distinct Values of the "AGE" Column

```
> summary(data$AGE)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  19.00   32.00   45.00   44.74   58.00   65.00
```

Output: Summary of the "AGE" Column

According to the summary, the youngest employee is 19 years old, while the oldest employee currently is 65 years old.

```
> unique(data$AGE)
 [1] 61 58 60 56 57 53 51 59 48 50 54 65 63 64 55 62 52 49 47 46 44 41 43 45 42 40 39 38
[29] 37 35 36 34 33 32 31 29 28 30 26 25 27 24 23 22 21 20 19
> n_distinct(data$AGE)
[1] 47
```

Output: Checking Unique Data and Number of Distinct Values of the "AGE" Column

The "AGE" column is checked for unique values. There are 47 unique ages in total.

```
class(data$AGE)
```

Code: Datatype of "AGE" Column

```
> class(data$AGE)
[1] "integer"
```

Output: Datatype of "AGE" Column

The datatype of this column is Integer.

```
## TABLE: Frequency of AGE
exp_age <- tabyl(data, AGE) %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE)
exp_age
```

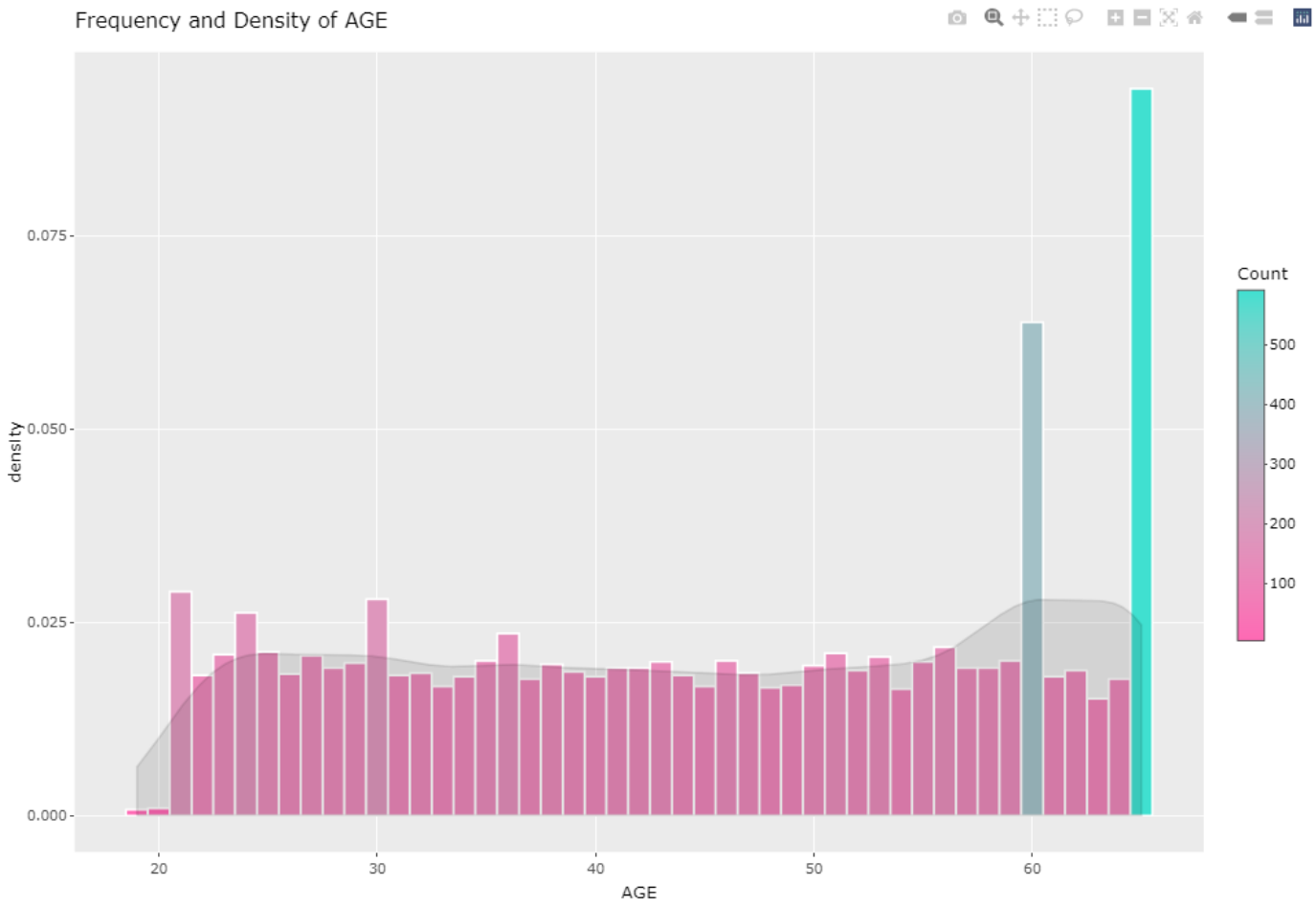
Code: Generating a Table of the Frequency of "AGE"

```
> exp_age
   AGE    n percent
    19    5   0.08%
    20    6   0.10%
    21  182   2.90%
    22  114   1.81%
    23  131   2.08%
    24  165   2.63%
    25  133   2.12%
    26  115   1.83%
    27  130   2.07%
    28  120   1.91%
    29  124   1.97%
    30  176   2.80%
    31  114   1.81%
    32  116   1.85%
    33  105   1.67%
    34  113   1.80%
    35  126   2.01%
    36  148   2.36%
    37  111   1.77%
    38  123   1.96%
    39  117   1.86%
    40  113   1.80%
    41  120   1.91%
    42  120   1.91%
    43  125   1.99%
    44  114   1.81%
    45  105   1.67%
    46  126   2.01%
    47  116   1.85%
    48  104   1.65%
    49  106   1.69%
    50  122   1.94%
    51  132   2.10%
    52  118   1.88%
    53  129   2.05%
    54  103   1.64%
    55  125   1.99%
    56  137   2.18%
    57  120   1.91%
    58  120   1.91%
    59  126   2.01%
    60  401   6.38%
    61  113   1.80%
    62  118   1.88%
    63   95   1.51%
    64  111   1.77%
    65  591   9.40%
 Total 6284 100.00%
```

Output: Table of the Frequency of "AGE"

```
## CHART: Frequency and Density of AGE
ggplotly(ggplot(data, aes(x=AGE)) +
    geom_histogram(colour="white", aes(y=..density.., fill=..count..), binwidth=1) +
    scale_fill_gradient("Count", low="hotpink", high="turquoise") +
    geom_density(alpha=.1, fill="black") +
    ggtitle("Frequency and Density of AGE"))
```

Code: Generating a Histogram and Density Graph of the Frequency and Density of "AGE"



Graph: Histogram and Density Graph of the Frequency and Density of "AGE"

From the output, we can see that only 5 employees (0.08%) in the age of 19 was hired, followed by 6 employees (0.10%) in the age of 20. Currently, there are 95 employees (1.51%) that are 63 years old, and all others ages have more than 100 employees. Because this dataset includes workers who were terminated in prior years, the age groups of 60 and 65 had significantly more employees than the others (401 employees (6.38%) and 591 employees (9.40%), respectively). This sudden increase of employees had also caused the density graph to be more denser as the age became older.

5.8 - "LENGTH_OF_SERVICE" COLUMN

This column indicates the length of years that the employees had served in the company.

```
## Data summary
summary(data$LENGTH_OF_SERVICE)

## Unique data
unique(data$LENGTH_OF_SERVICE)

## Counting the amount of distinct values
n_distinct(data$LENGTH_OF_SERVICE)
```

Code: Summary, Unique Data, and Distinct Values of the "LENGTH_OF_SERVICE" Column

```
> summary(data$LENGTH_OF_SERVICE)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.00    7.00   13.00   12.84   19.00   26.00
```

Output: Summary of the "LENGTH_OF_SERVICE" Column

According to the summary, some employees did not even serve for 1 year, while the longest years served in the company is 26 years.

```
> unique(data$LENGTH_OF_SERVICE)
 [1] 26 19 24 22 25 20 17 21 16 23 15 18 14 13 12 11 10  8  9  7  6  5  4  3  2  1  0
```

Output: Checking Unique Data of the "LENGTH_OF_SERVICE" Column

The "LENGTH_OF_SERVICE" column is checked for unique values. The length of service of the employees in the company is continuous from 0 to 26.

```
class(data$LENGTH_OF_SERVICE)
```

Code: Datatype of "LENGTH_OF_SERVICE" Column

```
> class(data$LENGTH_OF_SERVICE)
[1] "integer"
```

Output: Datatype of "LENGTH_OF_SERVICE" Column

The datatype of this column is Integer.

```
## TABLE: Frequency of LENGTH_OF_SERVICE
exp_len_service <- tabyl(data, LENGTH_OF_SERVICE) %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE)
exp_len_service
```

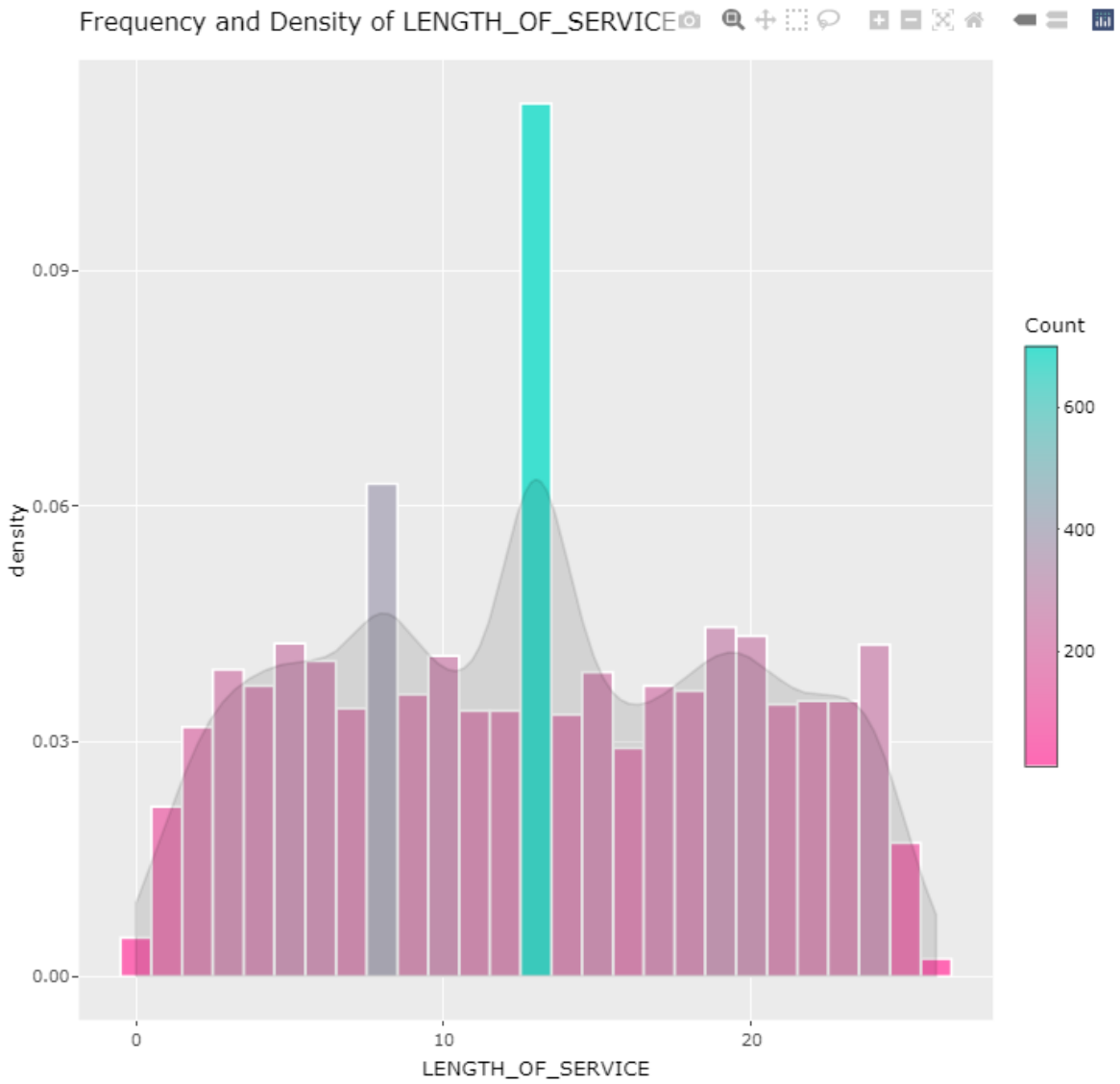
Code: Generating a Table of the Frequency of "LENGTH_OF_SERVICE"

LENGTH_OF_SERVICE	n	percent
0	31	0.49%
1	136	2.16%
2	200	3.18%
3	246	3.91%
4	233	3.71%
5	267	4.25%
6	253	4.03%
7	215	3.42%
8	395	6.29%
9	226	3.60%
10	257	4.09%
11	213	3.39%
12	213	3.39%
13	700	11.14%
14	210	3.34%
15	244	3.88%
16	183	2.91%
17	233	3.71%
18	229	3.64%
19	280	4.46%
20	273	4.34%
21	218	3.47%
22	221	3.52%
23	221	3.52%
24	266	4.23%
25	107	1.70%
26	14	0.22%
Total	6284	100.00%

Output: Table of the Frequency of "LENGTH_OF_SERVICE"

```
## CHART: Frequency and Density of LENGTH_OF_SERVICE
ggplotly(ggplot(data, aes(x=LENGTH_OF_SERVICE)) +
   geom_histogram(colour="white", aes(y=..density.., fill=..count..), binwidth=1) +
   scale_fill_gradient("Count", low="hotpink", high="turquoise") +
   geom_density(alpha=.1, fill="black") +
   ggtitle("Frequency and Density of LENGTH_OF_SERVICE"))
```

Code: Generating a Histogram and Density Graph of the Frequency and Density of "LENGTH_OF_SERVICE"



Graph: Histogram and Density Graph of the Frequency and Density of "LENGTH_OF_SERVICE"

From the output, we can see that only 14 employees (0.22%) served for 26 years in the company, followed by 31 employees who currently served for less than 1 year in the company. Besides that, the majority of employees (700 employees (11.14%)) have served for 13 years in the company followed by the second highest value of 8 years with 195 employees (6.29%). The sudden increase of employees working for 13 years had also caused the density graph to be more denser towards the middle.

5.9 - “CITY” Column

This column consists of the locations of all of the company's stores.

```
## Data summary
summary(data$CITY)

## Unique data
unique(data$CITY)

## Counting the amount of distinct values
nlevels(data$CITY)
```

Code: Checking Summary, Unique data, and the Number of Distinct Values

```
> summary(data$CITY)
     Abbotsford      Aldergrove     Bella Bella      Blue River         Burnaby      Chilliwack
             90              65              18               1             258             147
  Cortes Island       Cranbrook    Dawson Creek     Dease Lake     Fort Nelson    Fort St John
              6             219              18               2              47              89
    Grand Forks           Haney        Kamloops         Kelowna         Langley         Nanaimo
             33              24             267             305             118             481
         Nelson New Westminster  North Vancouver     Ocean Falls    Pitt Meadows  Port Coquitlam
             39             447              81               7               9              67
  Prince George       Princeton         Quesnel        Richmond        Squamish          Surrey
            264              19              94             176             101             197
        Terrace           Trail       Valemount       Vancouver         Vernon        Victoria
            160             121               5            1392             109             624
 West Vancouver      White Rock   Williams Lake
             84              28              72
> unique(data$CITY)
 [1] Vancouver       Terrace         Nanaimo         Nelson          Kelowna
 [6] Victoria        Kamloops        Fort St John    Surrey          Vernon
[11] Quesnel         Chilliwack      Dawson Creek    Squamish        New Westminster
[16] Port Coquitlam  Cortes Island   Burnaby         Bella Bella     Cranbrook
[21] Williams Lake   Trail           Prince George   Richmond        Grand Forks
[26] West Vancouver  Abbotsford      Aldergrove      Langley         North Vancouver
[31] White Rock      Fort Nelson     Haney           Valemount       Ocean Falls
[36] Pitt Meadows    Princeton       Dease Lake      Blue River
39 Levels: Abbotsford Aldergrove Bella Bella Blue River Burnaby Chilliwack ... Williams Lake
> nlevels(data$CITY)
[1] 39
```

Output: Summary,Unique data, and the Number of Distinct Values of the “CITY” Column

From the output, we can see that there are 39 unique cities in total, with all cities located in British Columbia, Canada.

```
class(data$CITY)
```

Code: Datatype of “CITY” Column

```
> class(data$CITY)
[1] "factor"
```

Output: Datatype of “CITY” Column

As converted earlier, the datatype of the column is Factor.

```r
## List of the frequency of each values
exp_city <- tabyl(data, CITY)

## TABLE: Frequency of CITY
exp1_city <- exp_city %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
exp1_city
```

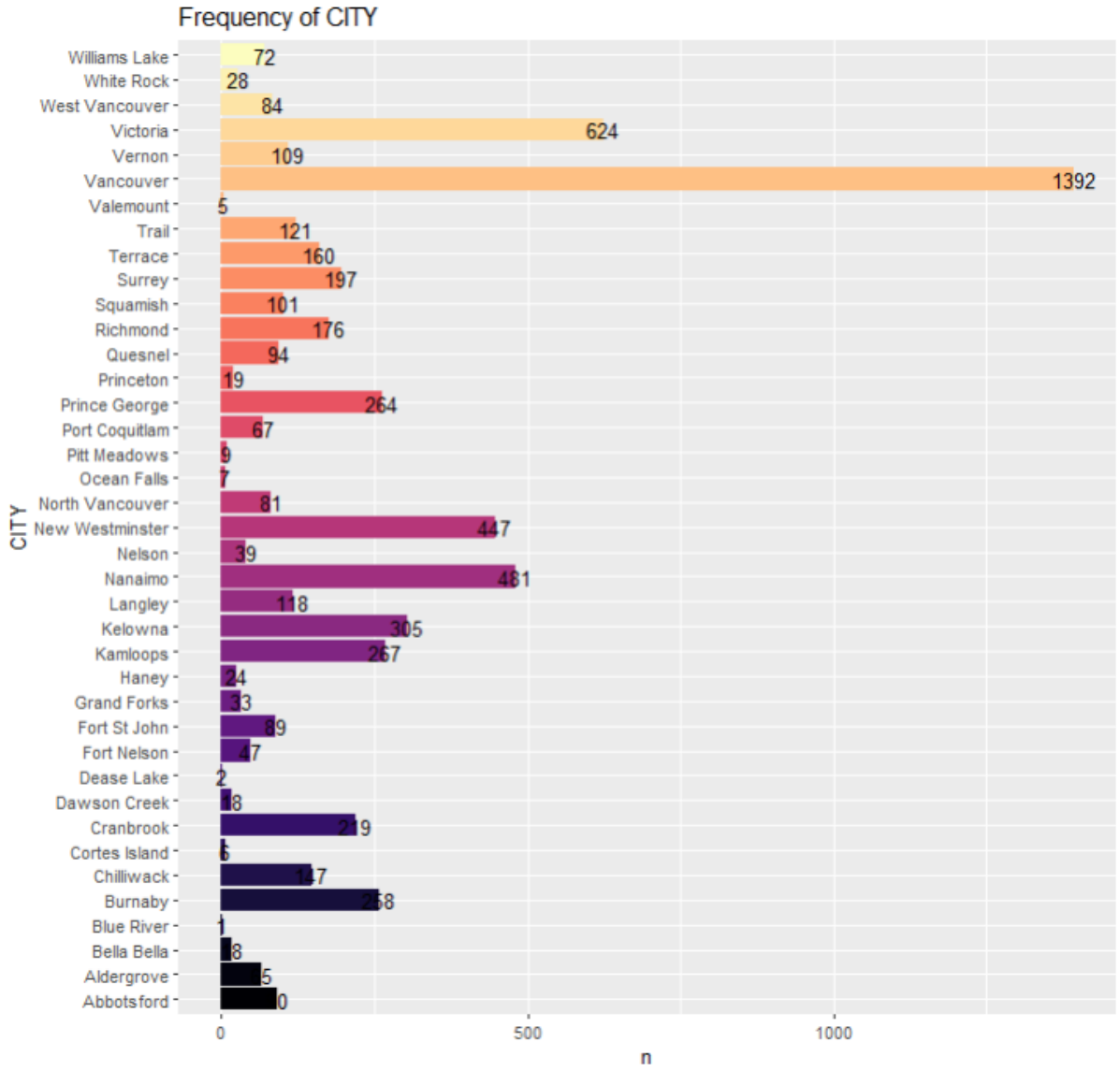
Code: Generating a Table of the Frequency of "CITY"

```
> exp1_city
            CITY    n percent
       Abbotsford   90   1.43%
       Aldergrove   65   1.03%
     Bella Bella    18   0.29%
      Blue River     1   0.02%
          Burnaby  258   4.11%
      Chilliwack   147   2.34%
   Cortes Island     6   0.10%
       Cranbrook   219   3.49%
    Dawson Creek    18   0.29%
      Dease Lake     2   0.03%
     Fort Nelson    47   0.75%
    Fort St John    89   1.42%
     Grand Forks    33   0.53%
           Haney    24   0.38%
        Kamloops   267   4.25%
         Kelowna   305   4.85%
         Langley   118   1.88%
         Nanaimo   481   7.65%
          Nelson    39   0.62%
 New Westminster   447   7.11%
 North Vancouver    81   1.29%
     Ocean Falls     7   0.11%
    Pitt Meadows     9   0.14%
  Port Coquitlam    67   1.07%
   Prince George   264   4.20%
        Princeton    19   0.30%
          Quesnel    94   1.50%
        Richmond   176   2.80%
        Squamish   101   1.61%
          Surrey   197   3.13%
         Terrace   160   2.55%
           Trail   121   1.93%
       Valemount     5   0.08%
       Vancouver  1392  22.15%
          Vernon   109   1.73%
        Victoria   624   9.93%
  West Vancouver    84   1.34%
      White Rock    28   0.45%
   Williams Lake    72   1.15%
           Total  6284 100.00%
```

Output: Table of the Frequency of "CITY"

```
ggplot(exp_city, aes(x=n, y=CITY)) +
  geom_bar(stat="identity", fill=magma(nlevels(data$CITY))) +
  geom_text(aes(label=n)) +
  ggtitle("Frequency of CITY")
```

Code: Generating a Bar Graph of the Frequency of "CITY"



Graph: Bar Graph of the Frequency of "CITY"

The output reveal that most employees are woking in Vancouver (1392 employees (22.15%)), followed by Victoria which has 624 employees (9.93%). The location that had the least amount of employees is Blue River, with only 1 employee working there (0.02%).

5.10 - “DEPARTMENT” Column

This column consists of all the departments of the company.

```
## Data summary
summary(data$DEPARTMENT)

## Unique data
unique(data$DEPARTMENT)

## Counting the amount of distinct values
nlevels(data$DEPARTMENT)
```

Code: Checking Summary, Unique data, and the Number of Distinct Values

```
> summary(data$DEPARTMENT)
            Accounting       Accounts Payable   Accounts Receiveable                  Audit
                     6                      4                      5                      4
                Bakery           Compensation       Customer Service                  Dairy
                   898                      4                   1190                   1033
      Employee Records              Executive         HR Technology Information Technology
                     6                     10                      9                      5
            Investment        Labor Relations                  Legal                  Meats
                     4                      6                      3                   1252
       Processed Foods               Produce            Recruitment       Store Management
                   736                   1060                      9                     35
              Training
                     5
> unique(data$DEPARTMENT)
 [1] Executive              Store Management       Meats                  Recruitment
 [5] Training               Labor Relations        HR Technology          Employee Records
 [9] Compensation           Legal                  Produce                Accounts Receiveable
[13] Bakery                 Information Technology Accounts Payable       Audit
[17] Accounting             Investment             Dairy                  Processed Foods
[21] Customer Service
21 Levels: Accounting Accounts Payable Accounts Receiveable Audit Bakery ... Training
> nlevels(data$DEPARTMENT)
[1] 21
```

Output: Summary,Unique data, and the Number of Distinct Values of the “DEPARTMENT” Column

From the output, we can see that there are 21 unique departments in total.

```
class(data$DEPARTMENT)
```

Code: Datatype of “DEPARTMENT” Column

```
> class(data$DEPARTMENT)
[1] "factor"
```

Output: Datatype of “DEPARTMENT” Column

As converted earlier, the datatype of the column is Factor.

```r
## List of the frequency of each values
exp_dept <- tabyl(data, DEPARTMENT)

## TABLE: Frequency of DEPARTMENT
exp1_dept <- exp_dept %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
exp1_dept
```

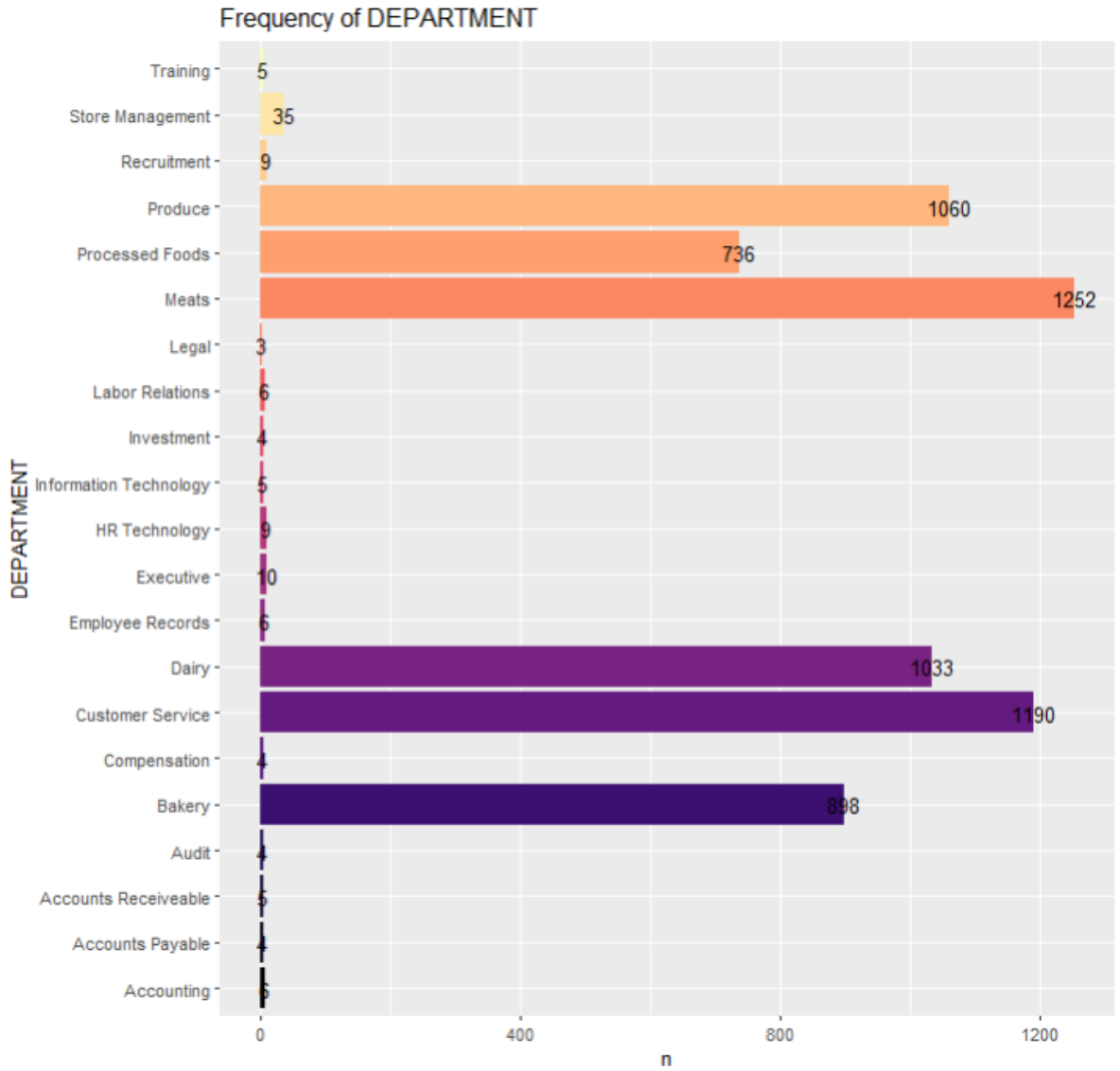
Code: Generating a Table of the Frequency of "DEPARTMENT"

```
> exp1_dept
             DEPARTMENT    n percent
             Accounting    6   0.10%
       Accounts Payable    4   0.06%
  Accounts Receiveable     5   0.08%
                 Audit     4   0.06%
                Bakery   898  14.29%
          Compensation     4   0.06%
       Customer Service 1190  18.94%
                 Dairy  1033  16.44%
       Employee Records    6   0.10%
              Executive   10   0.16%
         HR Technology     9   0.14%
 Information Technology    5   0.08%
            Investment     4   0.06%
        Labor Relations    6   0.10%
                 Legal     3   0.05%
                 Meats  1252  19.92%
        Processed Foods    736  11.71%
               Produce   1060  16.87%
           Recruitment     9   0.14%
       Store Management   35   0.56%
              Training     5   0.08%
                 Total  6284 100.00%
```

Output: Table of the Frequency of "DEPARTMENT"

```r
## CHART: Frequency of DEPARTMENT
ggplot(exp_dept, aes(x=n, y=DEPARTMENT)) +
  geom_bar(stat="identity", fill=magma(nlevels(data$DEPARTMENT))) +
  geom_text(aes(label=n)) +
  ggtitle("Frequency of DEPARTMENT")
```

Code: Generating a Bar Graph of the Frequency of "DEPARTMENT"



Graph: Bar Graph of the Frequency of "DEPARTMENT"

The output reveal that most employees are working in the Meats department (1252 employees (19.92%)), followed by the Customer Service department which has 1190 employees (18.94%). The department that had the least amount of employees is the Legal department, with only 3 employees working there (0.05%). We can see a pattern that most production related departments had way more employees compared to the executive departments.

5.11 - “JOB_TITLE” Column

This column consists of all the jobs available in the company.

```
## Data summary
summary(data$JOB_TITLE)

## Unique data
unique(data$JOB_TITLE)

## Counting the amount of distinct values
nlevels(data$JOB_TITLE)
```

Code: Checking Summary, Unique data, and the Number of Distinct Values

```
> summary(data$JOB_TITLE)
               Accounting Clerk          Accounts Payable Clerk      Accounts Receiveable Clerk
                              5                               3                               4
                        Auditor                           Baker                   Bakery Manager
                              3                             865                              33
                  Benefits Admin                         Cashier                             CEO
                              5                            1158                               1
       Chief Information Officer            Compensation Analyst                Corporate Lawyer
                              1                               3                               3
        Customer Service Manager                   Dairy Manager                    Dairy Person
                             32                               1                            1032
            Director, Accounting      Director, Accounts Payable   Director, Accounts Receivable
                              1                               1                               1
                 Director, Audit          Director, Compensation      Director, Employee Records
                              1                               1                               1
          Director, HR Technology           Director, Investments       Director, Labor Relations
                              1                               1                               1
            Director, Recruitment              Director, Training         Exec Assistant, Finance
                              1                               1                               1
Exec Assistant, Human Resources   Exec Assistant, Legal Counsel       Exec Assistant, VP Stores
                              1                               1                               1
                    HRIS Analyst              Investment Analyst         Labor Relations Analyst
                              8                               3                               5
                  Legal Counsel                     Meat Cutter                    Meats Manager
                              1                            1218                              34
         Processed Foods Manager                   Produce Clerk                 Produce Manager
                             32                            1027                              33
                       Recruiter                   Shelf Stocker                   Store Manager
                              8                             704                              35
                Systems Analyst                         Trainer                      VP Finance
                              5                               4                               1
             VP Human Resources                       VP Stores
                              1                               1
> unique(data$JOB_TITLE)
 [1] CEO                              VP Stores                          Legal Counsel
 [4] VP Human Resources               VP Finance                         Exec Assistant, VP Stores
 [7] Exec Assistant, Legal Counsel    Chief Information Officer          Store Manager
[10] Meats Manager                    Exec Assistant, Human Resources    Exec Assistant, Finance
[13] Director, Recruitment            Director, Training                 Director, Labor Relations
[16] Director, HR Technology          Director, Employee Records         Director, Compensation
[19] Corporate Lawyer                 Produce Manager                    Director, Accounts Receivable
[22] Bakery Manager                   Systems Analyst                    Director, Accounts Payable
[25] Director, Audit                  Director, Accounting               Director, Investments
[28] Dairy Person                     Recruiter                          Processed Foods Manager
[31] Customer Service Manager         Trainer                            Meat Cutter
[34] Labor Relations Analyst          Dairy Manager                      HRIS Analyst
[37] Benefits Admin                   Compensation Analyst               Accounts Receiveable Clerk
[40] Accounts Payable Clerk           Baker                              Auditor
[43] Accounting Clerk                 Investment Analyst                 Produce Clerk
[46] Shelf Stocker                    Cashier
47 Levels: Accounting Clerk Accounts Payable Clerk Accounts Receiveable Clerk Auditor Baker ... VP Stores
> nlevels(data$JOB_TITLE)
[1] 47
```

Output: Summary,Unique data, and the Number of Distinct Values of the "JOB_TITLE" Column

From the output, we can see that there are 47 unique jobs in total, with some jobs categorised into Directors and Executive Assistants.

```
class(data$JOB_TITLE)
```

Code: Datatype of "JOB_TITLE" Column

```
> class(data$JOB_TITLE)
[1] "factor"
```

Output: Datatype of "JOB_TITLE" Column

As converted earlier, the datatype of the column is Factor.

```
## List of the frequency of each values
exp_job <- tabyl(data, JOB_TITLE)

## List of the frequency of each values (Director)
select(exp_job, c(JOB_TITLE,n)) %>%
  dplyr::filter(grepl('Director', JOB_TITLE)) %>%
  adorn_totals()
```

Code: Generating a Table of the Frequency of Directors in "JOB_TITLE"

```
> exp_job <- tabyl(data, JOB_TITLE)
> select(exp_job, c(JOB_TITLE,n)) %>%
+   dplyr::filter(grepl('Director', JOB_TITLE)) %>%
+   adorn_totals()
                      JOB_TITLE  n
            Director, Accounting  1
       Director, Accounts Payable  1
    Director, Accounts Receivable  1
                  Director, Audit  1
           Director, Compensation  1
       Director, Employee Records  1
          Director, HR Technology  1
           Director, Investments  1
        Director, Labor Relations  1
            Director, Recruitment  1
               Director, Training  1
                           Total 11
```

Output: Table of the Frequency of Directors in "JOB_TITLE"

There are a total of 11 Directors for 11 departments in "JOB_TITLE" column.

```
## List of the frequency of each values (Exec Assistant)
select(exp_job, c(JOB_TITLE,n)) %>%
  dplyr::filter(grepl('Exec Assistant', JOB_TITLE)) %>%
  adorn_totals()
```

Code: Generating a Table of the Executive Assistants of Directors in "JOB_TITLE"

```
> select(exp_job, c(JOB_TITLE,n)) %>%
+   dplyr::filter(grepl('Exec Assistant', JOB_TITLE)) %>%
+   adorn_totals()
                      JOB_TITLE n
        Exec Assistant, Finance 1
 Exec Assistant, Human Resources 1
   Exec Assistant, Legal Counsel 1
      Exec Assistant, VP Stores 1
                          Total 4
```

Output: Table of the Frequency of Executive Assistants in "JOB_TITLE"

There are a total of 4 Executive Assistants for 4 departments in "JOB_TITLE" column.

```
## TABLE: Frequency of JOB_TITLE
exp1_job <- exp_job %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
exp1_job
```

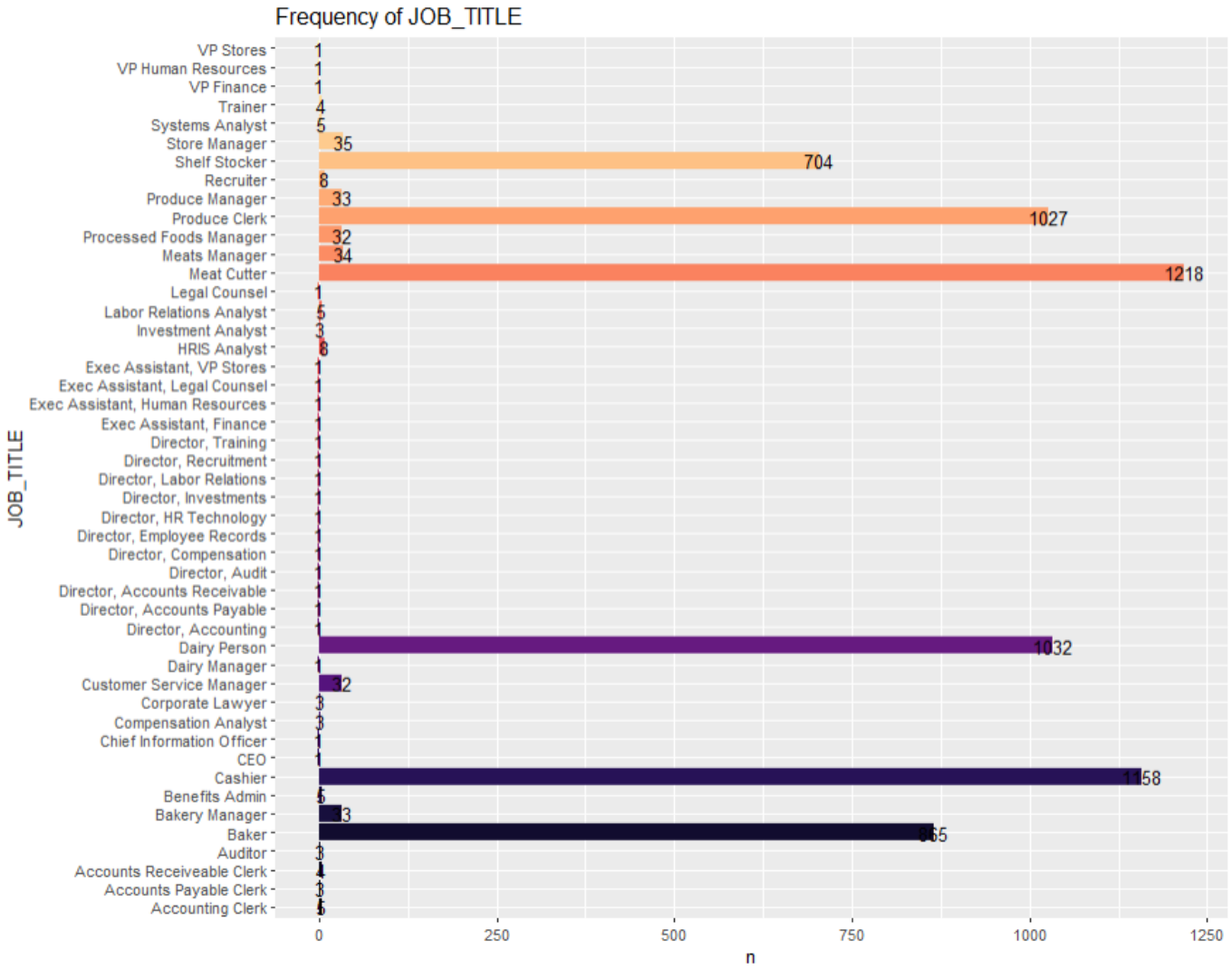
Code: Generating a Table of the Frequency of "JOB_TITLE"

```
> exp1_job
                      JOB_TITLE    n percent
                Accounting Clerk    5   0.08%
          Accounts Payable Clerk    3   0.05%
       Accounts Receiveable Clerk    4   0.06%
                         Auditor    3   0.05%
                           Baker  865  13.77%
                  Bakery Manager   33   0.53%
                  Benefits Admin    5   0.08%
                         Cashier 1158  18.43%
                             CEO    1   0.02%
       Chief Information Officer    1   0.02%
            Compensation Analyst    3   0.05%
                Corporate Lawyer    3   0.05%
         Customer Service Manager   32   0.51%
                   Dairy Manager    1   0.02%
                    Dairy Person 1032  16.42%
            Director, Accounting    1   0.02%
       Director, Accounts Payable    1   0.02%
    Director, Accounts Receivable    1   0.02%
                  Director, Audit    1   0.02%
           Director, Compensation    1   0.02%
       Director, Employee Records    1   0.02%
          Director, HR Technology    1   0.02%
            Director, Investments    1   0.02%
         Director, Labor Relations    1   0.02%
             Director, Recruitment    1   0.02%
                Director, Training    1   0.02%
          Exec Assistant, Finance    1   0.02%
  Exec Assistant, Human Resources    1   0.02%
    Exec Assistant, Legal Counsel    1   0.02%
       Exec Assistant, VP Stores    1   0.02%
                     HRIS Analyst    8   0.13%
              Investment Analyst    3   0.05%
          Labor Relations Analyst    5   0.08%
                   Legal Counsel    1   0.02%
                     Meat Cutter 1218  19.38%
                   Meats Manager   34   0.54%
          Processed Foods Manager   32   0.51%
                   Produce Clerk 1027  16.34%
                 Produce Manager   33   0.53%
                        Recruiter    8   0.13%
                    Shelf Stocker  704  11.20%
                    Store Manager   35   0.56%
                 Systems Analyst    5   0.08%
                         Trainer    4   0.06%
                      VP Finance    1   0.02%
              VP Human Resources    1   0.02%
                        VP Stores    1   0.02%
                            Total 6284 100.00%
```

Output: Table of the Frequency of "JOB_TITLE"

```
## CHART: Frequency of JOB_TITLE
ggplot(exp_job, aes(x=n, y=JOB_TITLE)) +
  geom_bar(stat="identity", fill=magma(nlevels(data$JOB_TITLE))) +
  geom_text(aes(label=n)) +
  ggtitle("Frequency of JOB_TITLE")
```

Code: Generating a Bar Graph of the Frequency of "JOB_TITLE"



Graph: Bar Graph of the Frequency of "JOB_TITLE"

The output shows that most employees are working as a Meat Cutter in the Meats department (1218 employees (19.38%)), followed by the Cashier in the Customer Service department which has 1158 employees (18.43%). The jobs that had the least amount of employees (1 employee (0.02%)) are mostly the directors, executive assistants and other executive roles.

5.12 - “STORE” Column

This column consists of all the store names of the company.

```
## Data summary
summary(data$STORE)

## Unique data
unique(data$STORE)

## Counting the amount of distinct values
nlevels(data$STORE)
```

Code: Checking Summary, Unique data, and the Number of Distinct Values

```
> summary(data$STORE)
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26
 90  65  18   1 258 147   6 219  18   2  47  89  33  24 267 305 118 481  39  44 403  81   7   9  67 264
 27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46
 19  94 176 101 197 160 121   5 221 109 102  84  28  72 178 392 311 284   6 522
> unique(data$STORE)
 [1] 35 32 18 19 16 37 15 12 31 36 28 6  9  30 21 46 25 7  5  3  8  40 33 26 29 41 13 38 42 1  2  43 44 17
[35] 45 22 39 20 11 14 34 23 24 27 10 4
46 Levels: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 ... 46
> nlevels(data$STORE)
[1] 46
```

Output: Summary,Unique data, and the Number of Distinct Values of the “STORE” Column

From the output, we can see that there are 46 unique store names that are named after numbers.

```
class(data$JOB_TITLE)
```

Code: Datatype of “STORE” Column

```
> class(data$JOB_TITLE)
[1] "factor"
```

Output: Datatype of “STORE” Column

As converted earlier, the datatype of the column is Factor.

```
## List of the frequency of each values
exp_store <- tabyl(data, STORE)

## TABLE: Frequency of STORE
exp1_store <- exp_store %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
exp1_store
```

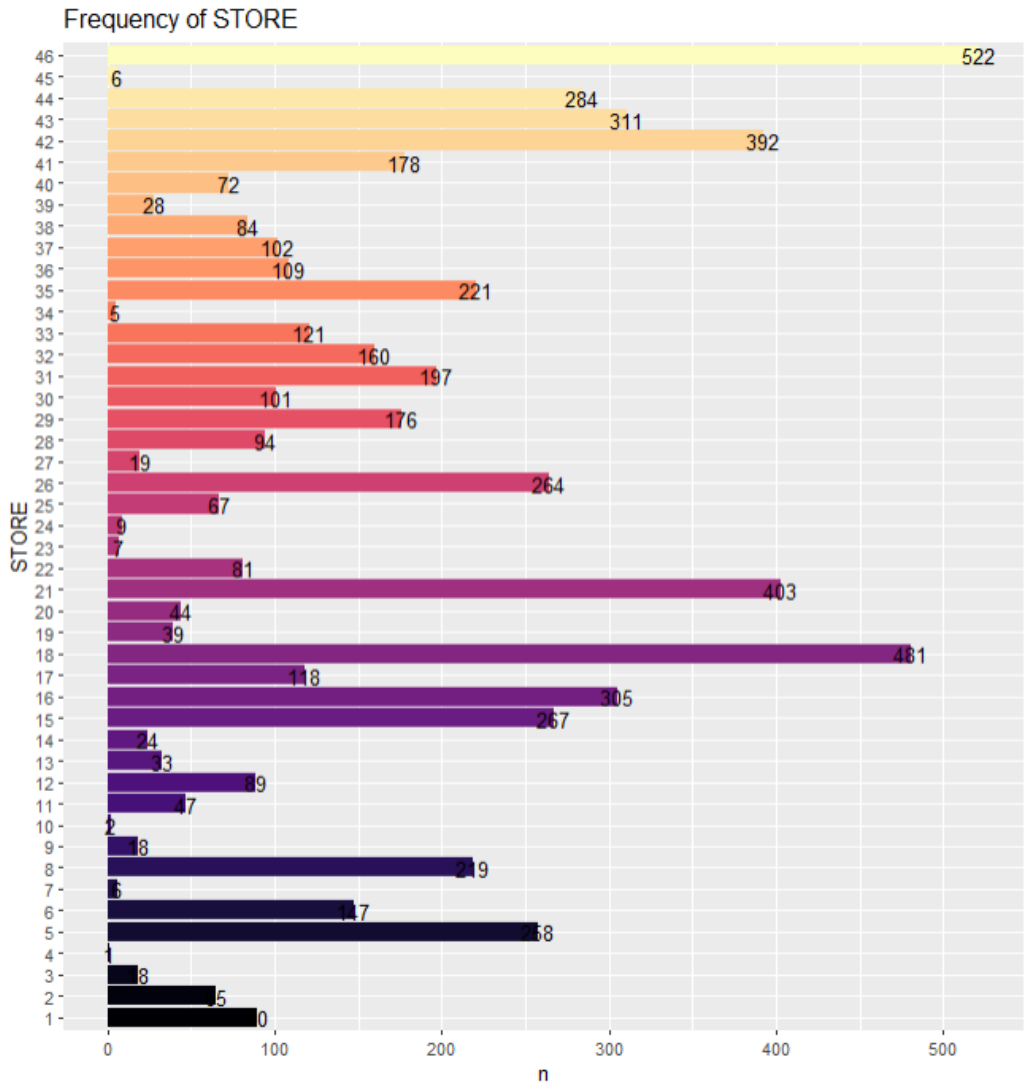
Code: Generating a Table of the Frequency of “STORE”

```
> exp1_store
 STORE    n percent
     1   90   1.43%
     2   65   1.03%
     3   18   0.29%
     4    1   0.02%
     5  258   4.11%
     6  147   2.34%
     7    6   0.10%
     8  219   3.49%
     9   18   0.29%
    10    2   0.03%
    11   47   0.75%
    12   89   1.42%
    13   33   0.53%
    14   24   0.38%
    15  267   4.25%
    16  305   4.85%
    17  118   1.88%
    18  481   7.65%
    19   39   0.62%
    20   44   0.70%
    21  403   6.41%
    22   81   1.29%
    23    7   0.11%
    24    9   0.14%
    25   67   1.07%
    26  264   4.20%
    27   19   0.30%
    28   94   1.50%
    29  176   2.80%
    30  101   1.61%
    31  197   3.13%
    32  160   2.55%
    33  121   1.93%
    34    5   0.08%
    35  221   3.52%
    36  109   1.73%
    37  102   1.62%
    38   84   1.34%
    39   28   0.45%
    40   72   1.15%
    41  178   2.83%
    42  392   6.24%
    43  311   4.95%
    44  284   4.52%
    45    6   0.10%
    46  522   8.31%
 Total 6284 100.00%
```

Output: Table of the Frequency of "STORE"

```
## CHART: Frequency of STORE
ggplot(exp_store, aes(x=n, y=STORE)) +
  geom_bar(stat="identity", fill=magma(nlevels(data$STORE))) +
  geom_text(aes(label=n)) +
  ggtitle("Frequency of STORE")
```

Code: Generating a Bar Graph of the Frequency of "STORE"



Graph: Bar Graph of the Frequency of "STORE"

The output reveal that most employees are working in Store 46 (522 employees (8.31%)), followed by Store 18 which has 481 employees (7.65%). The store that had the least amount of employees is Store 4, with only 1 employee working there (0.02%).

5.13 - “GENDER” Column

This column consists of the gender of the employees.

```
## Data summary
summary(data$GENDER)

## Unique data
unique(data$GENDER)

## Counting the amount of distinct values
nlevels(data$GENDER)
```

Code: Checking Summary, Unique data, and the Number of Distinct Values

```
> summary(data$GENDER)
Female   Male
  3278   3006
> unique(data$GENDER)
[1] Male   Female
Levels: Female Male
> nlevels(data$GENDER)
[1] 2
```

Output: Summary,Unique data, and the Number of Distinct Values of the “GENDER” Column

```
class(data$GENDER)
```

Code: Datatype of “GENDER” Column

```
> class(data$GENDER)
[1] "factor"
```

Output: Datatype of “GENDER” Column

As converted earlier, the datatype of the column is Factor.

```
## List of the frequency of each values
exp_gender <- tabyl(data, GENDER)

## TABLE: Frequency of GENDER
exp1_gender <- exp_gender %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE)
exp1_gender
```

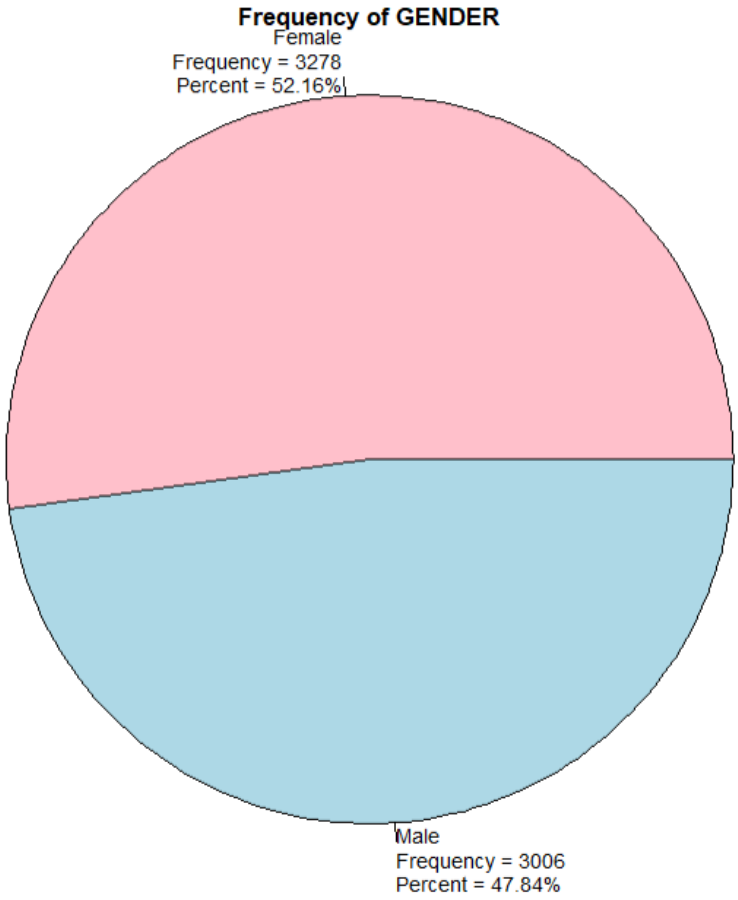
Code: Generating a Table of the Frequency of “GENDER”

```
> exp1_gender
 GENDER    n percent
 Female 3278  52.16%
   Male 3006  47.84%
  Total 6284 100.00%
```

Output: Table of the Frequency of “GENDER”

```r
## CHART: Frequency of GENDER
exp_gender_vis <- exp_gender %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
pie(exp_gender_vis$n,
    label=paste0(exp_gender_vis$GENDER,
                 "\nFrequency = ", exp_gender_vis$n,
                 "\nPercent = ", exp_gender_vis$percent),
    radius=1,
    main="Frequency of GENDER",
    col=c("pink", "light blue"))
```

Code: Generating a Pie Chart of the Frequency of "GENDER"



Graph: Pie Chart of the Frequency of "GENDER"

The output reveals that majority of employees are all female (3278 employees (52.16%)), while the rest being males (3006 employees (47.84%)).

5.14 - "TERMINATION_REASON" Column

This column consists of the termination reason of terminated employees. For active employees, their termination reason is shown as "Not Applicable".

```
## Data summary
summary(data$TERMINATION_REASON)

## Unique data
unique(data$TERMINATION_REASON)

## Counting the amount of distinct values
nlevels(data$TERMINATION_REASON)
```

Code: Checking Summary, Unique data, and the Number of Distinct Values

```
> summary(data$TERMINATION_REASON)
        Layoff Not Applicable    Resignation     Retirement
           215           4804            382            883
> unique(data$TERMINATION_REASON)
[1] Not Applicable Retirement     Resignation    Layoff
Levels: Layoff Not Applicable Resignation Retirement
> nlevels(data$TERMINATION_REASON)
[1] 4
```

Output: Summary,Unique data, and the Number of Distinct Values of the "TERMINATION_REASON" Column

From the output, we can see that there are 3 termination reasons excluding "Not Applicable". "Layoff" indicates that the employee was terminated involuntarily, while "Resignation" and "Retirement" indicates that the employee was terminated voluntarily.

```
class(data$TERMINATION_REASON)
```

Code: Datatype of "TERMINATION_REASON" Column

```
> class(data$TERMINATION_REASON)
[1] "factor"
```

Output: Datatype of "TERMINATION_REASON" Column

As converted earlier, the datatype of the column is Factor.

```
## List of the frequency of each values
exp_termreason <- tabyl(data, TERMINATION_REASON)

## TABLE: Frequency of TERMINATION_REASON
exp1_termreason <- exp_termreason %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
exp1_termreason
```

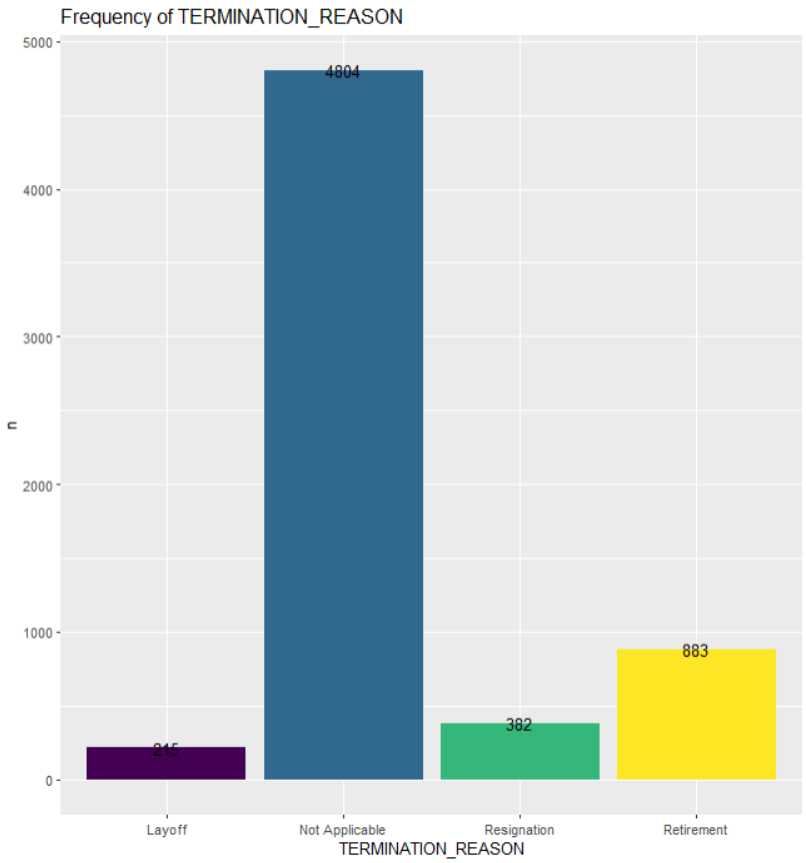
Code: Generating a Table of the Frequency of "TERMINATION_REASON"

```
> exp1_termreason
 TERMINATION_REASON    n percent
              Layoff  215   3.42%
     Not Applicable 4804  76.45%
        Resignation  382   6.08%
         Retirement  883  14.05%
              Total 6284 100.00%
```

Output: Table of the Frequency of "TERMINATION_REASON"

```
## CHART: Frequency of TERMINATION_REASON
ggplot(exp_termreason, aes(x=TERMINATION_REASON, y=n)) +
  geom_bar(stat="identity", fill=viridis(nlevels(data$TERMINATION_REASON))) +
  geom_text(aes(label=n)) +
  ggtitle("Frequency of TERMINATION_REASON")
```

Code: Generating a Bar Graph of the Frequency of "TERMINATION_REASON"



Graph: Bar Graph of the Frequency of "TERMINATION_REASON"

The results show that most employees are active employees (4804 employees (76.45%)), and the rest being terminated employees. When compared to active employees, laid-off employees made up 3.42% of the chart, with resignation accounting for 6.08% and retirement accounting for 14.05%.

```r
## TABLE: Frequency of TERMINATION_REASON (Excluding NA)
exp2_termreason <- filter(exp_termreason, TERMINATION_REASON!="Not Applicable") %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
exp2_termreason
```

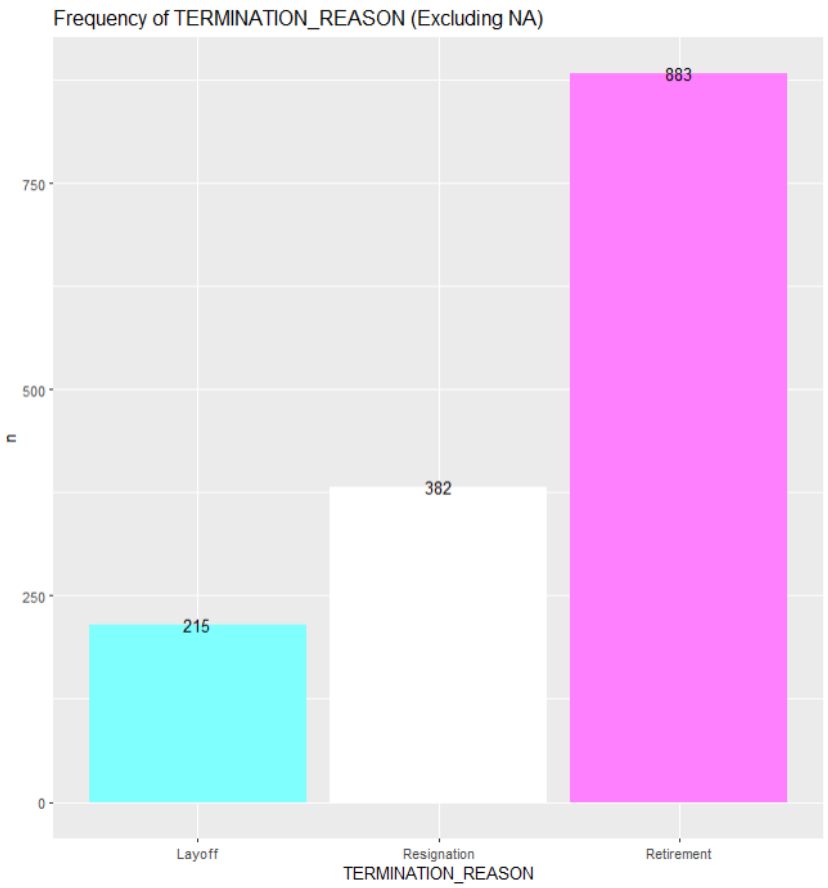
Code: Generating a Table of the Frequency of "TERMINATION_REASON" Excluding "Not Applicable" Values

```
> exp2_termreason
 TERMINATION_REASON    n percent
             Layoff  215   3.42%
         Resignation  382   6.08%
          Retirement  883  14.05%
               Total 1480  23.55%
```

Output: Table of the Frequency of "TERMINATION_REASON" Excluding "Not Applicable" Values

```r
## CHART: Frequency of TERMINATION_REASON (Excluding NA)
ggplot(filter(exp_termreason, TERMINATION_REASON!="Not Applicable"),
               aes(x=TERMINATION_REASON, y=n)) +
  geom_bar(stat="identity", fill=cm.colors(3)) +
  geom_text(aes(label=n)) +
  ggtitle("Frequency of TERMINATION_REASON (Excluding NA)")
```

Code: Generating a Bar Graph of the Frequency of "TERMINATION_REASON" Excluding "Not Applicable" Values



Graph: Bar Graph of the Frequency of "TERMINATION_REASON" Excluding "Not Applicable" Values

The output reveal that 1480 employees (23.55%) were terminated overall. Most employees were terminated through retirement (883 employees), follow by resignation (382 employees), then layoff (215 employees).

5.15 - "TERMINATION_TYPE" Column

This column consists of the of the termination type of terminated employees. For active employees, their termination type is shown as "Not Applicable".

```
## Data summary
summary(data$TERMINATION_TYPE)

## Unique data
unique(data$TERMINATION_TYPE)

## Counting the amount of distinct values
nlevels(data$TERMINATION_TYPE)
```

Code: Checking Summary, Unique data, and the Number of Distinct Values

```
> summary(data$TERMINATION_TYPE)
   Involuntary Not Applicable      Voluntary
           215           4804           1265
> unique(data$TERMINATION_TYPE)
[1] Not Applicable Voluntary      Involuntary
Levels: Involuntary Not Applicable Voluntary
> nlevels(data$TERMINATION_TYPE)
[1] 3
```

Output: Summary,Unique data, and the Number of Distinct Values of the "TERMINATION_TYPE" Column

From the output, we can see that there are 2 termination types excluding "Not Applicable". For active employees, their termination reason is shown as "Not Applicable".

```
class(data$TERMINATION_TYPE)
```

Code: Datatype of "TERMINATION_TYPE" Column

```
> class(data$TERMINATION_TYPE)
[1] "factor"
```

Output: Datatype of "TERMINATION_TYPE" Column

As converted earlier, the datatype of the column is Factor.

```
## List of the frequency of each values
exp_termtype <- tabyl(data, TERMINATION_TYPE)

## TABLE: Frequency of TERMINATION_TYPE
exp1_termtype <- exp_termtype %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
exp1_termtype
```

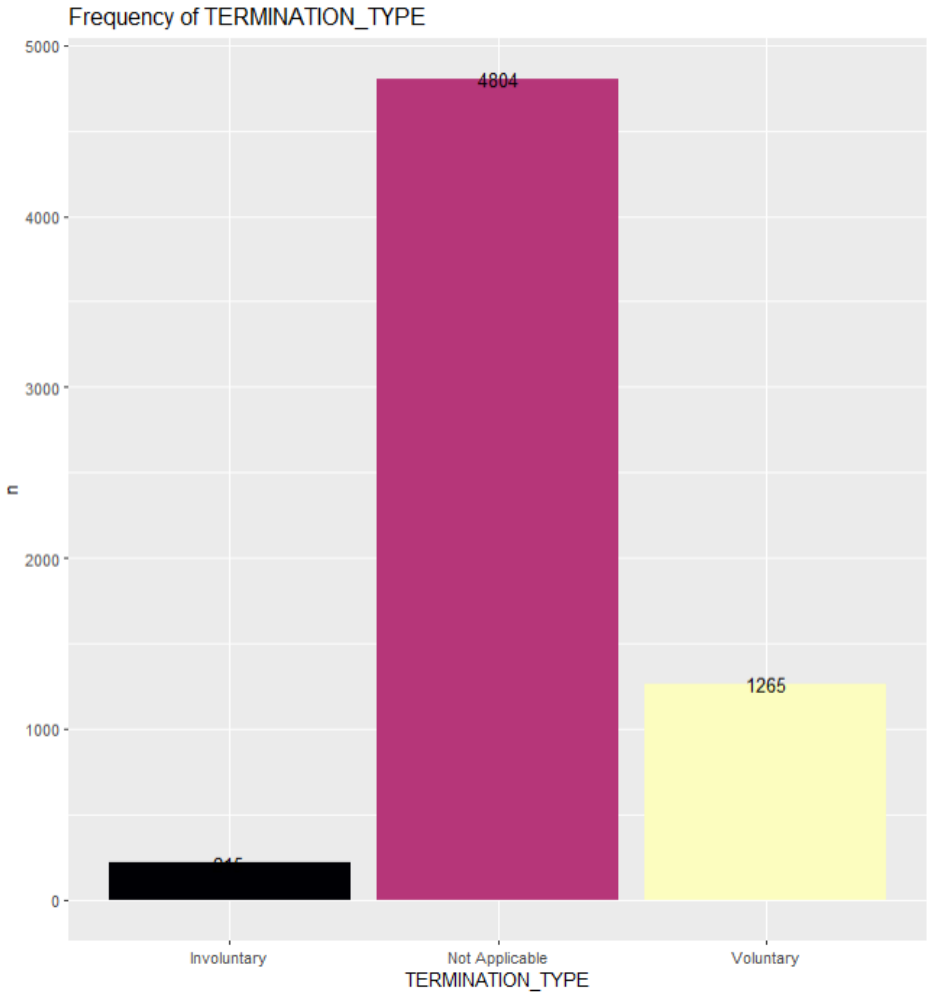
Code: Generating a Table of the Frequency of "TERMINATION_TYPE"

```
> exp1_termtype
 TERMINATION_TYPE    n percent
      Involuntary  215   3.42%
   Not Applicable 4804  76.45%
        Voluntary 1265  20.13%
            Total 6284 100.00%
```

Output: Table of the Frequency of "TERMINATION_TYPE"

```
## CHART: Frequency of TERMINATION_TYPE
ggplot(exp_termtype, aes(x=TERMINATION_TYPE, y=n)) +
  geom_bar(stat="identity", fill=magma(nlevels(data$TERMINATION_TYPE))) +
  geom_text(aes(label=n)) +
  ggtitle("Frequency of TERMINATION_TYPE")
```

Code: Generating a Bar Graph of the Frequency of "TERMINATION_TYPE"



Graph: Bar Graph of the Frequency of "TERMINATION_TYPE"

The results show that most employees are active employees (4804 employees (76.45%)), and the rest being terminated employees. When compared to active employees, employees that left voluntarily made up 20.13% of the chart, with employees that left involuntarily accounting for 3.42%.

```
## TABLE: Frequency of TERMINATION_TYPE (Excluding NA)
exp2_termtype <- filter(exp_termtype, TERMINATION_TYPE!="Not Applicable") %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
exp2_termtype
```

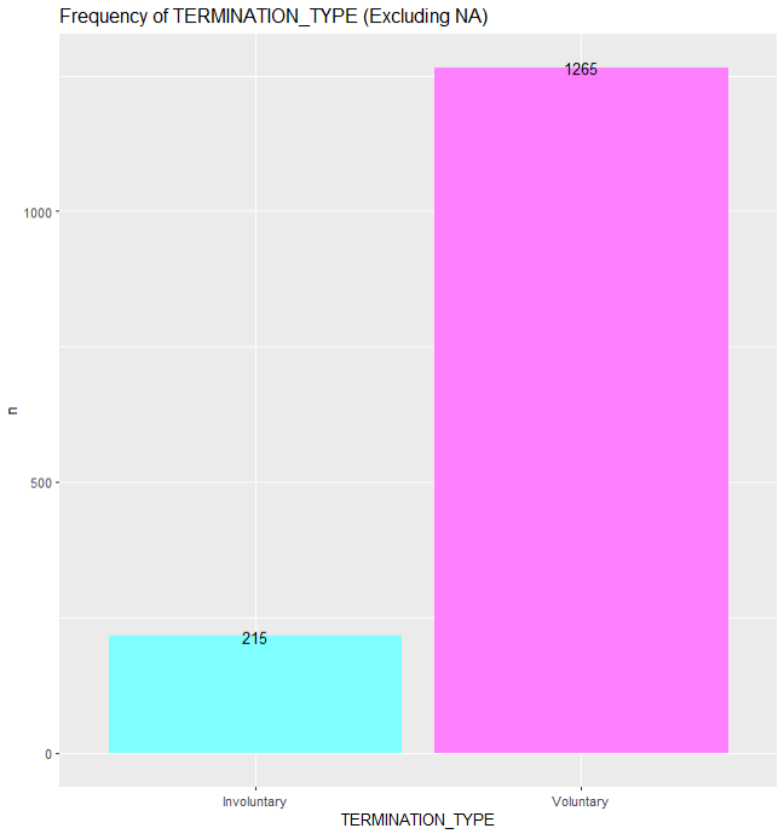
Code: Generating a Table of the Frequency of "TERMINATION_TYPE" Excluding "Not Applicable" Values

```
> exp2_termtype
 TERMINATION_TYPE    n percent
      Involuntary  215   3.42%
        Voluntary 1265  20.13%
            Total 1480  23.55%
```

Output: Table of the Frequency of "TERMINATION_TYPE" Excluding "Not Applicable" Values

```
## CHART: Frequency of TERMINATION_TYPE (Excluding NA)
ggplot(filter(exp_termtype, TERMINATION_TYPE!="Not Applicable"),
       aes(x=TERMINATION_TYPE, y=n)) +
  geom_bar(stat="identity", fill=cm.colors(2)) +
  geom_text(aes(label=n)) +
  ggtitle("Frequency of TERMINATION_TYPE (Excluding NA)")
```

Code: Generating a Bar Graph of the Frequency of "TERMINATION_TYPE" Excluding "Not Applicable" Values



Graph: Bar Graph of the Frequency of "TERMINATION_TYPE" Excluding "Not Applicable" Values

The output reveal that 1480 employees (23.55%) were terminated overall. Most employees were terminated voluntarily (1265 employees), followed by involuntarily (215 employees).

5.16 - “RECORD_YEAR” Column

This column indicates the year on which the employees' records were created.

```
## Unique data
unique(data$RECORD_YEAR)

## Counting the amount of distinct values
n_distinct(data$RECORD_YEAR)

## Determining the datatype
class(data$RECORD_YEAR)
```

Code: Checking Unique data, the Number of Distinct Values and Datatype

```
> unique(data$RECORD_YEAR)
 [1] 2015 2009 2014 2012 2007 2010 2011 2006 2008 2013
> n_distinct(data$RECORD_YEAR)
[1] 10
> class(data$RECORD_YEAR)
[1] "integer"
```

Output: Datatype of “RECORD_YEAR” Column

The unique data and the amount of distinct values are checked for the “RECORD_YEAR” Column. This code would only need to be executed once because the unique information for the "data" and "data_full" datasets would be the same. We can also observe that there are 10 unique years in the “RECORD_YEAR” Column. The datatype of this column is Integer.

5.16.1 - In “data_full” dataset

```
summary(data_full$RECORD_YEAR)
```

Code: Summary of the “RECORD_YEAR” Column

```
> summary(data_full$RECORD_YEAR)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2006    2008    2011    2011    2013    2015
```

Output: Summary of the “RECORD_YEAR” Column

As we can see, the data is recorded from year 2006 to 2015.

```
## List of the frequency of each year
exp_record_year <- tabyl(data_full, RECORD_YEAR)

## TABLE: Frequency of RECORD_YEAR
exp1_record_year <- exp_record_year %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE)
exp1_record_year
```

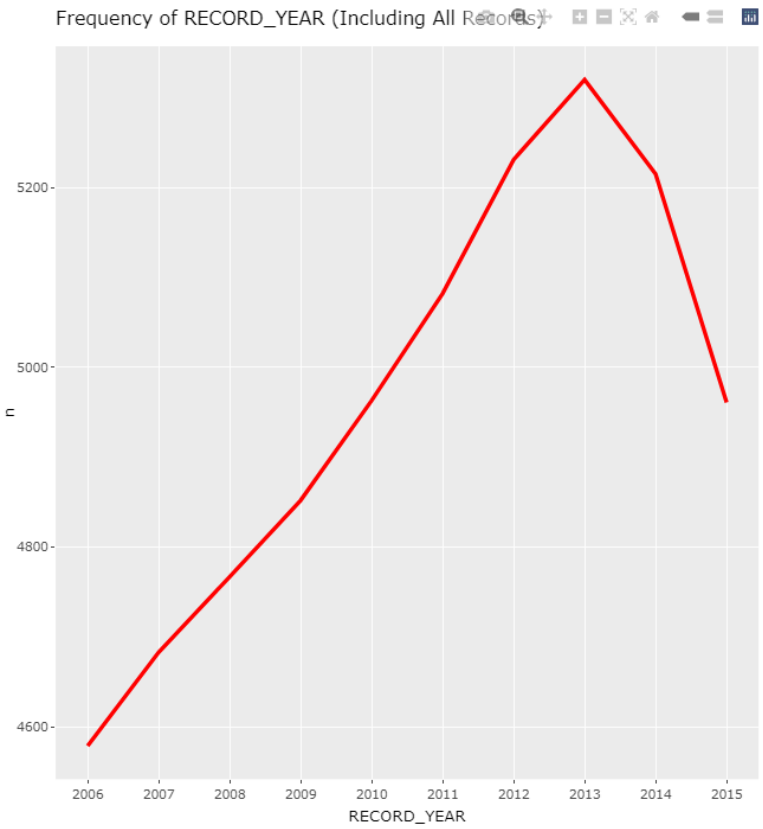
Code: Generating a Table of the Frequency of "RECORD_YEAR"

```
> exp1_record_year
 RECORD_YEAR     n percent
        2006  4579   9.22%
        2007  4683   9.43%
        2008  4767   9.60%
        2009  4852   9.77%
        2010  4963  10.00%
        2011  5082  10.24%
        2012  5231  10.54%
        2013  5320  10.71%
        2014  5215  10.50%
        2015  4961   9.99%
       Total 49653 100.00%
```

Output: Table of the Frequency of "RECORD_YEAR"

```
## CHART: Frequency of RECORD_YEAR
exp_record_year_vis <- ggplot(exp_record_year, aes(x=RECORD_YEAR, y=n)) +
  geom_line(col="red", size=1) +
  ggtitle("Frequency of RECORD_YEAR (Including All Records)") +
  scale_x_continuous(breaks=unique(data_full$RECORD_YEAR))
ggplotly(exp_record_year_vis)
```

Code: Generating a Line Graph of the Frequency of "RECORD_YEAR"



Graph: Line Graph of the Frequency of "RECORD_YEAR"

In the data, we can see that the most data recorded per year was 5320 (10.71%) in year 2013, while the least data recorded was 4579 (9.22%) in year 2006. The frequency of recorded year was gradually increasing from year 2006 to 2013, but it dropped drastically in year 2014 and 2015. The decrease in records per year indicates that the number of employees has decreased as well.

5.16.2 - In "data" dataset

```
summary(data$RECORD_YEAR)
```

Code: Summary of the "RECORD_YEAR" Column

```
> summary(data$RECORD_YEAR)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2006    2015    2015    2014    2015    2015
```

Output: Summary of the "RECORD_YEAR" Column

As we can see, the data is recorded from year 2006 to 2015. This is the same with the "data_full" dataset, with only slight changes in the first quartile, median, mean and 3rd quartile since the latest records are done in the latest year which is 2015.

```
## List of the frequency of each year
exp_record_year <- tabyl(data, RECORD_YEAR)

## TABLE: Frequency of RECORD_YEAR
exp2_record_year <- exp_record_year %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE)
exp2_record_year
```

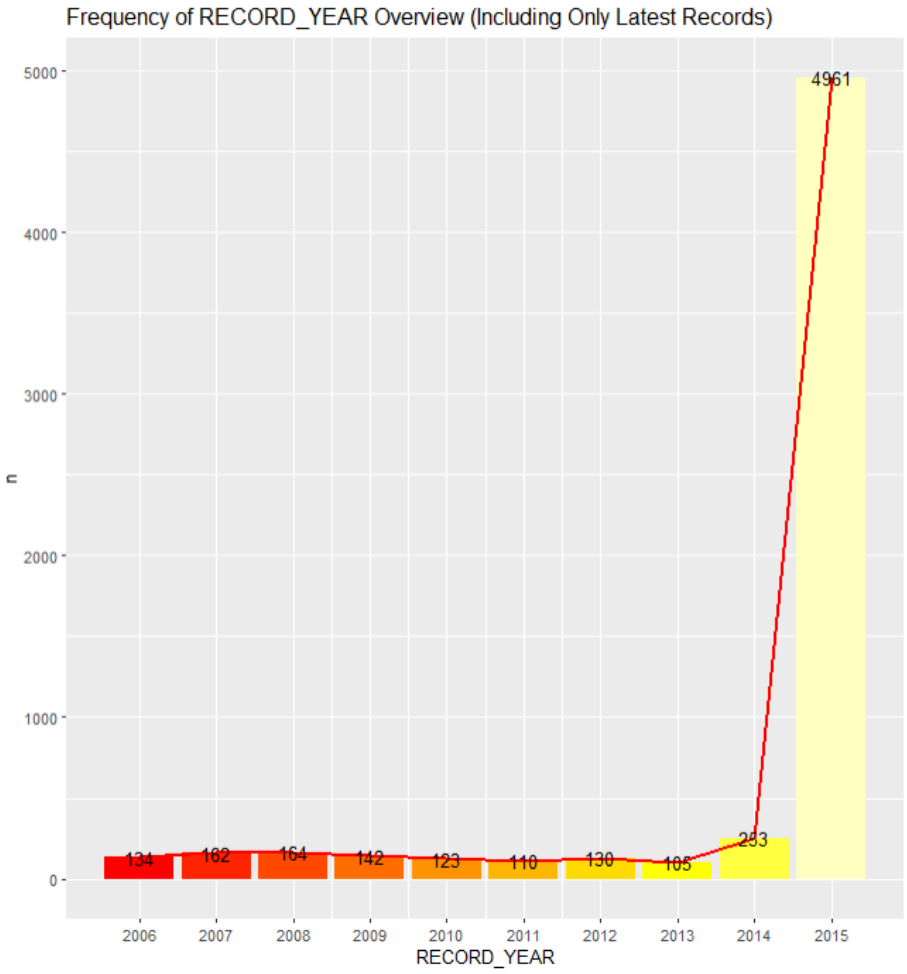
Code: Generating a Table of the Frequency of "RECORD_YEAR"

```
> exp2_record_year
 RECORD_YEAR    n percent
        2006  134   2.13%
        2007  162   2.58%
        2008  164   2.61%
        2009  142   2.26%
        2010  123   1.96%
        2011  110   1.75%
        2012  130   2.07%
        2013  105   1.67%
        2014  253   4.03%
        2015 4961  78.95%
       Total 6284 100.00%
```

Output: Table of the Frequency of "RECORD_YEAR"

```r
## CHART: Frequency of RECORD_YEAR
exp_record_year_vis2 <- exp_record_year
ggplot(exp_record_year_vis2, aes(x=RECORD_YEAR, y=n)) +
  geom_bar(stat="identity",
           fill=heat.colors(n_distinct(exp_record_year))) +
  geom_line(col="red", size=1) +
  geom_text(aes(label=n)) +
  ggtitle("Frequency of RECORD_YEAR Overview (Including Only Latest Records)") +
  scale_x_continuous(breaks=unique(data$RECORD_YEAR))
```

Code: Generating a Bar and Line Graph of the Frequency of "RECORD_YEAR"



Graph: Bar and Line Graph of the Frequency of "RECORD_YEAR"

In the data, we can see that the most data recorded per year was 4961 (78.95%) in year 2015, while the least data recorded was 105 (1.67%) in year 2013. The amount of records in 2015 increased drastically due to the fact that the latest records are done in the latest year.

5.17 - "STATUS" Column

This column consists of the status of the employees.

```
## Data summary
summary(data$STATUS)

## Unique data
unique(data$STATUS)

## Counting the amount of distinct values
nlevels(data$STATUS)
```

Code: Checking Summary, Unique data, and the Number of Distinct Values

```
> summary(data$STATUS)
    ACTIVE TERMINATED
      4804       1480
> unique(data$STATUS)
[1] ACTIVE     TERMINATED
Levels: ACTIVE TERMINATED
> nlevels(data$STATUS)
[1] 2
```

Output: Summary,Unique data, and the Number of Distinct Values of the "STATUS" Column

From the output, we can see that there are 2 different status. "Active" indicates that the employee is still working in the company, while "Terminated" indicates that the employee left the company.

```
class(data$STATUS)
```

Code: Datatype of "STATUS" Column

```
> class(data$STATUS)
[1] "factor"
```

Output: Datatype of "STATUS" Column

As converted earlier, the datatype of the column is Factor.

```
## List of the frequency of each values
exp_status <- tabyl(data, STATUS)

## TABLE: Frequency of STATUS
exp1_status <- exp_status %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE)
exp1_status
```

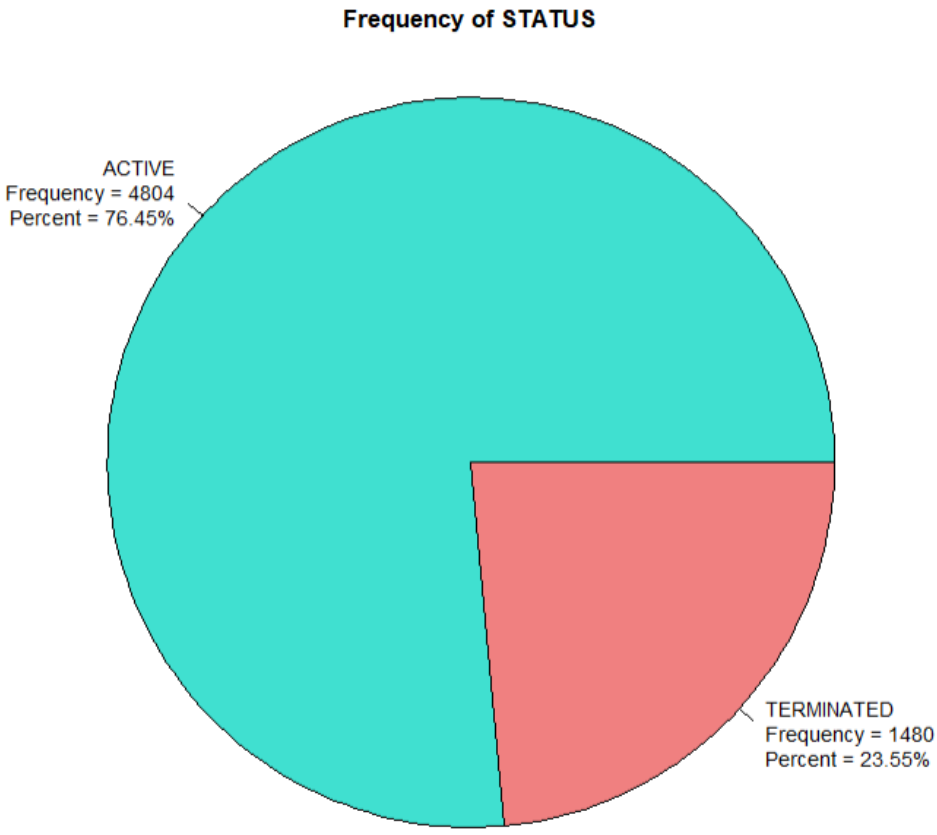
Code: Generating a Table of the Frequency of "STATUS"

```
> exp1_status
     STATUS    n percent
     ACTIVE 4804  76.45%
 TERMINATED 1480  23.55%
      Total 6284 100.00%
```

Output: Table of the Frequency of "STATUS"

```
## CHART: Frequency of STATUS
exp_status_vis <- exp_status %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
pie(exp_status_vis$n,
    label=paste0(exp_status_vis$STATUS,
                 "\nFrequency = ", exp_status_vis$n,
                 "\nPercent = ", exp_status_vis$percent),
    radius=1,
    main="Frequency of STATUS",
    col=c("turquoise", "lightcoral"))
```

Code: Generating a Pie Chart of the Frequency of "STATUS"



Graph: Pie Chart of the Frequency of "STATUS"

The output reveals that the majority of employees are active (4804 employees (76.45%)), while the rest was terminated (1480 employees (23.55%)).

5.18 - "BUSINESS_UNIT" Column

This column consists of the business unit of the employees.

```
## Data summary
summary(data$BUSINESS_UNIT)

## Unique data
unique(data$BUSINESS_UNIT)

## Counting the amount of distinct values
nlevels(data$BUSINESS_UNIT)
```

Code: Checking Summary, Unique data, and the Number of Distinct Values

```
> summary(data$BUSINESS_UNIT)
HEADOFFICE     STORES
        80       6204
> unique(data$BUSINESS_UNIT)
[1] HEADOFFICE STORES
Levels: HEADOFFICE STORES
> nlevels(data$BUSINESS_UNIT)
[1] 2
```

Output: Summary,Unique data, and the Number of Distinct Values of the "BUSINESS_UNIT" Column

From the output, we can see that there are 2 different business units, one is the Head Office, and another are Stores.

```
class(data$BUSINESS_UNIT)
```

Code: Datatype of "BUSINESS_UNIT" Column

```
> class(data$BUSINESS_UNIT)
[1] "factor"
```

Output: Datatype of "BUSINESS_UNIT" Column

As converted earlier, the datatype of the column is Factor.

```
## List of the frequency of each values
exp_business_unit <- tabyl(data, BUSINESS_UNIT)

## TABLE: Frequency of BUSINESS_UNIT
exp1_business_unit <- exp_business_unit %>%
  adorn_totals() %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE)
exp1_business_unit
```

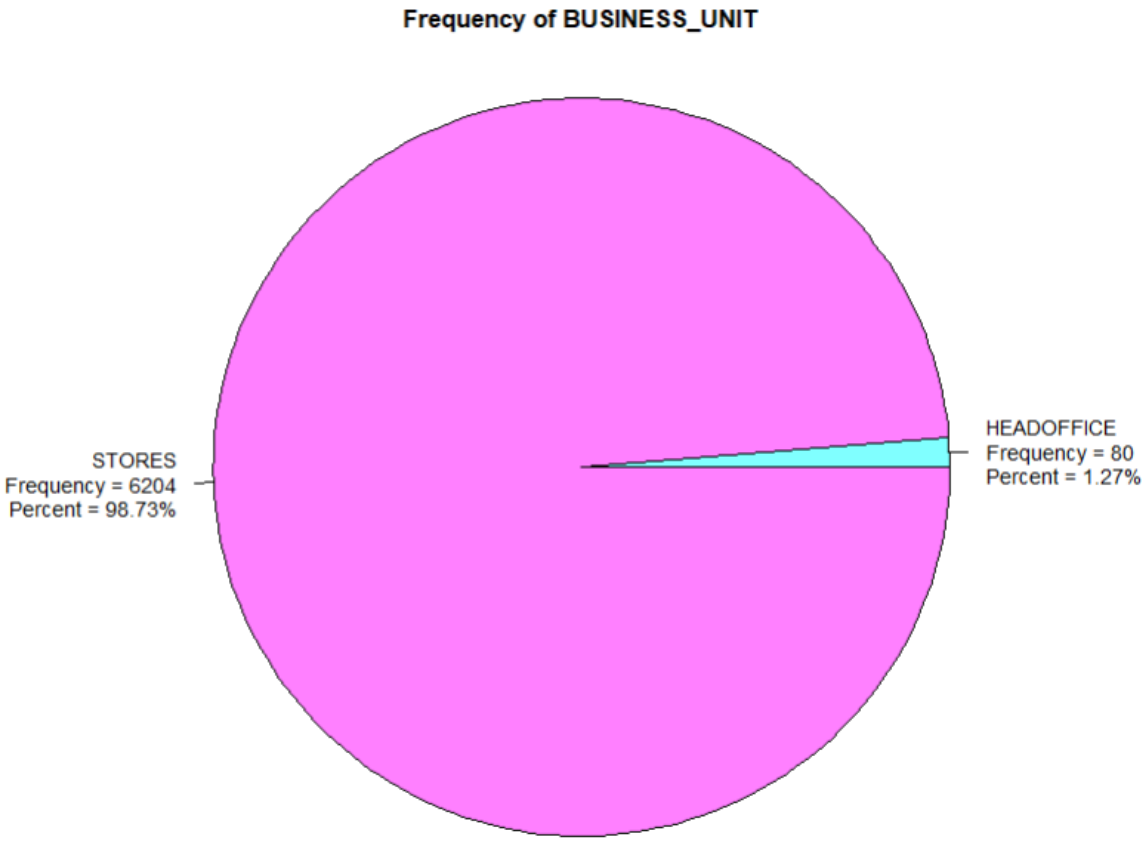
Code: Generating a Table of the Frequency of "BUSINESS_UNIT"

```
> exp1_business_unit
 BUSINESS_UNIT    n percent
    HEADOFFICE   80   1.27%
        STORES 6204  98.73%
         Total 6284 100.00%
```

Output: Table of the Frequency of "BUSINESS_UNIT"

```r
## CHART: Frequency of BUSINESS_UNIT
exp_business_unit_vis <- exp_business_unit %>%
  adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
pie(exp_business_unit_vis$n,
    label=paste0(exp_business_unit_vis$BUSINESS_UNIT,
                 "\nFrequency = ", exp_business_unit_vis$n,
                 "\nPercent = ", exp_business_unit_vis$percent),
    radius=1,
    main="Frequency of BUSINESS_UNIT",
    col=cm.colors(2))
```

Code: Generating a Pie Chart of the Frequency of "BUSINESS_UNIT"



Graph: Pie Chart of the Frequency of "BUSINESS_UNIT"

The output reveal that majority of employees works at the stores (6204 employees (98.73%)), while the rest works at the head office (80 employees (1.27%)).

6.0 QUESTION AND ANALYSIS

6.1 - Question 1: What is the Company's Layout?

This question is to find out the company's layout in terms of its head office, the distribution of the stores, and certain characteristics of the employees.

6.1.1 - Question 1.1 - Which Store is the Head Office?

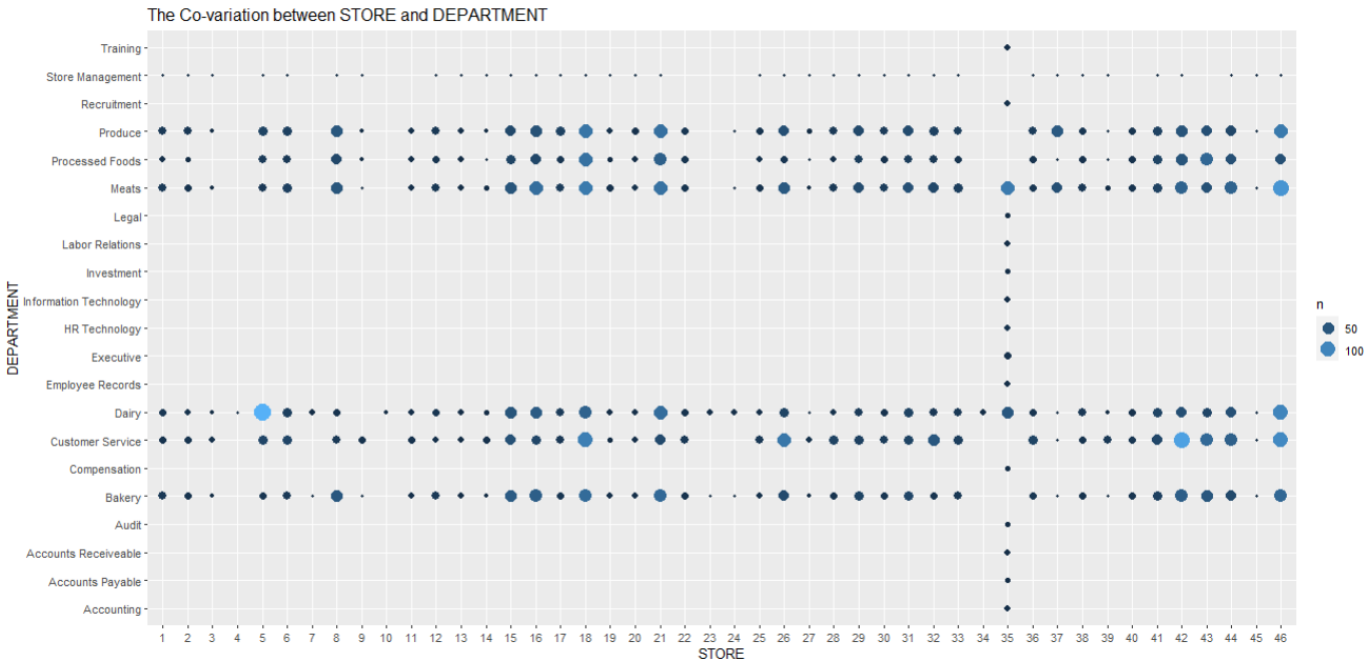
This question is to find out which store is the Head Office located in.

Analysis 6.1.1.1 - Check if certain stores are focused on certain departments

This analysis is to check if certain stores are focused on certain departments to locate the Head Office.

```
ggplot(data, aes(x=STORE, y=DEPARTMENT)) +
  geom_count(aes(color = ..n.., size = ..n..)) +
  guides(color = 'legend') +
  labs(title="The Co-variation between STORE and DEPARTMENT", x="STORE", y="DEPARTMENT")
```

Code: Count Plot for the Co-variation between STORE and DEPARTMENT



Graph: Count Plot for the Co-variation between STORE and DEPARTMENT

The results indicates the Head Office is very likely to be located in Store 35. This is because it contains employees in the Training, Recruitment, Legal, Labor Relations, Investment, Information Technology, HR Technology, Executive, Employee Records, Compensation, Audit, Accounts Receivable, Accounts

Payable and Acoounting department that other stores do not have. Moreover, Store 35 does not have a Customer Service department. From the graph, we could also see that the only department the headquarters have in common with other stores are the Meats and Dairy department, which leads to a possibility that Store 35 consists of both Head Office and store. Other stores are quite balanced as most of them have Produce, Processed Foods, Meats, Dairy, Customer Service and Bakery departments.

Analysis 6.1.1.2 - Comparing Store 35 with BUSINESS_UNIT data

This analysis is to confirm if the Head Office is really Store 35 by comparing it with the BUSINESS_UNIT data we have.

```
subset(exp_business_unit, BUSINESS_UNIT=="HEADOFFICE", select=c(BUSINESS_UNIT,n))
subset(exp_store, STORE=="35", select=c(STORE,n))
```

Code: Comparing Store 35 with BUSINESS_UNIT data

```
> subset(exp_business_unit, BUSINESS_UNIT=="HEADOFFICE", select=c(BUSINESS_UNIT,n))
 BUSINESS_UNIT  n
    HEADOFFICE 80
> subset(exp_store, STORE=="35", select=c(STORE,n))
 STORE   n
    35 221
```

Output: Comparing Store 35 with BUSINESS_UNIT data

Since we can see that the Head Office only have 80 employees while Store 35 itself has 221 employees, we can deduce that in Store 35, not all departments belongs to the Head Office.

Analysis 6.1.1.3 - Departments that the Head Office consists of

Since we know that the only department the headquarters have in common with other stores are the Meats and Dairy department, this analysis is to find the number of employees in Store 35 that does not work in the Meats and Dairy department.

```
store35_HO = nrow(filter(data, STORE=="35" & DEPARTMENT!=("Meats") & DEPARTMENT!=("Dairy")))
store35_HO
```

Code: Departments that the Head Office consists of

```
> store35_HO = nrow(filter(data, STORE=="35" & DEPARTMENT!=("Meats") & DEPARTMENT!=("Dairy")))
> store35_HO
[1] 80
```

Output: Departments that the Head Office consists of

The number of employees in Store 35 that does not work in the Meats and Dairy department matches the number of employees that work in the Head Office from the BUSINESS_UNIT data. Hence, we can deduce that Store 35 consists of both Head Office and store.

Analysis 6.1.1.4 - Employees that work in Head Office and Stores Departments of Store 35

This analysis is to calculate the number of employees that work in Head Office and Stores Departments of Store 35.

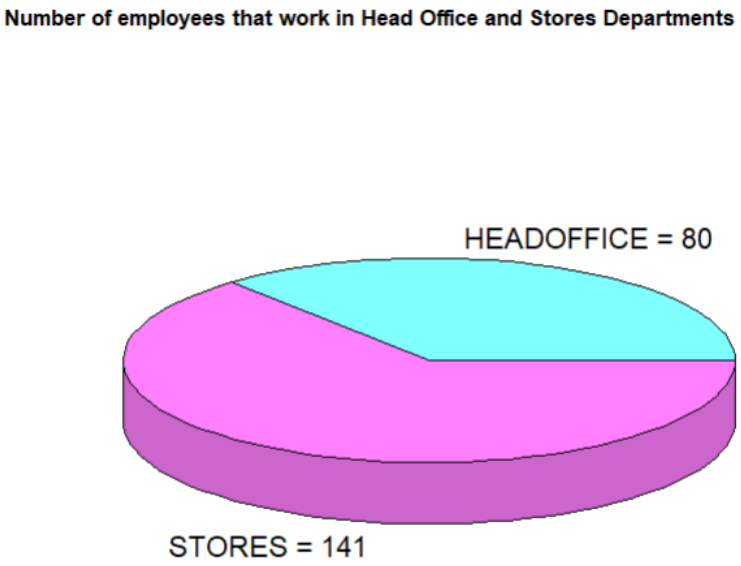
```
store35_ST = nrow(filter(data, STORE=="35" & (DEPARTMENT==("Meats") | DEPARTMENT==("Dairy"))))
store35_ST

pie3D(c(store35_HO,store35_ST),
      labels=c(paste0("HEADOFFICE = ",store35_HO),paste0("STORES = ",store35_ST)),
      col=cm.colors(2),
      main="Number of Employees that Work in Head Office and Stores Departments of Store 35")
```

Code: Employees that work in Head Office and Stores Departments of Store 35

```
> store35_ST = nrow(filter(data, STORE=="35" & (DEPARTMENT==("Meats") | DEPARTMENT==("Dairy"))))
> store35_ST
[1] 141
```

Output: Employees that work in Head Office and Stores Departments of Store 35



Graph: Pie Chart for Number of Employees that Work in Head Office and Stores Departments of Store 35

From the output, we can see that 80 employees works in the Head Office, and 141 employees works in the store for Store 35. This aligns with our findings from Section 5.12, which was the data exploration for the STORE column that stated that Store 35 had a total of 221 employees.

Analysis 6.1.1.5 - Location of the Head Office (Store 35)

This analysis is to find where Store 35 is located.

```
unique(subset(data, STORE=="35", select=CITY))
```

Code: Location of the Head Office (Store 35)

```
> unique(subset(data, STORE=="35", select=CITY))
       CITY
1 Vancouver
```

Output: Location of the Head Office (Store 35)

From the output, we could see that Store 35 is located in Vancouver.

Conclusion 1.1

Store 35 is located in Vancouver, and it consists of both the Head Office and Store, with a total of 221 employees. The Head Office contains a total of 80 employees in the Training, Recruitment, Legal, Labor Relations, Investment, Information Technology, HR Technology, Executive, Employee Records, Compensation, Audit, Accounts Receivable, Accounts Payable and Accounting department (15 departments) that other stores do not have. Moreover, Store 35 does not have a Customer Service department. The store section of Store 35 consists of the Meats department and Dairy department (2 departments), which has a total of 141 employees. Other stores are quite balanced as most of them have Produce, Processed Foods, Meats, Dairy, Customer Service and Bakery departments.

6.1.2 - Question 1.2 - How are the Stores distributed geographically by City?

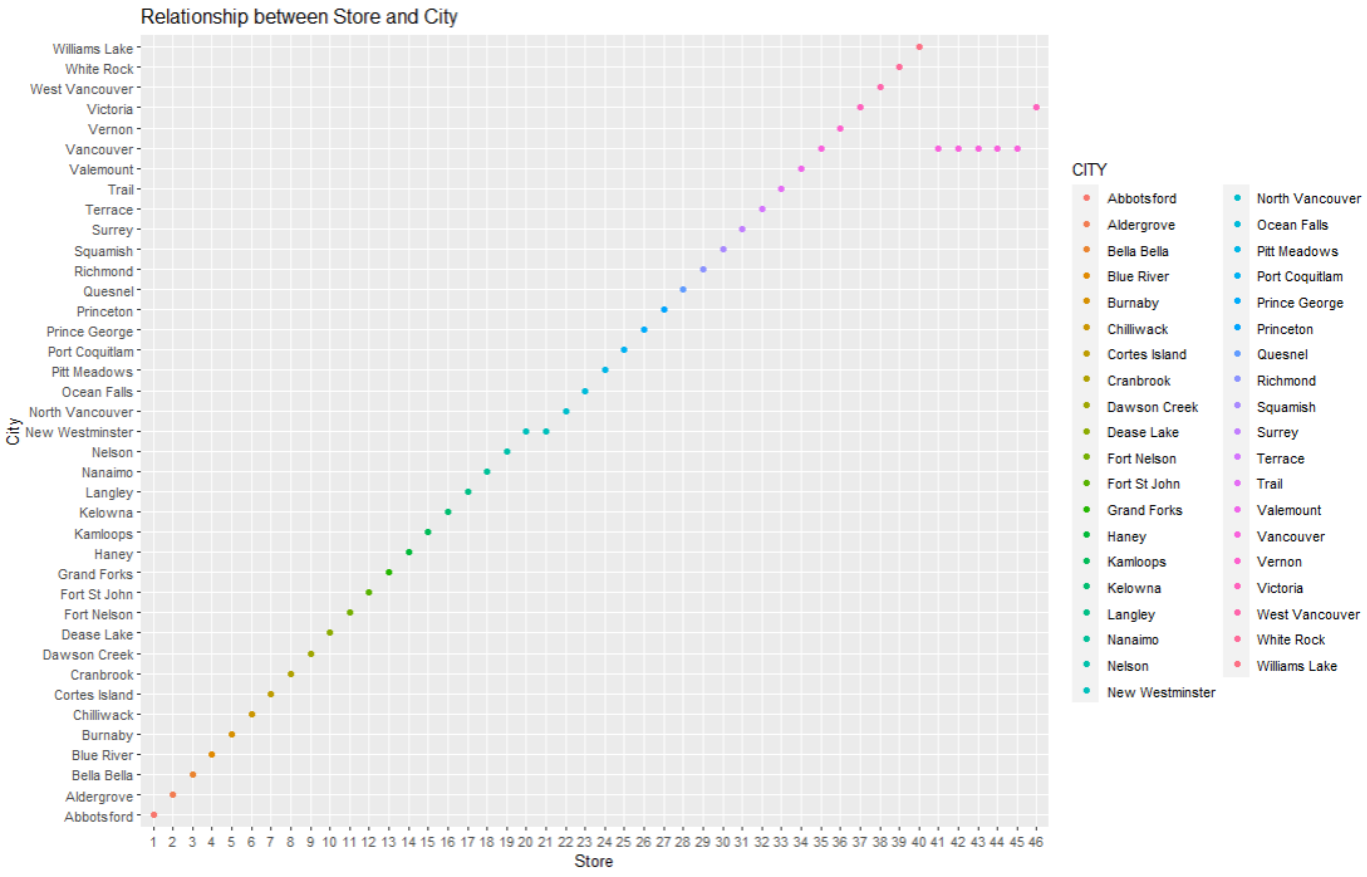
This question is to find out how are the stores distributed geographically by city.

Analysis 6.1.2.1 - Relationship between Store and City

This analysis is to find out the relationship between STORE and CITY.

```
data %>% select(STORE, CITY) %>%
  ggplot(aes(x=STORE, y=CITY, colour=CITY)) +
  geom_point() +
  labs(colour="CITY", x="Store", y="City", title="Relationship between Store and City")
```

Code: Generating Point Graph for Relationship between Store and City



Graph: Point Graph for Relationship between Store and City

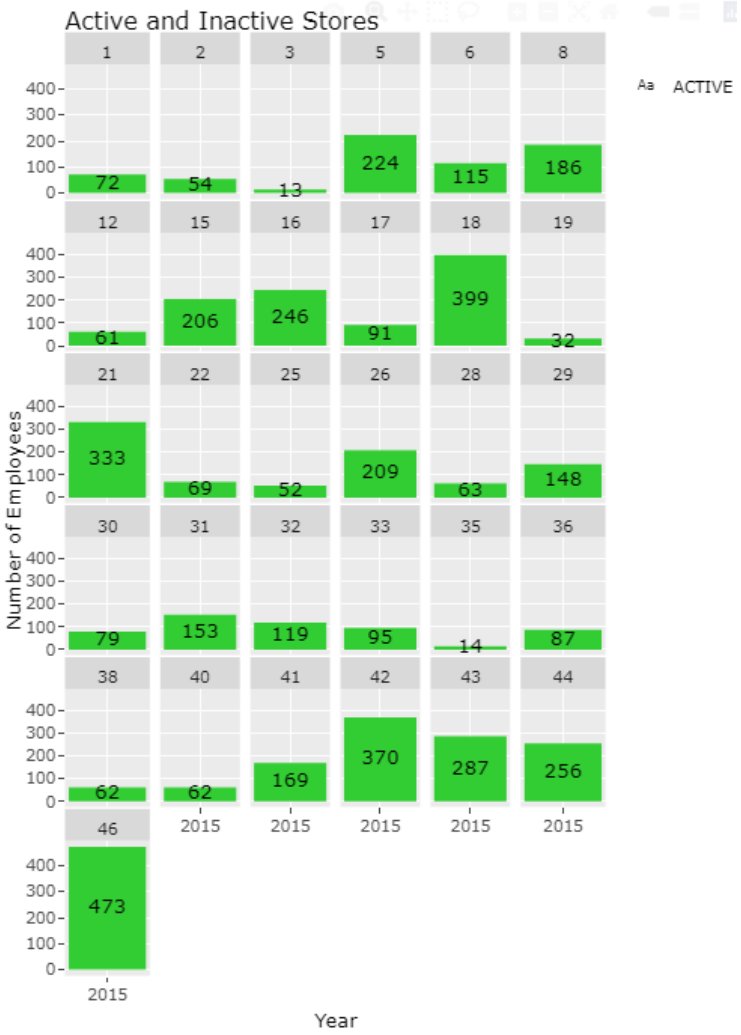
From the graph, we could see that most cities has only 1 store, but there are exceptions. Both New Westminster and Victoria has 2 stores, which is Store 20, 21 and Store 37, 46 respectively. Vancouver however, has the most number of stores which is 6 stores, namely Store 35, Store 41, Store 42, Store 43, Store 44, and Store 45. I believe that this is the reason why the Head Office is located in Vancouver, due to the amount of stores in the city alone. We could also see a pattern that the numbering of the store names corresponds to the alphabetical order of the city.

Analysis 6.1.2.2 - Active and Inactive Stores

This analysis is to find out if every store is currently active.

```
active_store <- data_full %>%
  select(RECORD_YEAR, STORE, STATUS) %>%
  filter(STATUS=="ACTIVE" & RECORD_YEAR==2015)
ggplotly(ggplot(active_store, aes(x=RECORD_YEAR, fill=STATUS)) +
  geom_bar(fill="limegreen") +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(x="Year", y="Number of Employees", title="Active and Inactive Stores") +
  scale_x_continuous(breaks=unique(data$RECORD_YEAR)) +
  facet_wrap(. ~ STORE))
```

Code: Checking for Active and Inactive Stores



Graph: Bar Grpah to Check for Active and Inactive Stores

Since every store has at least one active employee in 2015, it is confirmed that every store is active.

Conclusion 1.2

Most cities has only 1 store, but there are exceptional cases. Both New Westminster and Victoria has 2 stores, which is Store 20, 21 and Store 37, 46 respectively. Vancouver on the other hand, has the most number of stores which is 6 stores, namely Store 35, Store 41, Store 42, Store 43, Store 44, and Store 45. This is also the reason why the Head Office is located in Vancouver, given the number of stores in the city alone. Furthermore, the numbering of the store names corresponds to the alphabetical order of the city.

6.1.3 - Question 1.3 - Does the Age and Length of Service Correlates in this Company?

This question is to find out if the age and length of service correlates in the company.

Analysis 6.1.3.1 - Correlation between Age and Length of Service

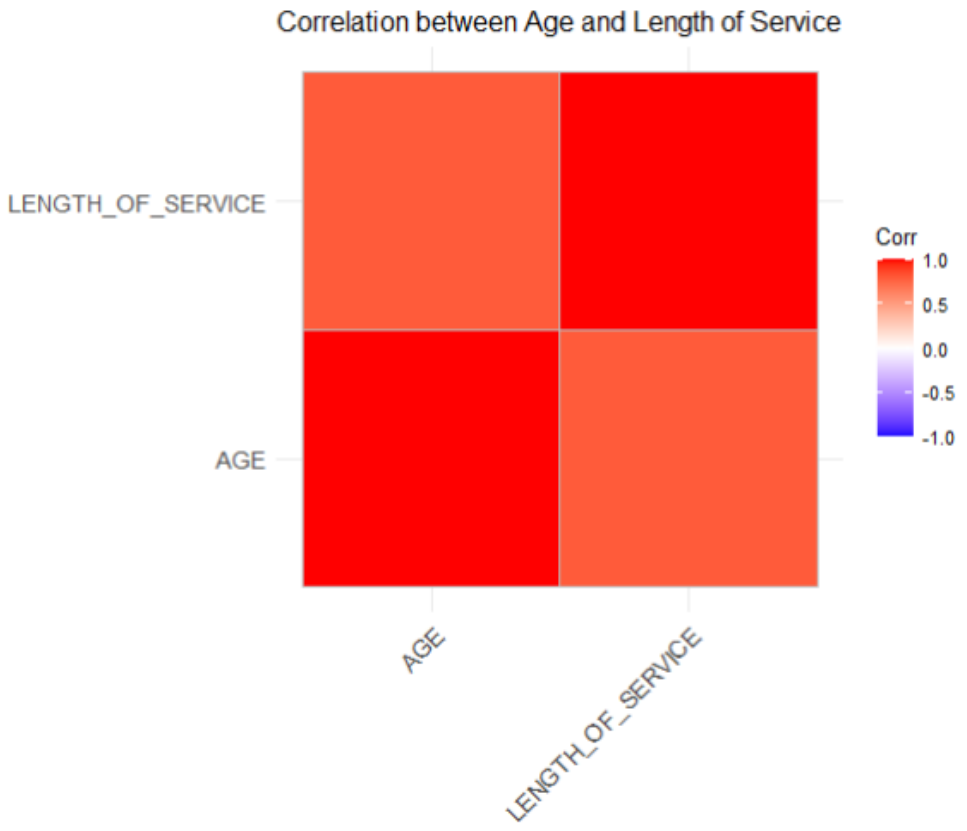
This analysis is to find out the correlation between AGE and LENGTH_OF_SERVICE.

```
age_los <- data %>% select(AGE, LENGTH_OF_SERVICE)
cor_matrix <- round(cor(age_los), 1)
cor_matrix
ggcorrplot(cor_matrix) + labs(title="Correlation between Age and Length of Service")
```

Code: Correlation Matrix between Age and Length of Service

```
> cor_matrix
                  AGE LENGTH_OF_SERVICE
AGE               1.0               0.8
LENGTH_OF_SERVICE 0.8               1.0
```

Output: Correlation Matrix between Age and Length of Service



Graph: Correlation Matrix between Age and Length of Service

The relationship between "AGE" and "LENGTH_OF_SERVICE" is quantified using the Pearson correlation coefficient, which is a measure of the linear association between two variables. From the output, the correlation coefficients along the diagonal of the table are all equal to 1 because each variable is perfectly correlated with itself. These cells aren't useful for interpretation, so we should focus on the other cells in the matrix. The correlation between "AGE" and "LENGTH_OF_SERVICE" is 0.8, which

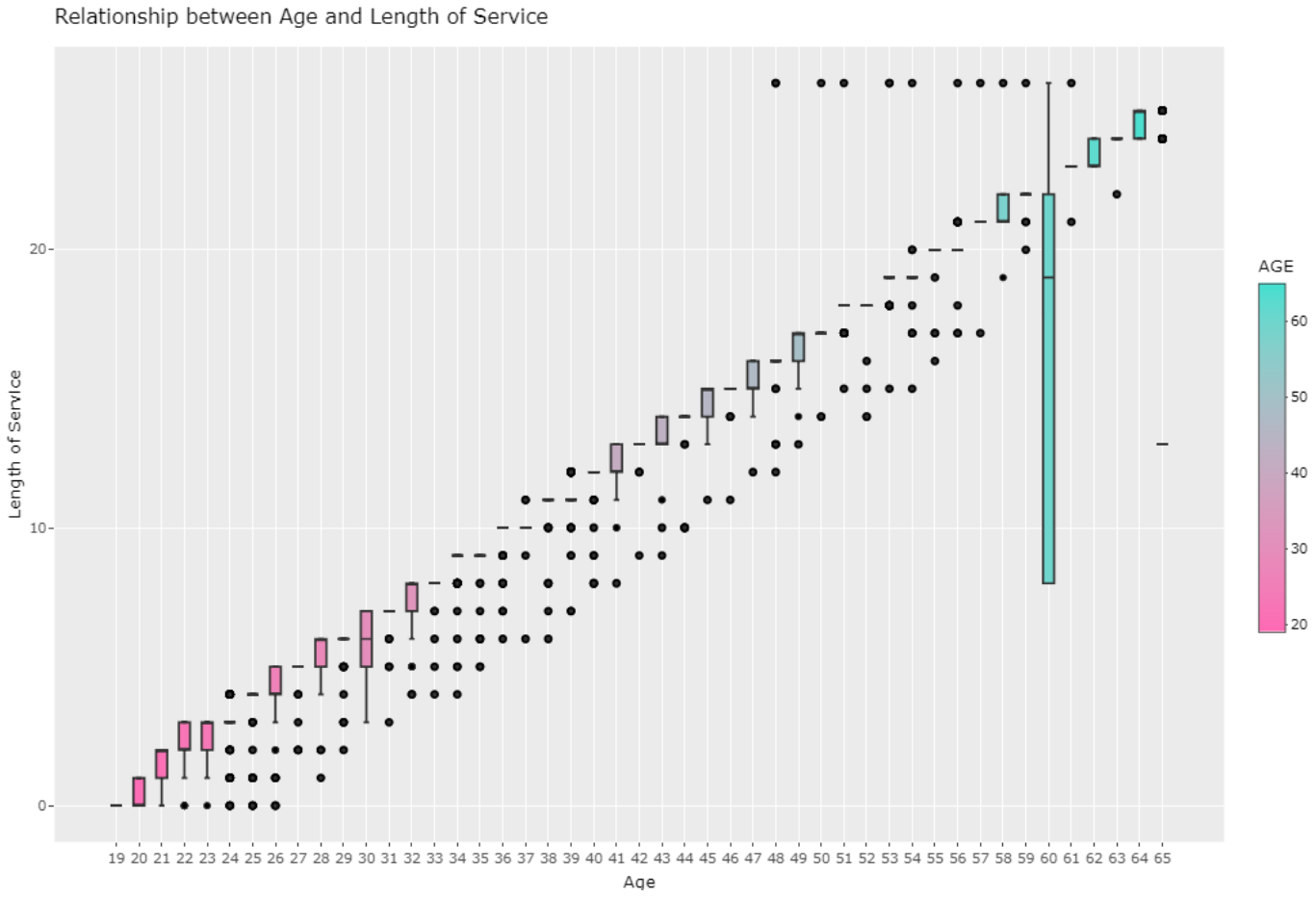
indicates that they're strongly positively correlated. Older age is strongly related to higher length of service. (Zach, 2022)

Analysis 6.1.3.2 - Relationship between Age and Length of Service

This analysis is to find out the relationship between AGE and LENGTH_OF_SERVICE.

```
ggplotly(data %>% select(AGE, LENGTH_OF_SERVICE) %>%
   ggplot(aes(x=AGE, y=LENGTH_OF_SERVICE, fill=AGE)) +
   geom_boxplot(aes(group=AGE)) +
   scale_fill_gradient("AGE", low="hotpink", high="turquoise") +
   labs(x="Age", y="Length of Service", title="Relationship between Age and Length of Service") +
   scale_x_continuous(breaks=unique(data$AGE)))
```

Code: Box Plot for the Relationship between Age and Length of Service



Graph: Box Plot for the Relationship between Age and Length of Service

From the graph, we can see that the relationship between "AGE" and "LENGTH_OF_SERVICE" is linear. From 48 years old onwards, we could see that some employees have served the company for 26 years, which is the highest length of service in the dataset. Hence we know that the employees that have served the company for the longest time are now 48 to 61 years old. Another thing to note is that for ages 60 and 65, some employees served for 8 and 13 years respectively, which was below average.

Conclusion 1.3

"AGE" and "LENGTH OF SERVICE" have a correlation of 0.8, indicating that they are strongly positively correlated, as when plotted, the graph also suggests that the relationship between the two is strongly linear. This also means that the older the age, the greater the length of service. However, some values deviated from the average, as for ages 60 and 65, some employees served for 8 and 13 years respectively, which was below average.

6.2 - Question 2: What Year or Month are Most Employees Getting Terminated?

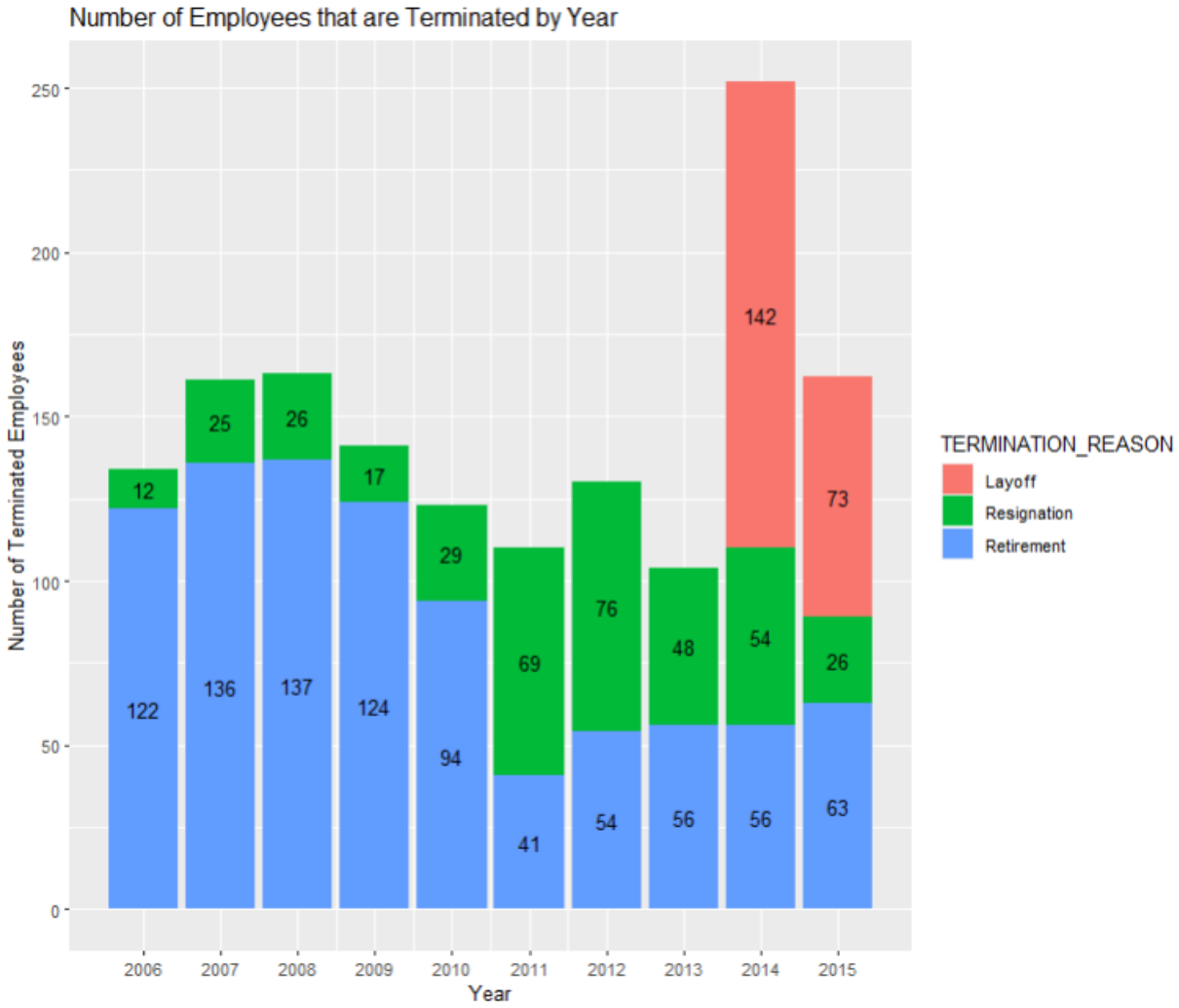
This question is to determine the year and month in which the majority of employees are terminated.

Analysis 6.2.1 - Number of Employees that are Terminated by Year

This analysis is to calculate the number of employees that are terminated by year.

```
data %>% filter(STATUS == "TERMINATED") %>%
  select(RECORD_YEAR,TERMINATION_REASON) %>%
  ggplot(aes(x=RECORD_YEAR, fill=TERMINATION_REASON)) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(x="Year", y="Number of Terminated Employees", title="Number of Employees that are Terminated by Year") +
  scale_x_continuous(breaks=unique(data$RECORD_YEAR))
```

Code: Bar Graph for the Number of Employees that are Terminated by Year



Graph: Bar Graph for the Number of Employees that are Terminated by Year

According to the graph, the most number of terminations occurred in 2014. Aside from that, we can see that employees began to be laid off starting in 2014, with a significant number of 142 employees. We can

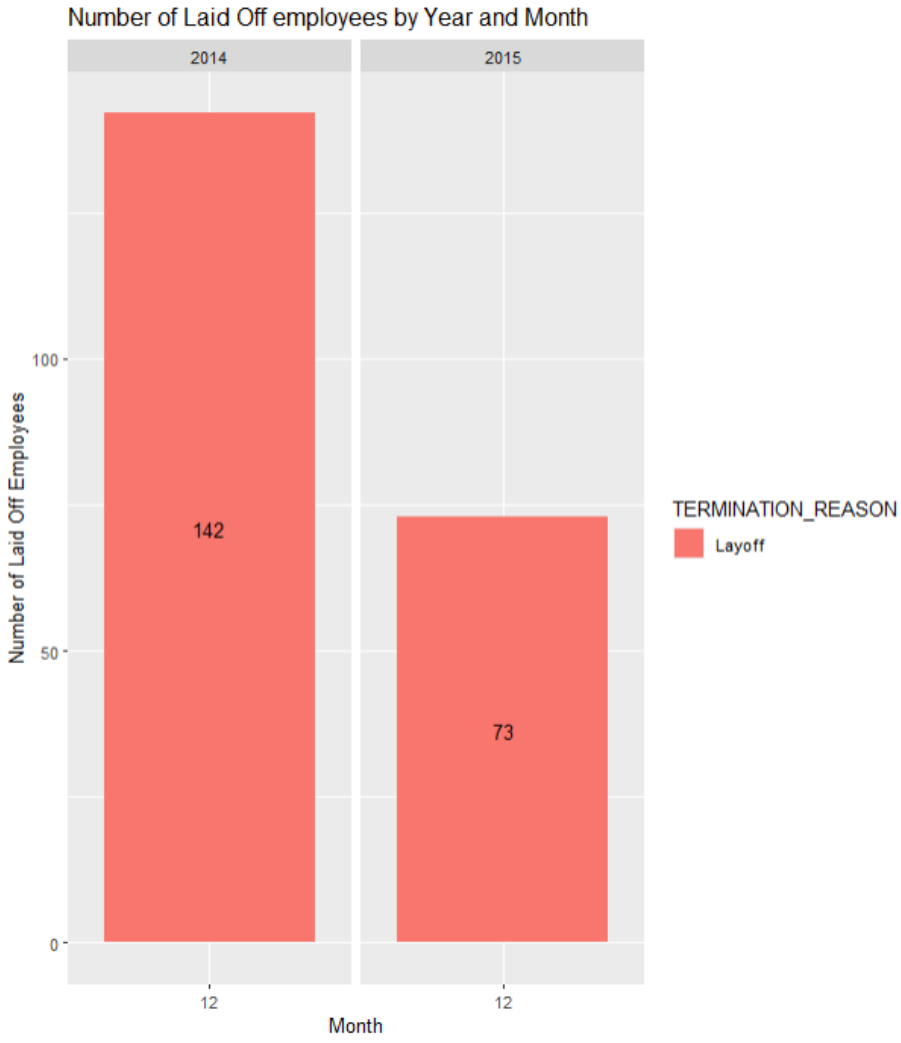
also see that many employees retired between the year 2006 to 2010, while many employees resigned between the year 2011 to 2014.

Analysis 6.2.2 - Number of Laid Off employees by Year and Month

This analysis is to determine the number of laid-off employees by year and month.

```
data %>% filter(TERMINATION_REASON == "Layoff") %>%
  mutate(RECORD_MONTH = format(as.Date(RECORD_DATE, format="%Y/%m/%d"),"%m")) %>%
  select(RECORD_YEAR, RECORD_MONTH,TERMINATION_REASON) %>%
  ggplot(aes(x=RECORD_MONTH, fill=TERMINATION_REASON)) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(x="Month", y="Number of Laid Off Employees", title="Number of Laid Off employees by Year and Month") +
  facet_grid(. ~ RECORD_YEAR)
```

Code: Bar Graph for the Number of Laid Off employees by Year and Month



Graph: Bar Graph for the Number of Laid Off employees by Year and Month

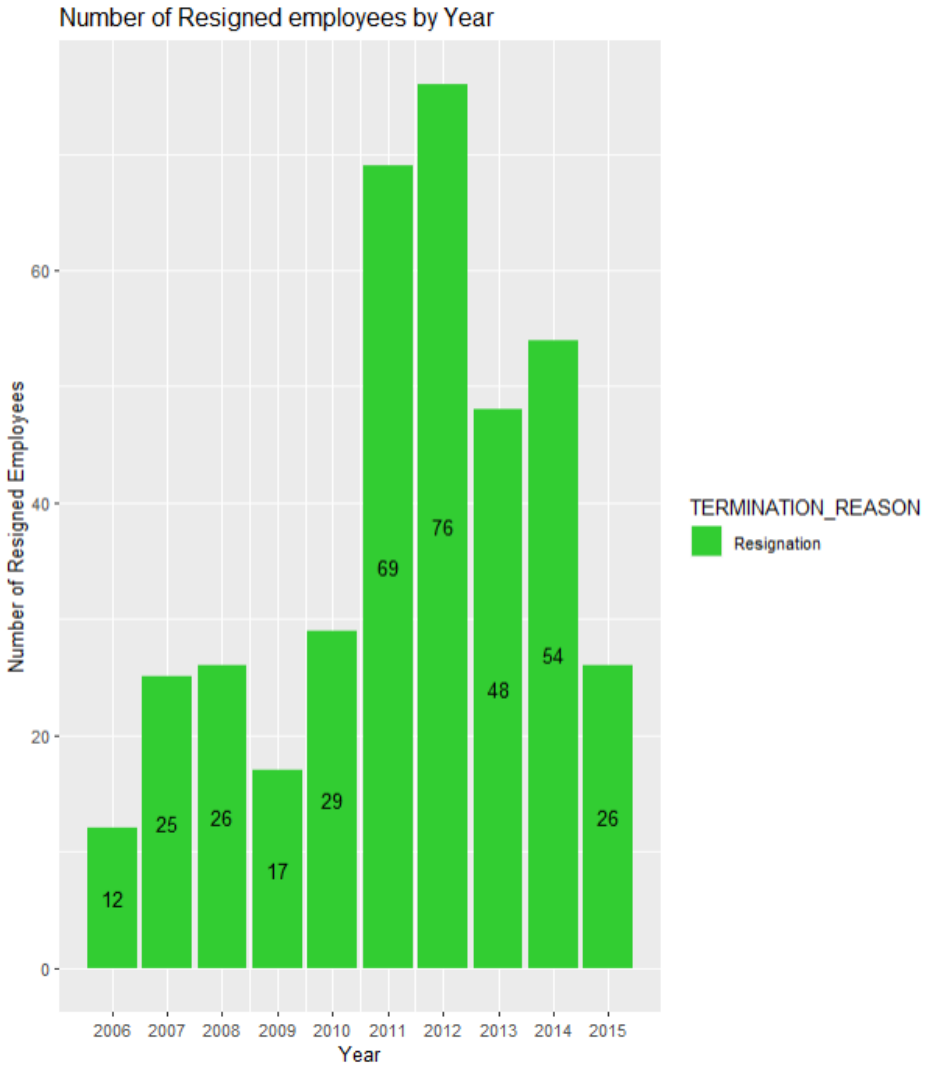
According to the graph, layoffs occurred only in December 2014 and 2015, with the year 2014 having the most number of laid off employees (142 employees).

Analysis 6.2.3 - Number of Resigned Employees by Year

This analysis is to determine the number of resigned employees by year.

```
data %>% filter(TERMINATION_REASON == "Resignation") %>%
  select(RECORD_YEAR, TERMINATION_REASON) %>%
  ggplot(aes(x=RECORD_YEAR, fill=TERMINATION_REASON)) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(x="Year", y="Number of Resigned Employees", title="Number of Resigned employees by Year") +
  scale_x_continuous(breaks=unique(data$RECORD_YEAR)) +
  scale_fill_manual(values = "limegreen")
```

Code: Bar Graph for the Number of Resigned Employees by Year



Graph: Bar Graph for the Number of Resigned employees by Year

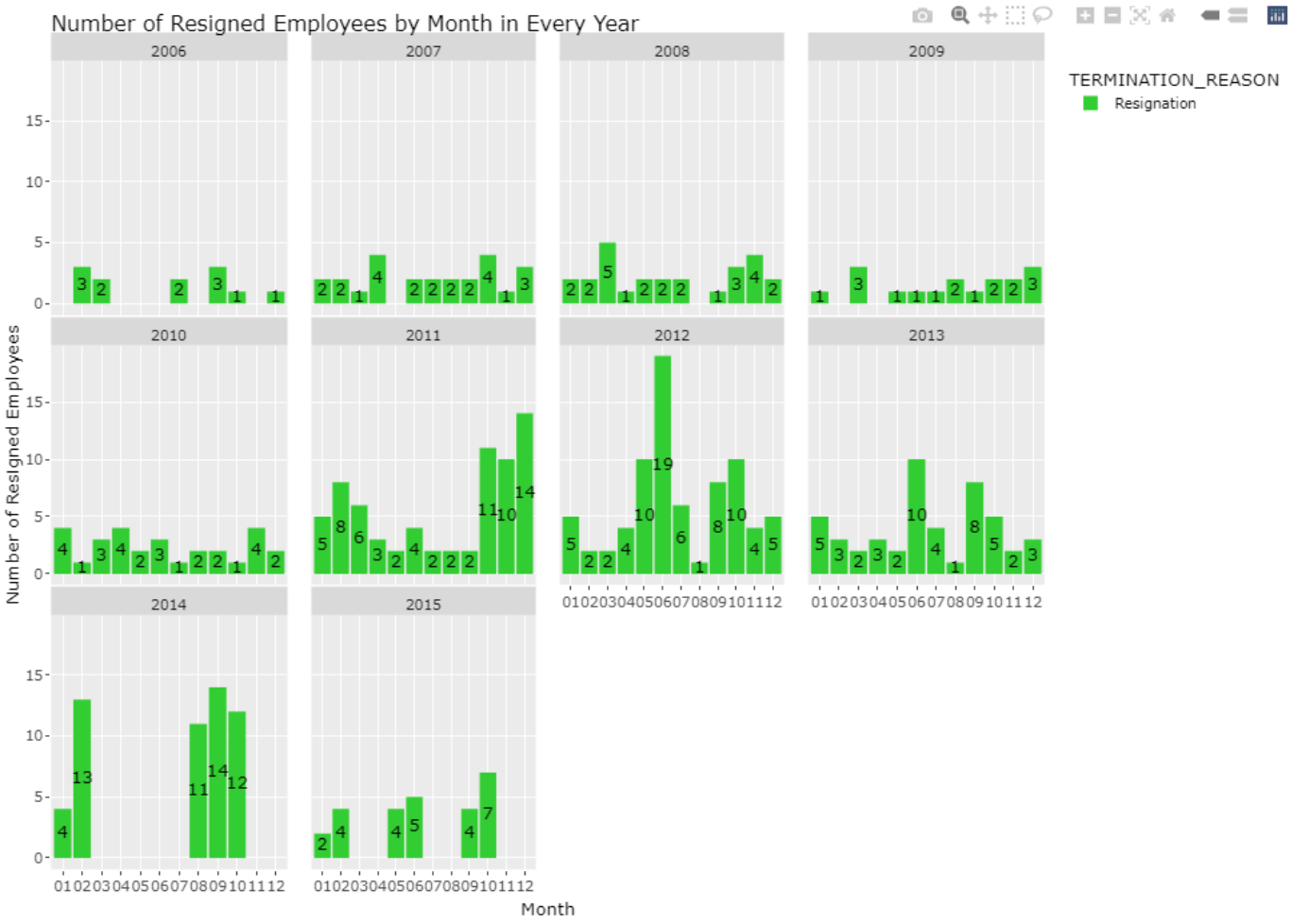
From the graph, we can deduce that the year 2012 has the highest number of resigned employees (76 employees). The year with the fewest resigned employees is 2006, which has only 12 employees. Besides that, the number of resignations also had a significant increase when the year 2011 arrived, from 29 employees in 2010 to 69 employees in 2011.

Analysis 6.2.3.1 - Number of Resigned Employees by Month in Every Year

This analysis is to calculate the number of resigned employees by month in every year.

```
resig_year <- data %>% filter(TERMINATION_REASON == "Resignation") %>%
  mutate(RECORD_MONTH = format(as.Date(RECORD_DATE, format="%Y/%m/%d"),"%m")) %>%
  select(RECORD_YEAR, RECORD_MONTH,TERMINATION_REASON)
ggplotly(ggplot(resig_year, aes(x=RECORD_MONTH, fill=TERMINATION_REASON)) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(x="Month", y="Number of Resigned Employees", title="Number of Resigned Employees by Month in Every Year") +
  scale_fill_manual(values = "limegreen") +
  facet_wrap(. ~ RECORD_YEAR))
```

Code: Bar Graph for the Number of Resigned employees by Month in Every Year



Graph:Bar Graph for the Number of Resigned employees by Month in Every Year

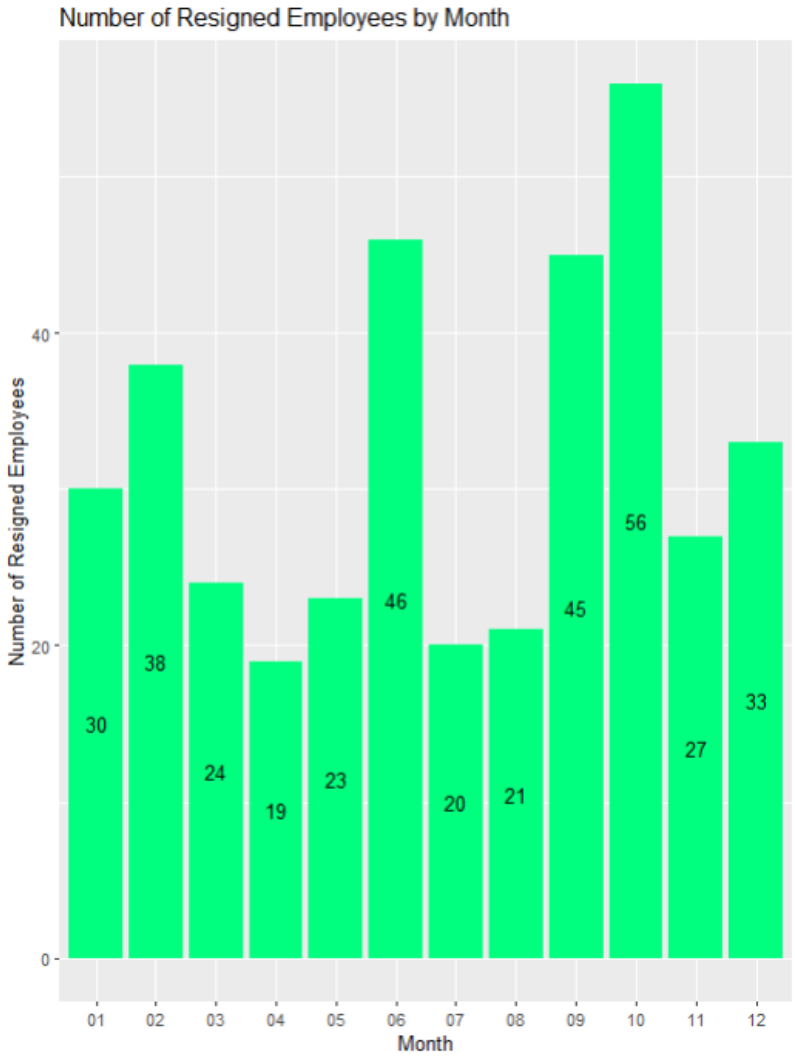
From the graph, we can see that most employees resigned near the end of the year, from September to December. In comparison to the end of the year, we can see that fewer employees resigned at the beginning and middle of the year, with the exception of the mid-year in 2012.

Analysis 6.2.3.2 - Number of Resigned Employees by Month

This analysis is to calculate the number of resigned employees by month.

```
ggplot(resig_year, aes(x=RECORD_MONTH, fill=RECORD_YEAR)) +
  geom_bar(fill="springgreen") +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(x="Month", y="Number of Resigned Employees", title="Number of Resigned Employees by Month")
```

Code: Bar Graph for the Number of Resigned employees by Month



Graph : Bar Graph for the Number of Resigned employees by Month

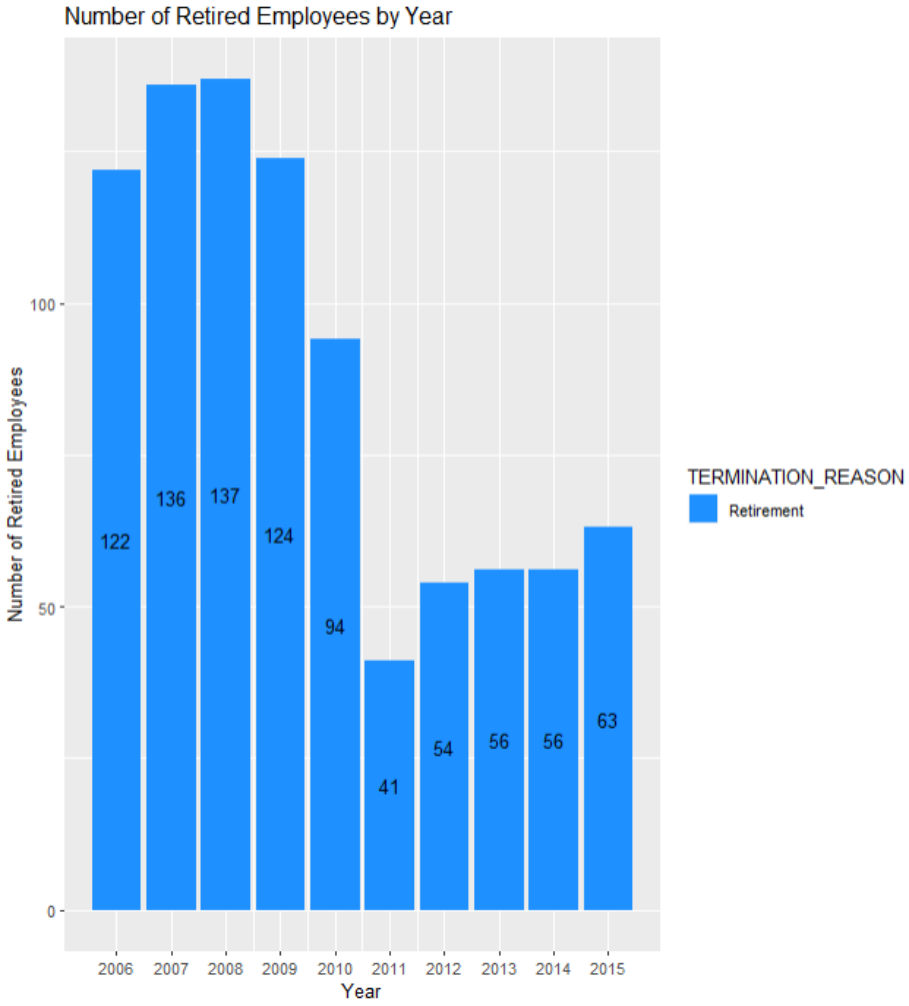
According to the graph, the majority of employees resigned in October (56 employees), followed by 46 employees in June. Moreover, we can deduce that most employees resigned near the end of the year (September to December), as opposed to the beginning and middle of the year. The month with the fewest resigned employees is April, with only 19 employees.

Analysis 6.2.4 - Number of Retired Employees by Year

This analysis is to determine the number of retired employees by year.

```
data %>% filter(TERMINATION_REASON == "Retirement") %>%
  select(RECORD_YEAR, TERMINATION_REASON) %>%
  ggplot(aes(x=RECORD_YEAR, fill=TERMINATION_REASON)) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(x="Year", y="Number of Retired Employees", title="Number of Retired Employees by Year") +
  scale_x_continuous(breaks=unique(data$RECORD_YEAR)) +
  scale_fill_manual(values = "dodgerblue")
```

Code: Bar Graph for the Number of Retired Employees by Year



Graph: Bar Graph for the Number of Retired Employees by Year

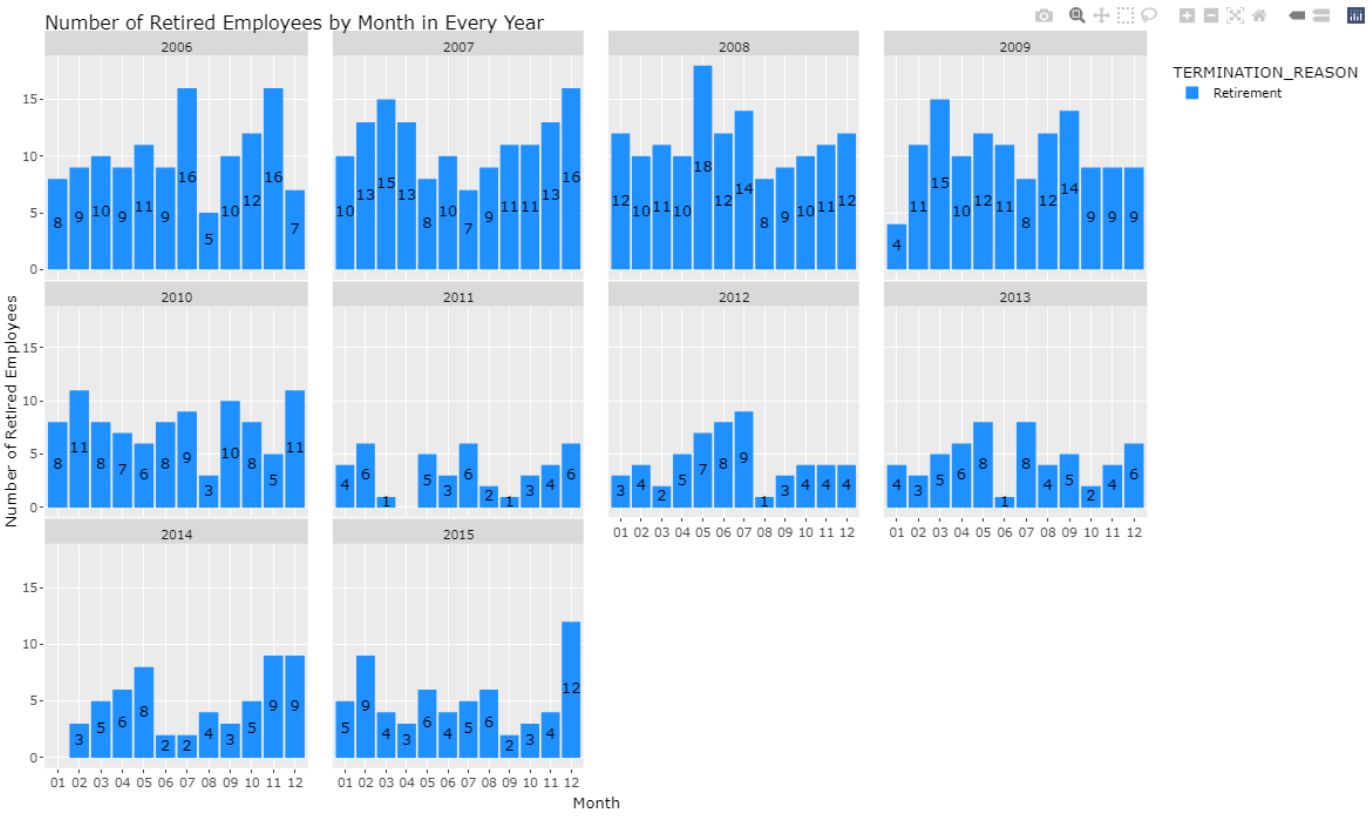
According to the graph, we can deduce that the year 2008 has the most number of retired employees (137 employees), followed closely by the year 2007 with 136 employees. The year with the fewest retired employees is 2011, which has only 41 employees. We could also see that a lot of employees retired from the year 2006 to 2010 compared to the more recent years. Furthermore, the number of resignations decreased significantly from 94 employees in 2010 to 41 employees in 2011.

Analysis 6.2.4.1 - Number of Retired Employees by Month in Every Year

This analysis is to calculate the number of retired employees by month in every year.

```
retire_year <- data %>% filter(TERMINATION_REASON == "Retirement") %>%
  mutate(RECORD_MONTH = format(as.Date(RECORD_DATE, format="%Y/%m/%d"),"%m")) %>%
  select(RECORD_YEAR, RECORD_MONTH,TERMINATION_REASON)
ggplotly(ggplot(retire_year, aes(x=RECORD_MONTH, fill=TERMINATION_REASON)) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(x="Month", y="Number of Retired Employees", title="Number of Retired Employees by Month in Every Year") +
  scale_fill_manual(values = "dodgerblue") +
  facet_wrap(. ~ RECORD_YEAR))
```

Code: Bar Graph for the Number of Retired Employees by Month in Every Year



Graph:Bar Graph for the Number of Retired Employees by Month in Every Year

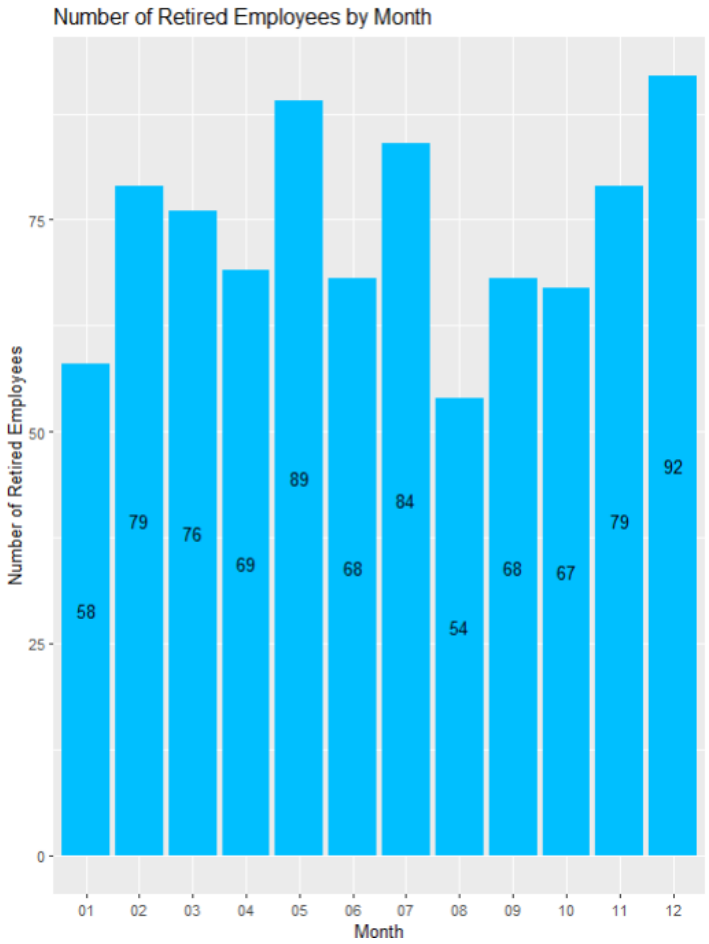
From the graph, we can see that most employees retired near the end of the year, from September to December.

Analysis 6.2.4.2 - Number of Retired Employees by Month

This analysis is to calculate the number of retired employees by month.

```
ggplot(retire_year, aes(x=RECORD_MONTH, fill=RECORD_YEAR)) +
  geom_bar(fill="deepskyblue") +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(x="Month", y="Number of Retired Employees", title="Number of Retired Employees by Month")
```

Code: Bar Graph for the Number of Retired Employees by Month



Graph : Bar Graph for the Number of Retired Employees by Month

Based on the graph, it is evident that most employees retired in December (92 employees), followed by 89 employees in May. The month with the fewest resigned employees is August, with only 54 employees.

Conclusion 2

From Section 5.14, we knew that most employees were terminated through retirement (883 employees), followed by resignation (382 employees), then layoff (215 employees). The most number of terminations occurred in 2014. Many employees retired between the year 2006 to 2010, while many employees resigned between the year 2011 to 2014. Most employees retired between 2006 and 2010, and the year with the most retired employees is 2008 (137 employees), with the majority of employees retiring in December. 2012 has the highest number of resigned employees (76 employees), and most employees resigned near the end of the year, from September to December, with the majority of employees resigning in October (56 employees). Employees are only laid off in December 2014 and 2015, with 2014 having the highest number of layoffs (142 employees).

6.3 - Question 3: What is the Nature of Layoff?

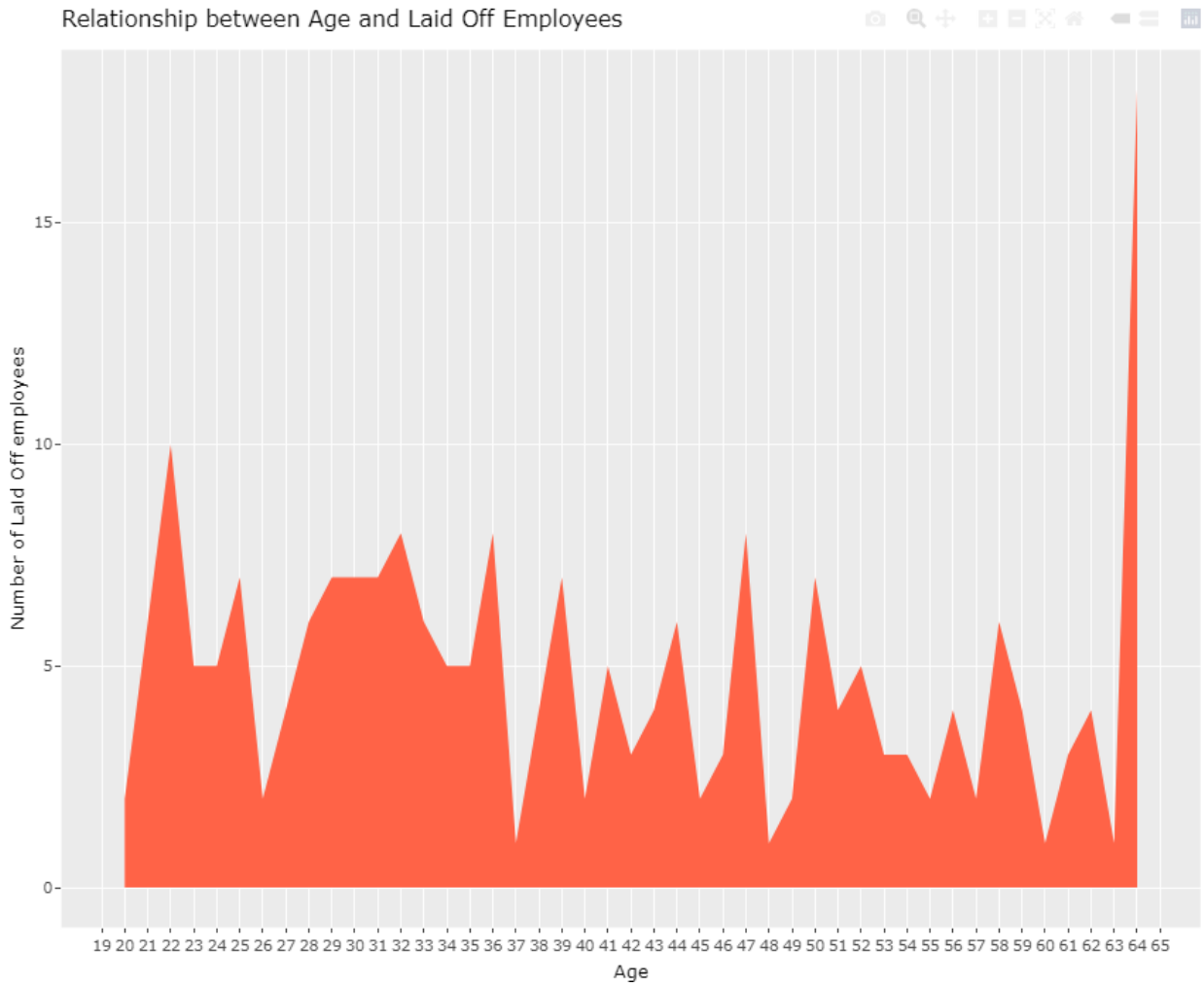
Among all termination reasons, the least employees were terminated through layoff (215 employees), but it only happened abruptly in recent years. Hence, this question is to find out the nature of employee layoff.

Analysis 6.3.1 - Relationship between Age and Laid Off employees

This analysis is to determine the relationship between Age and Laid off employees.

```
ggplotly(data %>% filter(TERMINATION_REASON == "Layoff") %>%
   select(RECORD_YEAR, AGE) %>%
   ggplot(aes(x=AGE)) +
   geom_area(aes(y = ..count..), stat ="bin", binwidth=1, fill="tomato") +
   labs(colour="YEAR", x="Age", y="Number of Laid Off employees", title="Relationship between Age and Laid Off employees") +
   scale_x_continuous(breaks=unique(data$AGE)))
```

Code: Area Graph for the Relationship between Age and Laid Off Employees



Graph: Area Graph for the Relationship between Age and Laid Off Employees

The graph shows that the most employees laid off are 64 years old (18 employees), followed by 10 employees who are 22 years old.

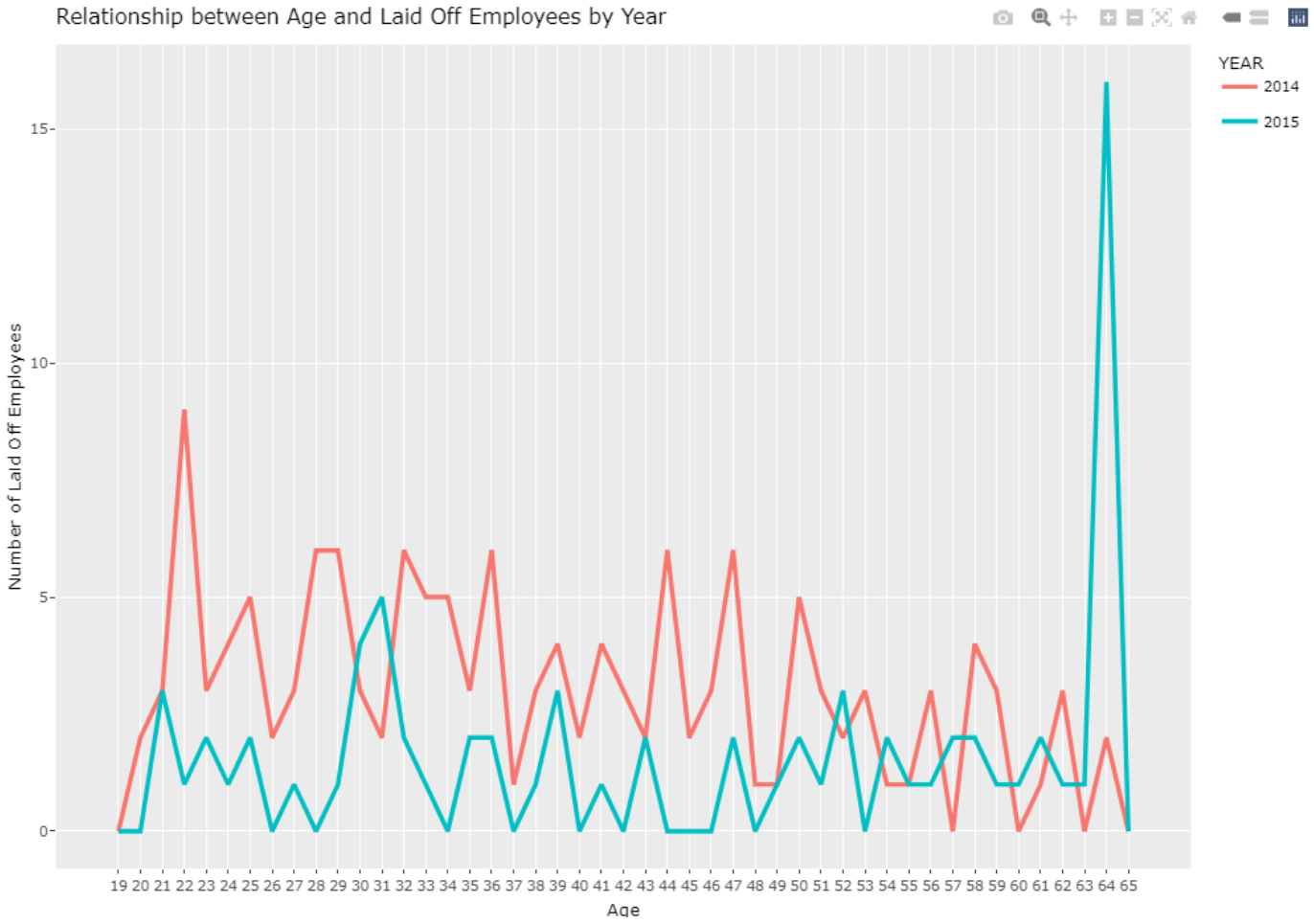
It is reasonable to assume that younger employees are laid off based on their years of experience, which correlates to their age. Since from a logical point of view, we know that a company would not layoff experienced employees, hence we need to think of why seniors may be laid off from different point of views.

Analysis 6.3.1.1 - Relationship between Age and Laid Off employees by Year

This analysis is to determine the relationship between Age and Laid off employees by year.

```
ggplotly(data %>% filter(TERMINATION_REASON == "Layoff") %>%
   select(RECORD_YEAR, AGE) %>%
   ggplot(aes(x=AGE, colour=factor(RECORD_YEAR))) +
   geom_freqpoly(size=1, binwidth=1) +
   labs(colour="YEAR", x="Age", y="Number of Laid Off Employees", title="Relationship between Age and Laid Off Employees by Year") +
   scale_x_continuous(breaks=unique(data$AGE)))
```

Code: Frequency Polygon Graph for the Relationship between Age and Laid Off Employees by Year



Graph: Frequency Polygon Graph for the Relationship between Age and Laid Off Employees by Year

According to this graph, more young employees were fired in 2014, while more senior employees were fired in 2015.

Analysis 6.3.1.2 - Correlation between Age and Length of Service for Laid Off Employees

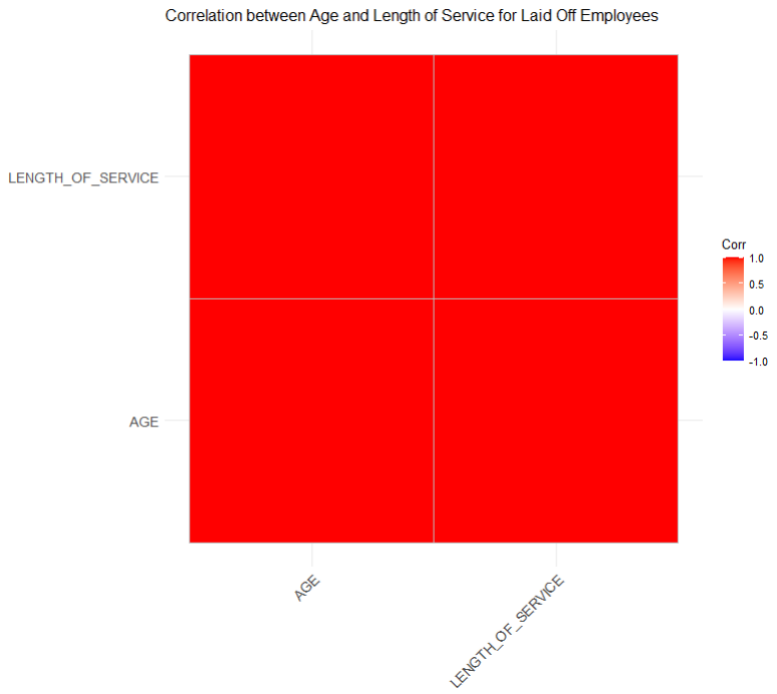
This analysis is to determine if there is any correlation between the age and length of service for laid-off employees.

```
age_los <- data %>% filter(TERMINATION_REASON=="Layoff") %>% select(AGE, LENGTH_OF_SERVICE)
cor_matrix <- round(cor(age_los), 1)
cor_matrix
ggcorrplot(cor_matrix) +
  labs(title="Correlation between Age and Length of Service for Laid Off Employees")
```

Code: Correlation Matrix between Age and Length of Service

```
                  AGE LENGTH_OF_SERVICE
AGE                 1                 1
LENGTH_OF_SERVICE   1                 1
```

Output: Correlation Matrix between Age and Length of Service

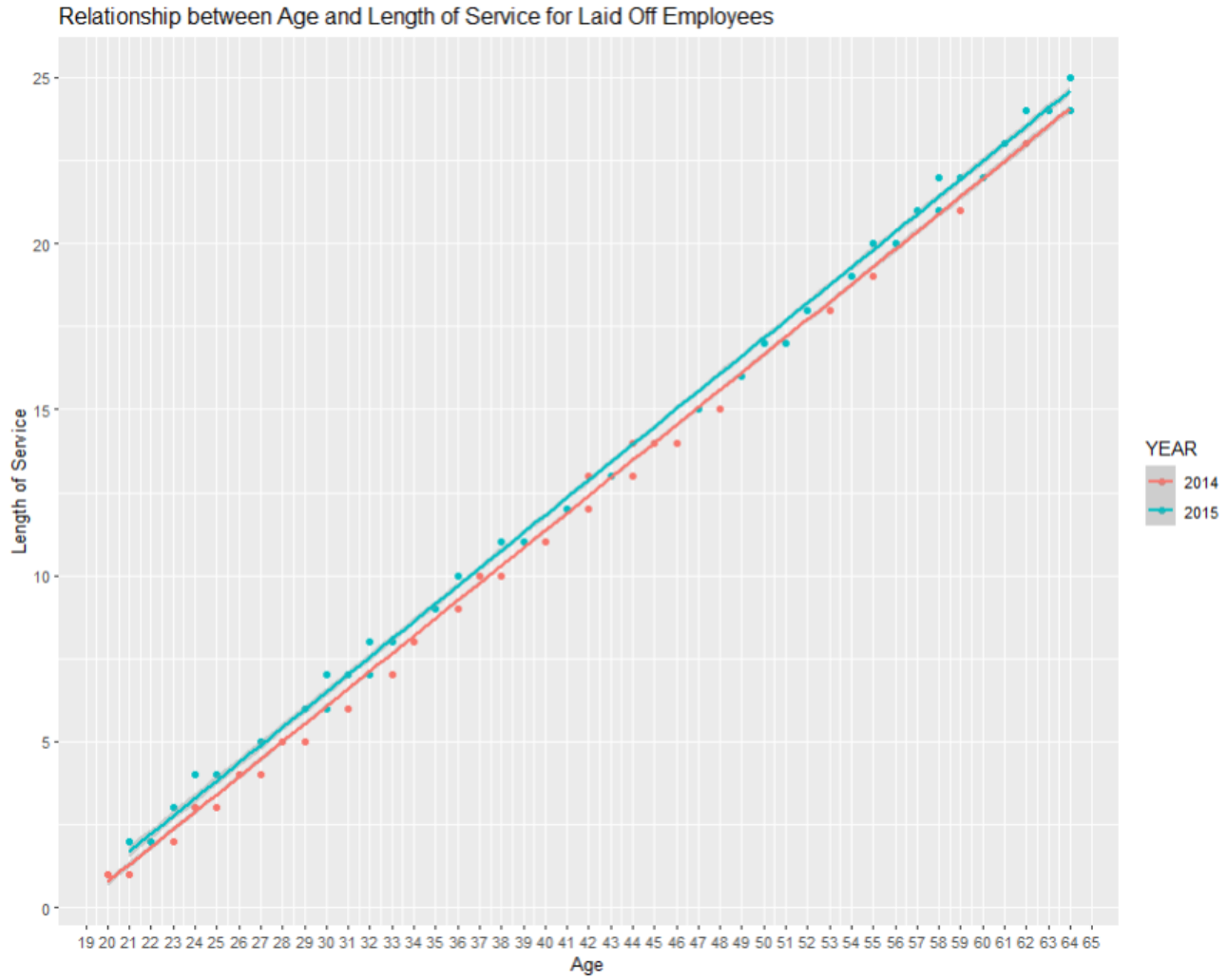


Graph: Correlation Matrix between Age and Length of Service

The Pearson correlation coefficient, which measures the linear association between two variables, is used to quantify the relationship between "AGE" and "LENGTH OF SERVICE." From the output, the correlation coefficients along the diagonal of the table are all equal to 1 because each variable is perfectly correlated with itself. These cells aren't useful for interpretation, so we should focus on the other cells in the matrix. However, the correlation between "AGE" and "LENGTH_OF_SERVICE" is also a 1, indicating that the two variables have a perfectly positive linear correlation. (Zach, 2022)

```
data %>% filter(TERMINATION_REASON == "Layoff") %>%
  select(RECORD_YEAR, AGE, LENGTH_OF_SERVICE) %>%
  ggplot(aes(x=AGE, y=LENGTH_OF_SERVICE, colour=factor(RECORD_YEAR))) +
  geom_point() +
  labs(colour="YEAR", x="Age", y="Length of Service", title="Relationship between Age and Length of Service for Laid Off Employees") +
  scale_x_continuous(breaks=unique(data$AGE)) +
  stat_smooth(method=lm)
```

Code: Relationship between Age and Length of Service for Laid Off Employees



Graph: Point Graph for Relationship between Age and Length of Service for Laid Off Employees

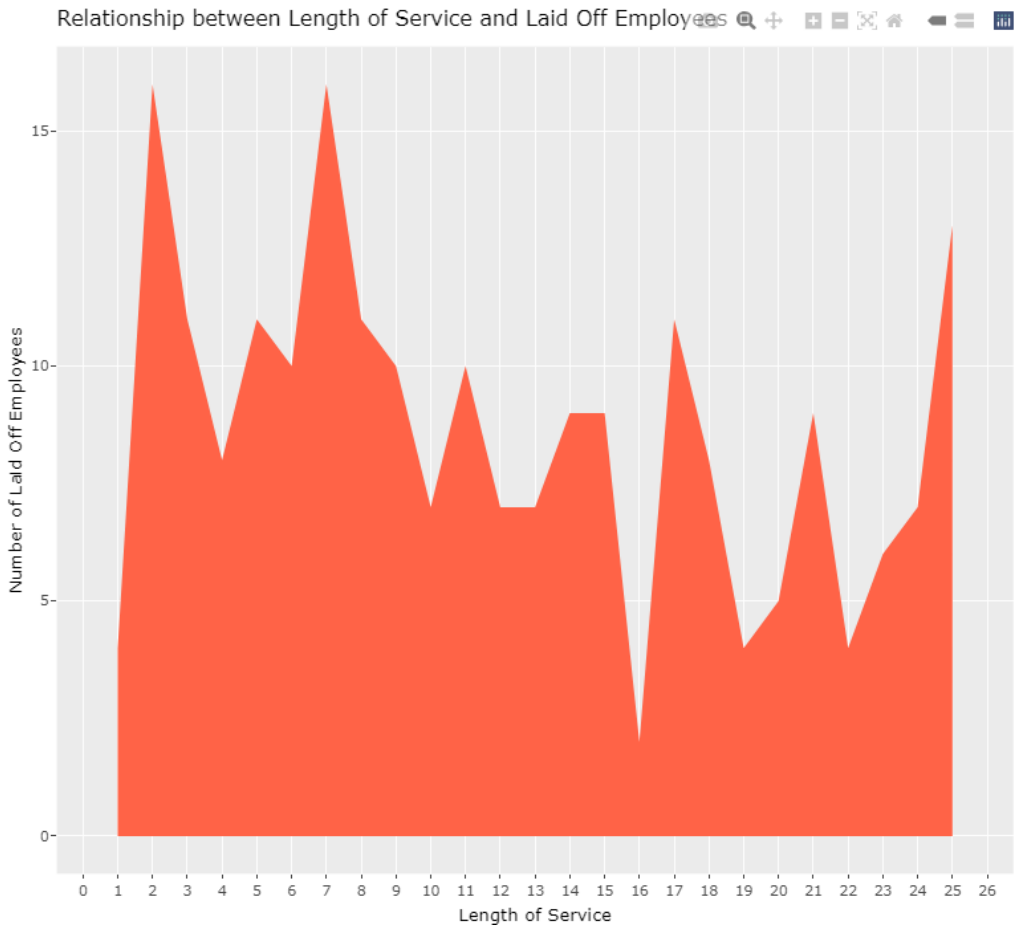
As indicated by both graphs, there is a strong linear relationship between the two variables. As a result, older age is strongly associated with greater length of service. (Zach, 2022)

Analysis 6.3.1.3 - Relationship between Length of Service and Laid Off Employees

This analysis is to determine the relationship between the length of service and Laid off employees.

```
ggplotly(data %>% filter(TERMINATION_REASON == "Layoff") %>%
  select(RECORD_YEAR, LENGTH_OF_SERVICE) %>%
  ggplot(aes(x=LENGTH_OF_SERVICE)) +
  geom_area(aes(y = ..count..), stat ="bin", binwidth=1, fill="tomato") +
  labs(colour="YEAR", x="Length of Service", y="Number of Laid Off Employees", title="Relationship between Length of Service and Laid Off Employees") +
  scale_x_continuous(breaks=unique(data$LENGTH_OF_SERVICE)))
```

Code: Relationship between Length of Service for Laid Off Employees



Graph: Area Graph for the Relationship between Length of Service and Laid Off Employees

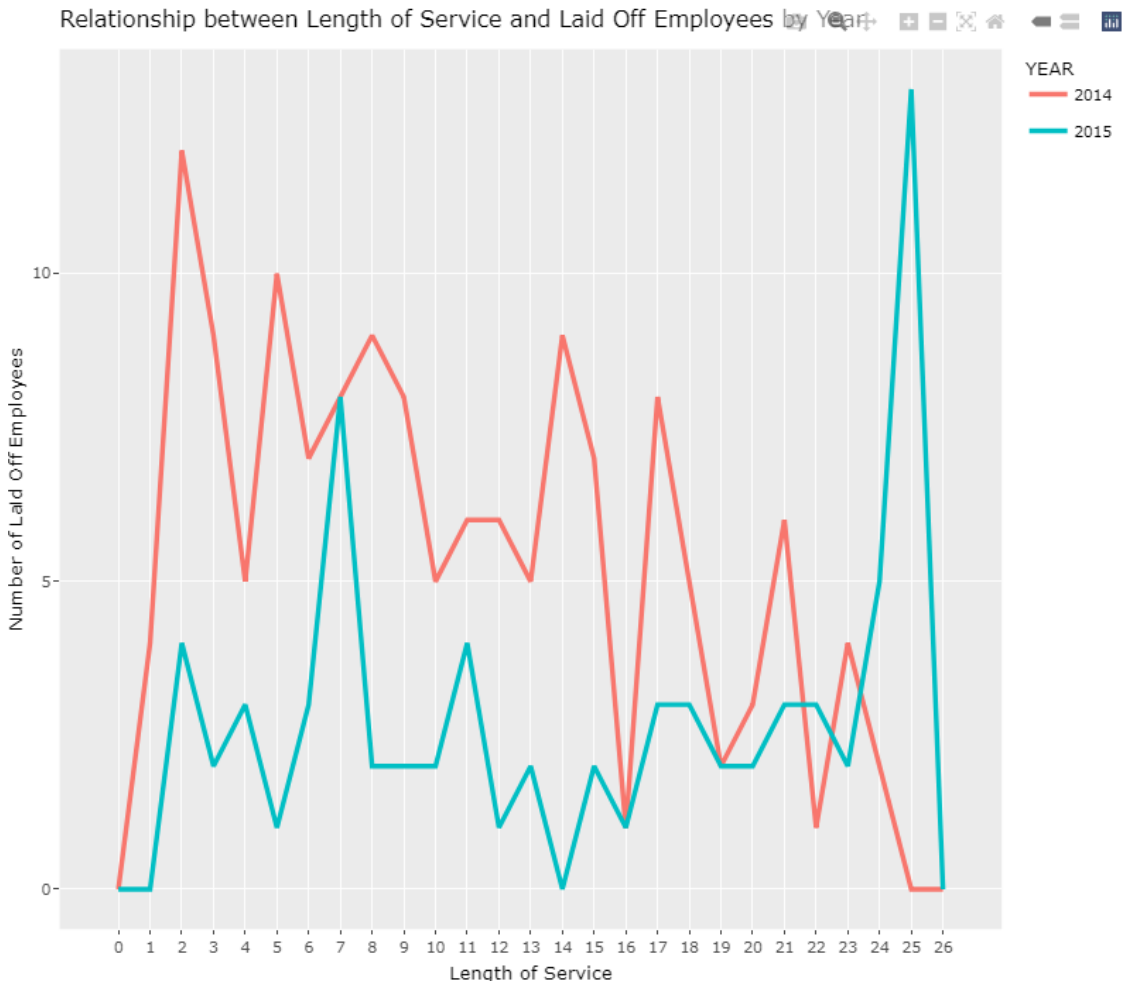
Based on the graph, 13 laid off employees have served for 25 years, and 16 laid off employees have served for 2 or 7 years. We already knew from Analysis 6.3.1.1 that the majority of employees laid off are 64 years old (18 employees), followed by 10 employees who are 22 years old. Because the correlation between "AGE" and "LENGTH OF SERVICE" in Analysis 6.3.1.2 is a perfect positive linear correlation, we can conclude that no seniors have less than 17 years of experience in the company. Hence, we can deduce that the reason why younger employees are laid off is due to their lack of experience, as indicated by their short length of service.

Analysis 6.3.1.4 - Relationship between Length of Service and Laid Off Employees by Year

This analysis is to determine the relationship between the length of service and Laid off employees by year.

```
ggplotly(data %>% filter(TERMINATION_REASON == "Layoff") %>%
  select(RECORD_YEAR, LENGTH_OF_SERVICE) %>%
  ggplot(aes(x=LENGTH_OF_SERVICE, colour=factor(RECORD_YEAR))) +
  geom_freqpoly(size=1, binwidth=1) +
  labs(colour="YEAR", x="Length of Service", y="Number of Laid Off Employees", title="Relationship between Length of Service and Laid Off Employees by Year") +
  scale_x_continuous(breaks=unique(data$LENGTH_OF_SERVICE)))
```

Code: Relationship between Length of Service and Laid Off Employees by Year



Graph: Frequency Polygon Graph for the Relationship between Length of Service and Laid Off Employees by Year

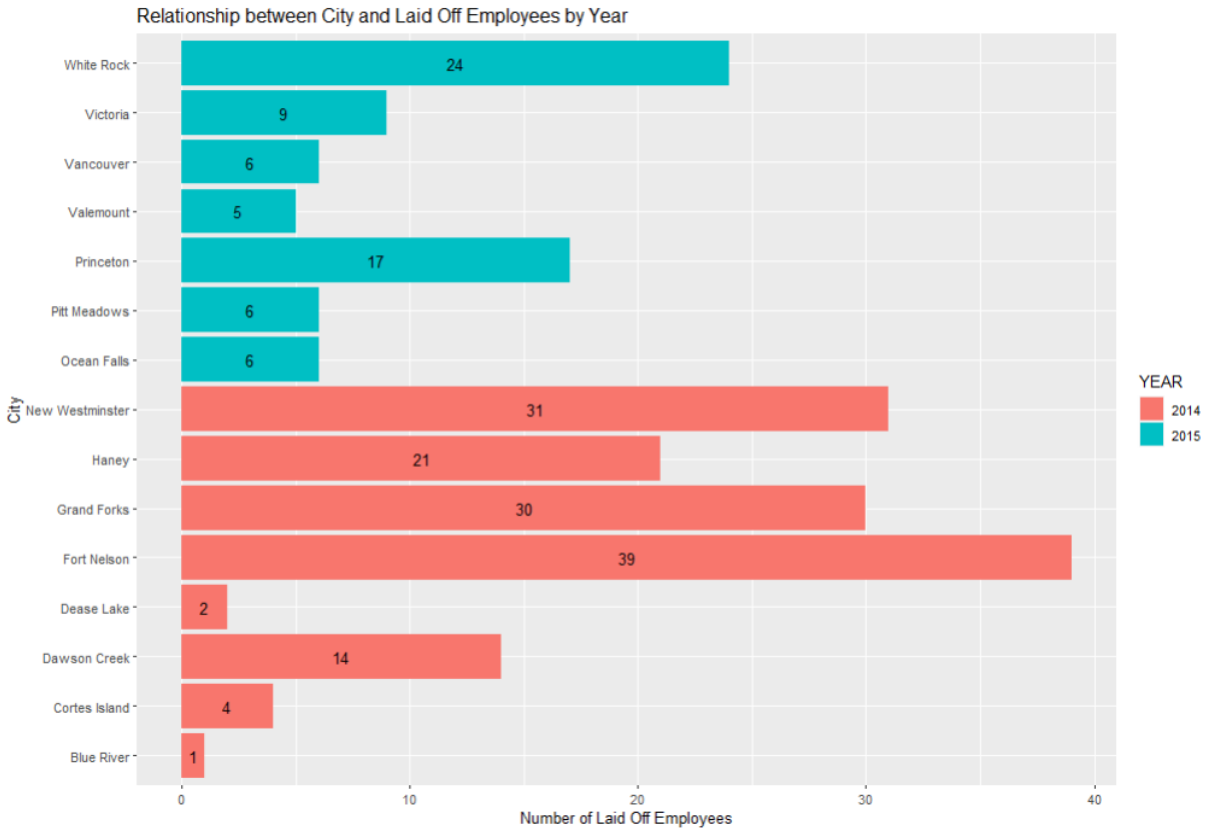
Since the correlation between "AGE" and "LENGTH_OF_SERVICE" is a perfect positive linear correlation from Analysis 6.3.1.2, this graph would definitely resemble Analysis 6.3.1.1. We know more young employees were fired during the year 2014, hence most young employees (12 employees) have a short length of service which is 2 years. On the other hand, we know more older employees were fired during the year 2015, hence most older employees (13 employees) have a long length of service which is 25 years.

Analysis 6.3.2 - Relationship between City and Laid Off employees

This analysis is to determine the relationship between the city and Laid off employees by year.

```
data %>% filter(TERMINATION_REASON == "Layoff") %>%
  select(RECORD_YEAR, CITY) %>%
  ggplot(aes(x=CITY, fill=factor(RECORD_YEAR))) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="YEAR", x="City", y="Number of Laid Off Employees", title="Relationship between City and Laid Off Employees by Year") +
  coord_flip()
```

Code: Relationship between City and Laid Off Employees by Year



Graph: Bar Graph of the Relationship between City and Laid Off Employees by Year

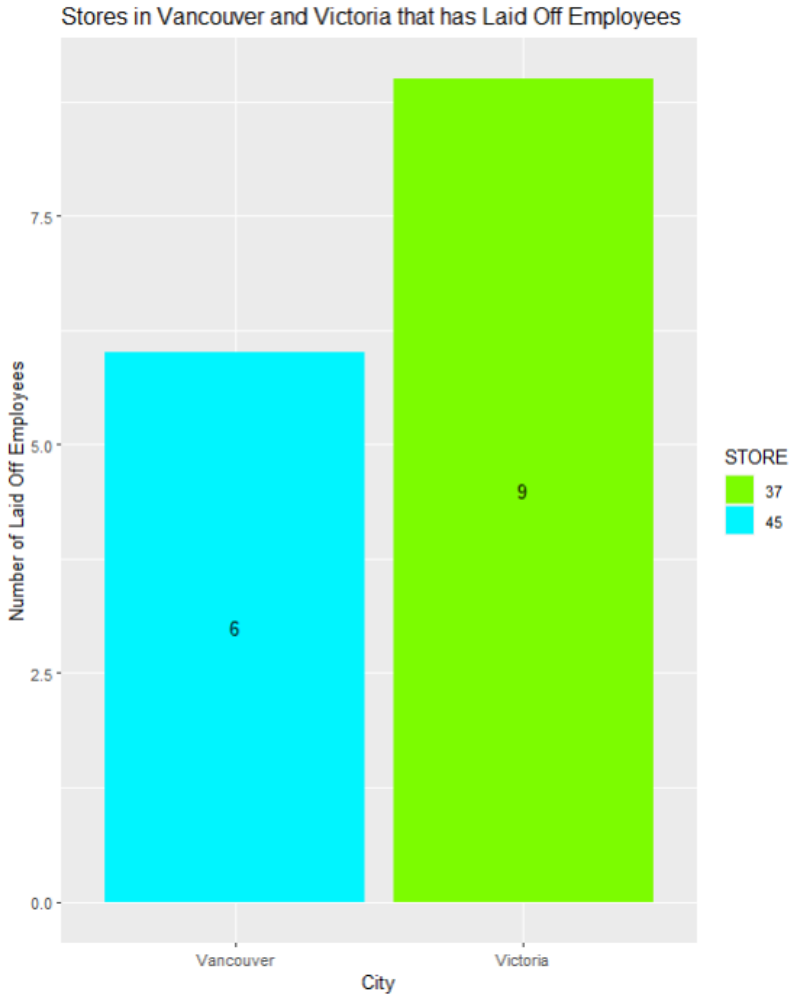
From the graph, we could see that the company fired a number of employees in certain cities in 2014, the most of which were 39 employees in Fort Nelson, and then in certain cities in 2015, the most of which were 24 employees in White Rock. The least amount of laid-off employees in 2014 is 1 employee in Blue River, and 5 employees in Valemount in 2015. Since Vancouver and Victoria that has more than 1 store are on the list, we must determine which Stores in Vancouver and Victoria have laid-off employees.

Analysis 6.3.2.1 - Stores in Vancouver and Victoria that have Laid Off employees

This analysis is to determine which stores in Vancouver and Victoria have Laid Off employees.

```
data %>% filter(TERMINATION_REASON == "Layoff", (CITY=="Vancouver" | CITY=="Victoria")) %>%
  select(STORE, CITY) %>%
  ggplot(aes(x=CITY, fill=STORE)) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="STORE", x="City", y="Number of Laid Off Employees", title="Stores in Vancouver and Victoria that has Laid Off Employees") +
  scale_fill_manual(values = c("lawngreen","turquoise1"))
```

Code: Stores in Vancouver and Victoria that has Laid Off Employees



Graph: Bar Graph of Stores in Vancouver and Victoria that has Laid Off Employees

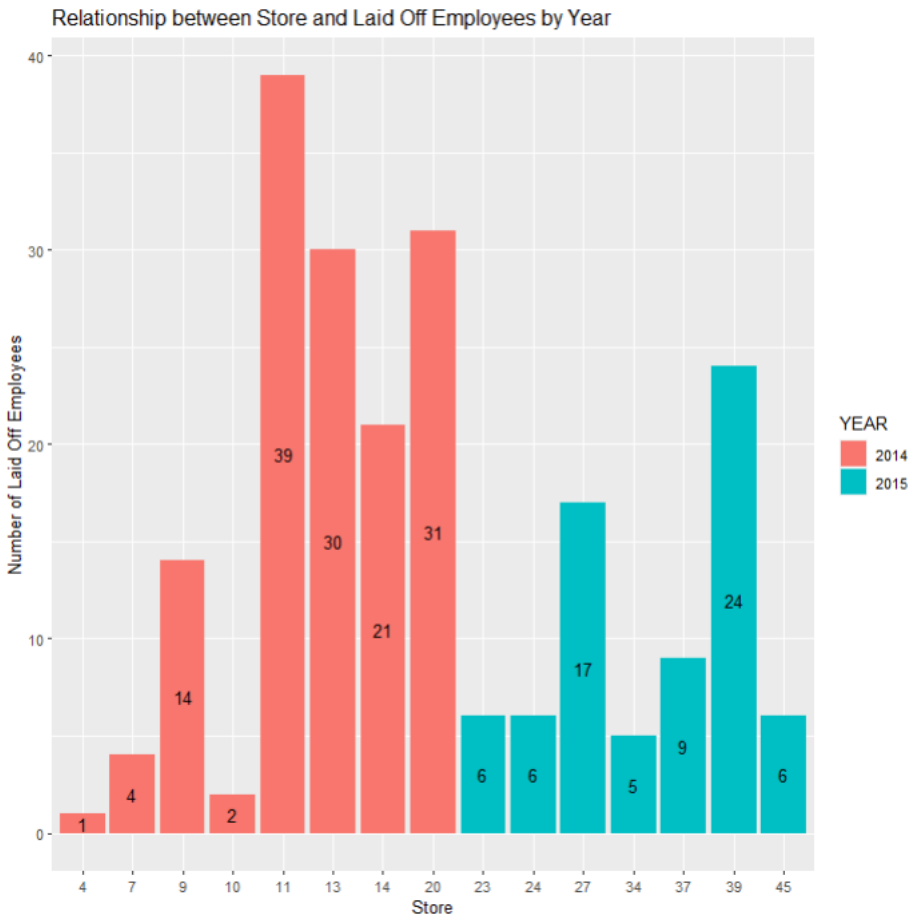
From the output, we know that 6 laid-off employees in Vancouver came from Store 45, while 9 laid-off employees in Victoria came from Store 37. It is important to note that no one was laid-off from the Head Office (Store 35) in Vancouver.

Analysis 6.3.2.2 - Relationship between Store and Laid Off employees

This analysis is to determine the relationship between Store and Laid Off employees.

```
data %>% filter(TERMINATION_REASON == "Layoff") %>%
  select(RECORD_YEAR, STORE) %>%
  ggplot(aes(x=STORE, fill=factor(RECORD_YEAR))) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="YEAR", x="Store", y="Number of Laid Off Employees", title="Relationship between Store and Laid Off Employees by Year")
```

Code: Relationship between Store and Laid Off employees



Graph: Bar Graph of Relationship between Store and Laid Off employees

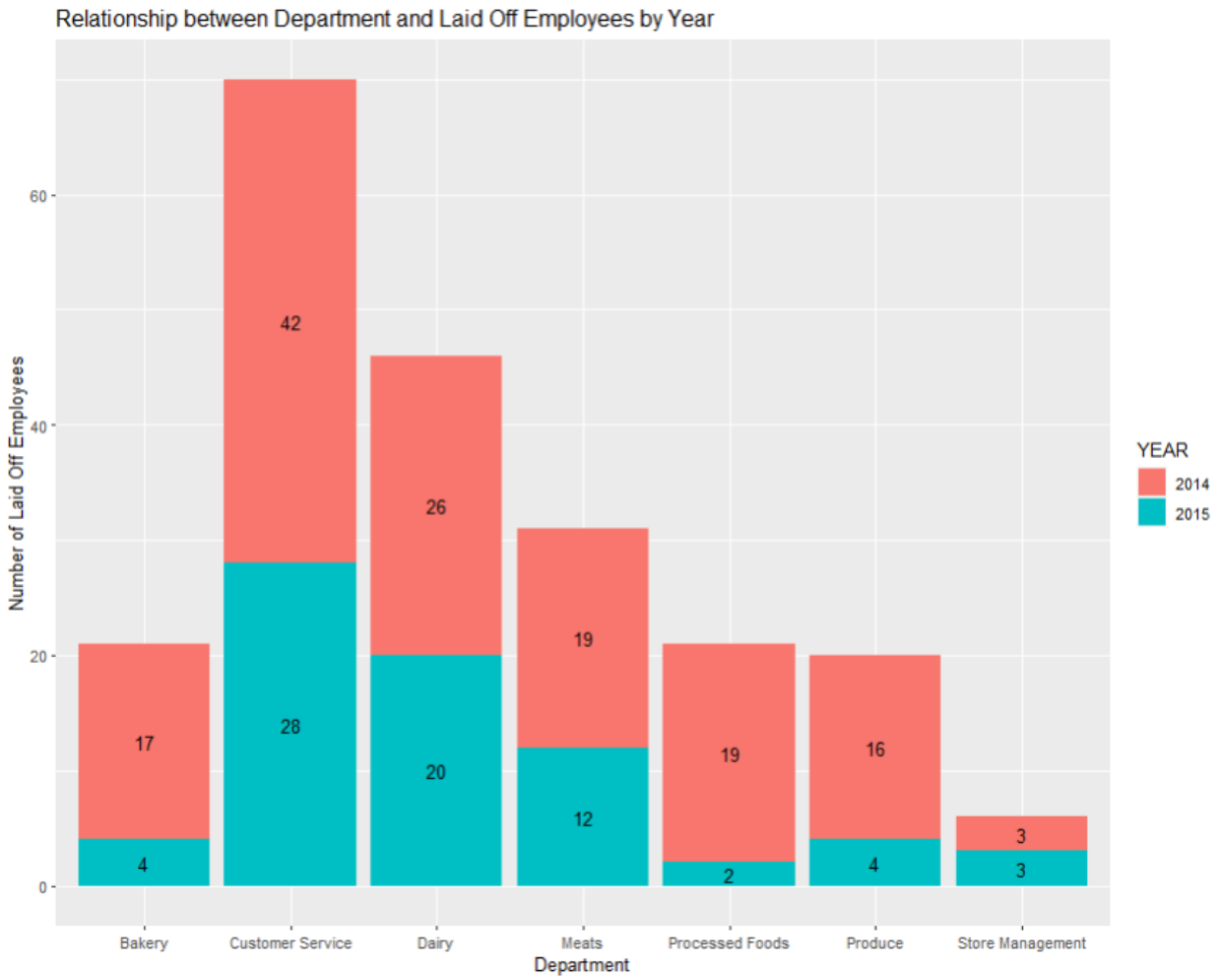
This graph aligns with Analysis 6.3.2. From the graph, we could see that the company fired a number of employees in certain stores in 2014, the most of which were 39 employees at Fort Nelson (Store 11), then certain stores in 2015, the most of which were 24 employees at White Rock (Store 39). The least amount of laid-off employees in 2014 is 1 employee in Blue River (Store 1), and 5 employees in Valemount (Store 34) in 2015.

Analysis 6.3.3 - Relationship between Department and Laid Off employees

This analysis is to determine the relationship between Department and Laid Off employees.

```
data %>% filter(TERMINATION_REASON == "Layoff") %>%
  select(RECORD_YEAR, DEPARTMENT) %>%
  ggplot(aes(x=DEPARTMENT, fill=factor(RECORD_YEAR))) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="YEAR", x="Department", y="Number of Laid Off Employees", title="Relationship between Department and Laid Off Employees by Year")
```

Code: Relationship between Department and Laid Off Employees by Year



Graph: Stacked Bar Graph of the Relationship between Department and Laid Off Employees by Year

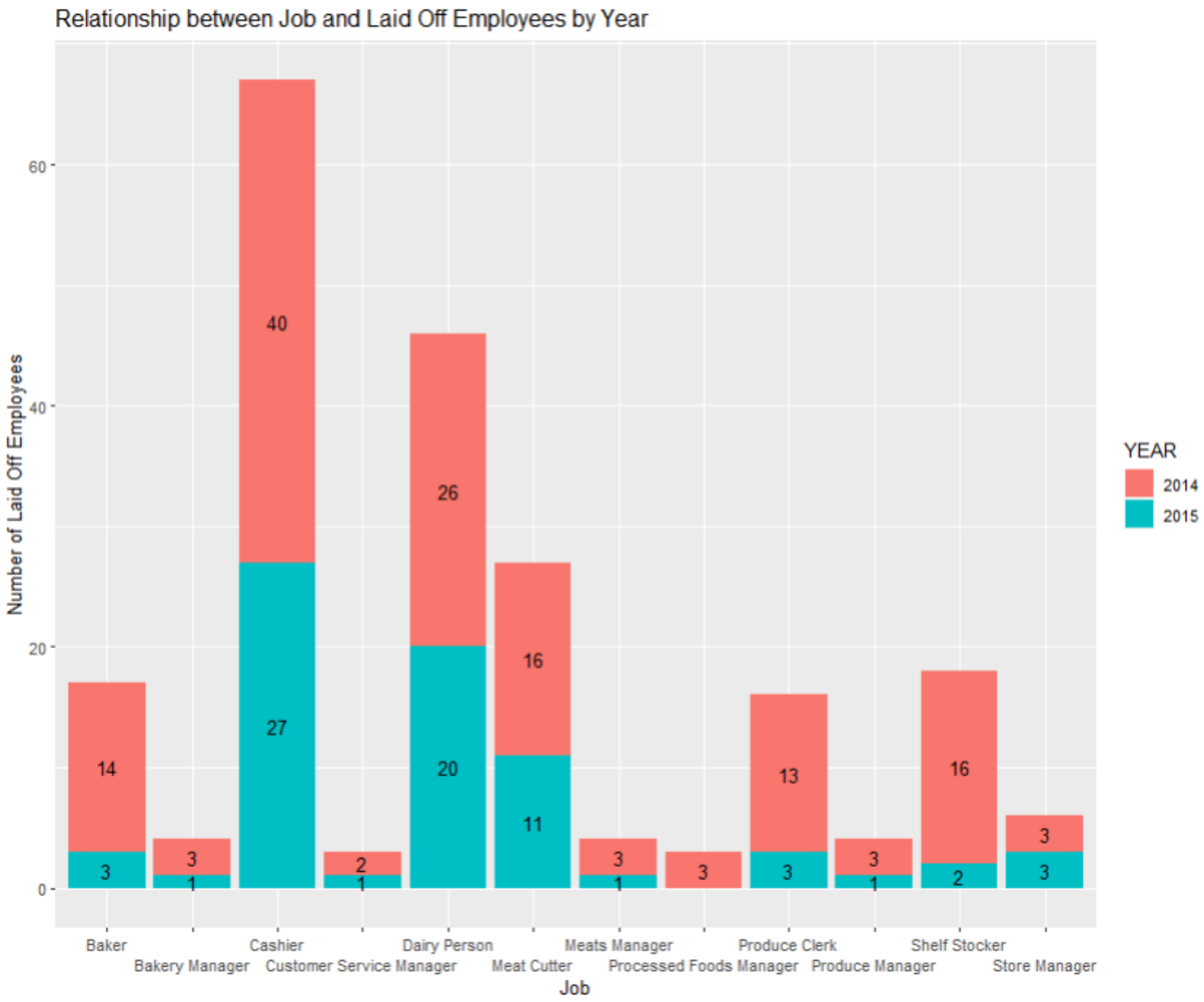
Since no one from Head Office is laid off, the statistics only include departments that are commonly found in Stores. For both years, a large number of employees (42 employees in 2014 and 28 employees in 2015) were laid off from the Customer Service department. The department that had the fewest layoffs (3 employees) in 2014 is the Store Management department, while the department that had the fewest layoffs (2 employees) in 2015 is the Processed Foods department.

Analysis 6.3.4 - Relationship between Job and Laid Off employees

This analysis is to determine the relationship between Job and Laid Off employees.

```
data %>% filter(TERMINATION_REASON == "Layoff") %>%
  select(RECORD_YEAR, JOB_TITLE) %>%
  ggplot(aes(x=JOB_TITLE, fill=factor(RECORD_YEAR))) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="YEAR", x="Job", y="Number of Laid Off Employees", title="Relationship between Job and Laid Off Employees by Year") +
  scale_x_discrete(guide=guide_axis(n.dodge = 2))
```

Code: Relationship between Job and Laid Off Employees by Year



Graph: Stacked Bar Chart of the Relationship between Job and Laid Off Employees by Year

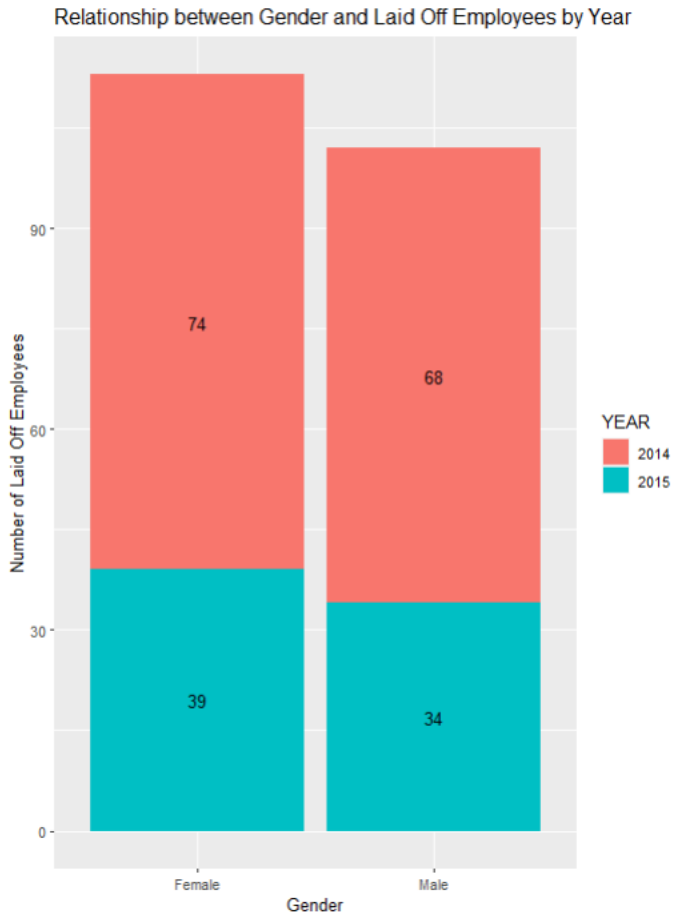
Since we know that from Analysis 6.3.3, a lot of employees were laid off from the Customer Service department for both years, therefore, the job that has the highest employee layoffs (40 employees in 2014 and 27 employees in 2015) is cashier, which is under the Customer Service department. The job that had the least employee layoffs (2 employees) in 2014 is the Customer Service Manager from the Customer Service department, while the department that had the least employee layoff (1 employee) in 2015 is the Bakery Manager from the Bakery department, Customer Service Manager from the Customer Service department, Meats Manager from the Meats department, and Produce Manager from the Produce department.

Analysis 6.3.5 - Relationship between Gender and Laid Off Employees

This analysis is to determine the relationship between Gender and Laid Off employees by year.

```
data %>% filter(TERMINATION_REASON == "Layoff") %>%
  select(RECORD_YEAR, GENDER) %>%
  ggplot(aes(x=GENDER, fill=factor(RECORD_YEAR))) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="YEAR", x="Gender", y="Number of Laid Off Employees", title="Relationship between Gender and Laid Off Employees by Year")
```

Code: Relationship between Gender and Laid Off Employees by Year



Graph: Stacked Bar Chart of the Relationship between Gender and Laid Off Employees by Year

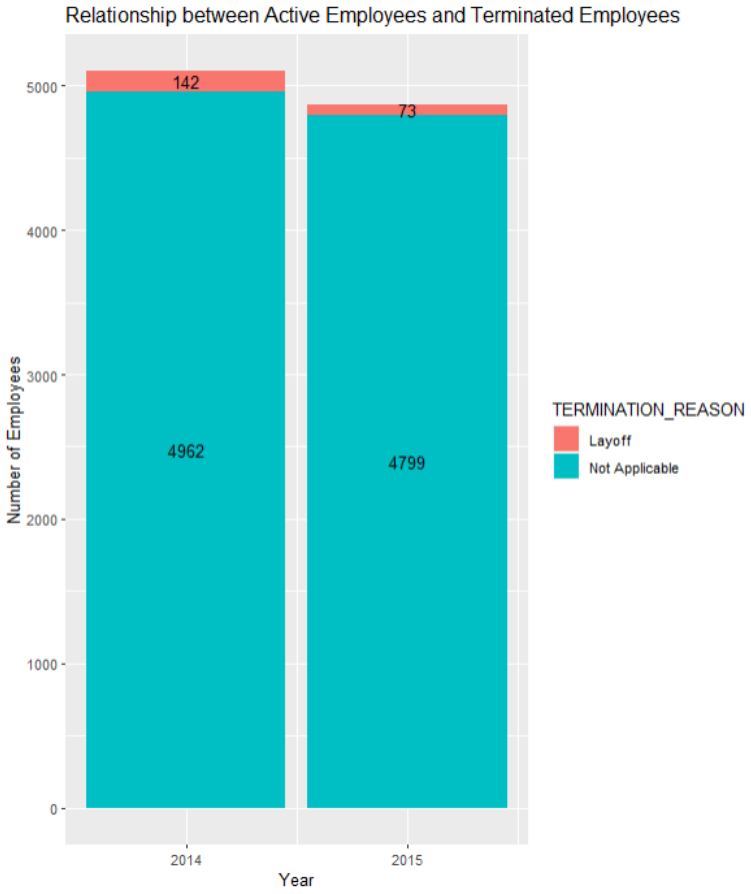
Overall, more female employees are getting laid off in both years than males. But since we know from Section 5.13 that the majority of employees are all female (3278 employees (52.16%)), while the rest being males (3006 employees (47.84%)), we can't determine whether gender is a factor in employees being laid off.

Analysis 6.3.6 - Determining the Layoff Criteria of employees by comparing with the status of Active employees

This analysis is to determine the layoff criteria of employees by comparing with the current attributes of Active employees. To this, we should first examine the relationship between active and terminated employees.

```
data_full %>% select(RECORD_YEAR, STATUS, TERMINATION_REASON) %>%
  filter(TERMINATION_REASON=="Layoff" | TERMINATION_REASON=="Not Applicable" & (RECORD_YEAR==2014 | RECORD_YEAR==2015)) %>%
  ggplot(aes(x=RECORD_YEAR, fill=TERMINATION_REASON)) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(x="Year", y="Number of Employees", title="Relationship between Active Employees and Terminated Employees") +
  scale_x_continuous(breaks=unique(data$RECORD_YEAR))
```

Code: Relationship between Active Employees and Terminated Employees



Graph: Stacked Bar Graph for the Relationship between Active Employees and Terminated Employees

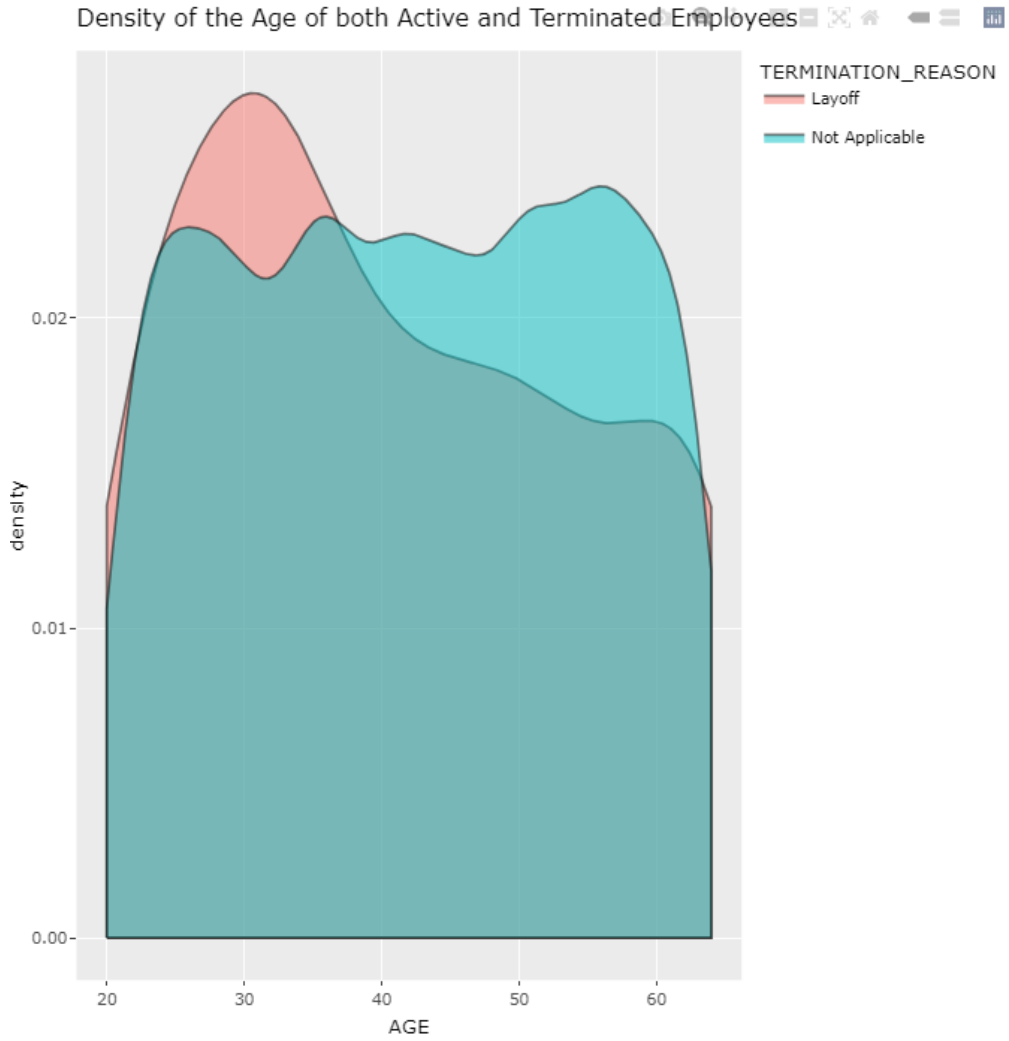
From the graph, we know that the number of layoff employees is only a small percentage compared to the total number of active employees. We could also see that the number of active employees decreased in tandem with the number of layoffs.

Analysis 6.3.6.1 - Age of both Active and Terminated employees

This analysis is to determine the layoff criteria of employees by comparing the age of both active and terminated employees.

```
age_at <- data_full %>%
  select(RECORD_YEAR, TERMINATION_REASON, AGE) %>%
  filter(TERMINATION_REASON=="Layoff" | TERMINATION_REASON=="Not Applicable" & (RECORD_YEAR==2014 | RECORD_YEAR==2015))
ggplotly(ggplot(age_at, aes(x=AGE, fill=TERMINATION_REASON)) +
  geom_density(alpha=.5) +
  ggtitle("Density of the Age of both Active and Terminated Employees"))
```

Code: Density of the Age of both Active and Terminated Employees



Graph: Density Graph for the Density of the Age of both Active and Terminated Employees

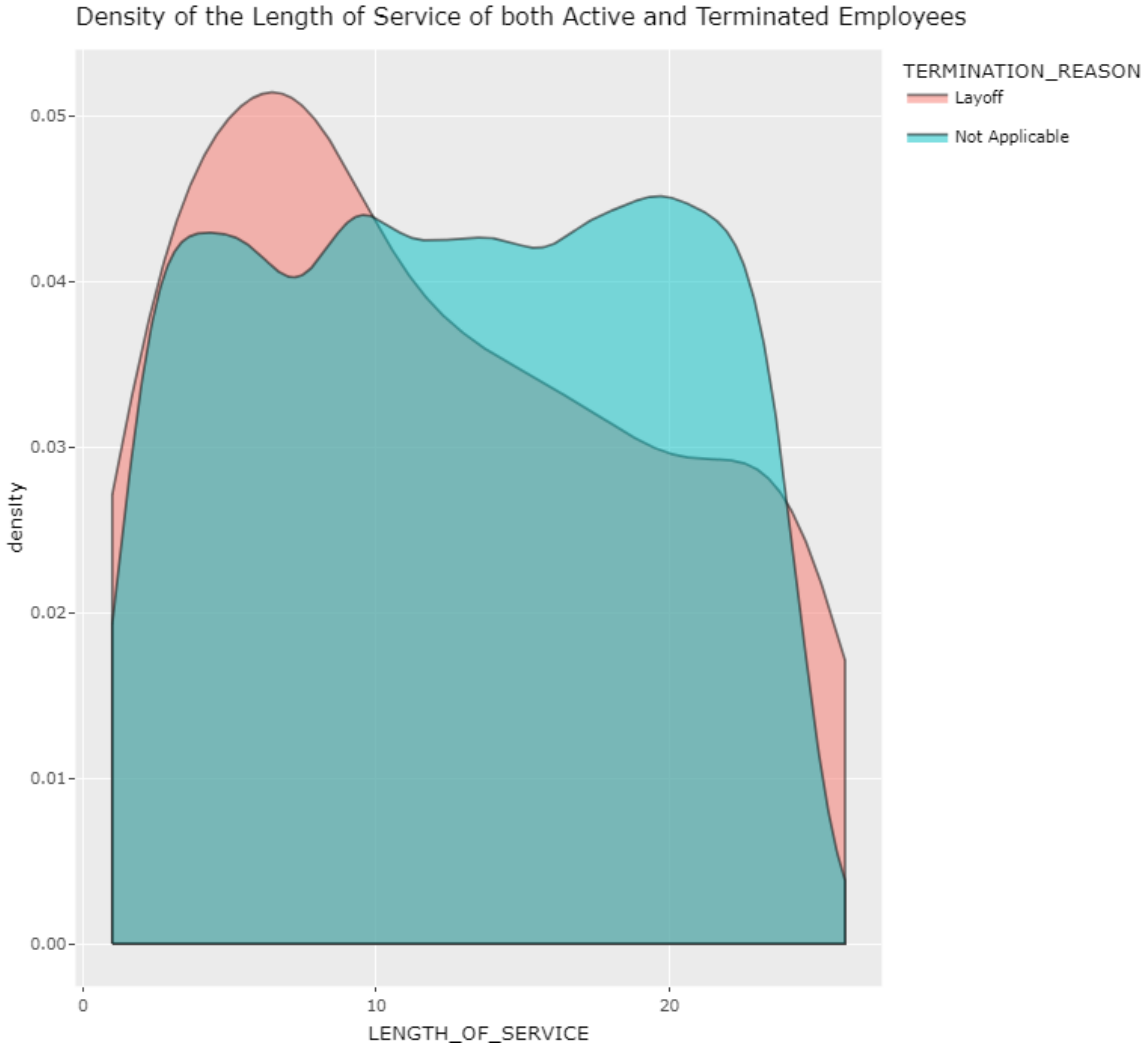
The graph shows that laid-off employees were mostly younger in age, as the graph has a higher density value for younger ages, whereas active employees are more evenly distributed in age. This demonstrates that age is one of the criteria for employee layoff, which may be due to the lack of experience of younger employees.

Analysis 6.3.6.2 - Length of service of both Active and Terminated employees

This analysis is to determine the layoff criteria of employees by comparing the length of service of both active and terminated employees.

```
los_at <- data_full %>%
  select(RECORD_YEAR, TERMINATION_REASON, LENGTH_OF_SERVICE) %>%
  filter(TERMINATION_REASON=="Layoff" | TERMINATION_REASON=="Not Applicable" & (RECORD_YEAR==2014 | RECORD_YEAR==2015))
ggplotly(ggplot(los_at, aes(x=LENGTH_OF_SERVICE, fill=TERMINATION_REASON)) +
  geom_density(alpha=.5) +
  ggtitle("Density of the Length of Service of both Active and Terminated Employees"))
```

Code: Density of the Length of Service of both Active and Terminated Employees



Graph: Density Graph for the Density of the Length of Service of both Active and Terminated Employees

From the graph, laid-off employees were mostly those with a shorter length of service, as the graph has a higher density value for short length of service, whereas active employees have a more balanced length of service. This shows that length of service is one of the criteria for employee layoff, which reasons align with Analysis 6.3.6.2 since they are strongly correlated, as indicated in Analysis 6.1.3.1.

Analysis 6.3.6.3 - Cities, Stores and Business Units of both Active and Terminated employees

This analysis is to determine the layoff criteria of employees by comparing the cities of both active and terminated employees. From the output of the cities compared, we can determine if store and the business unit is the layoff criteria of employees as well.

```
city_at <- data_full %>%
  select(RECORD_YEAR, CITY, TERMINATION_REASON) %>%
  filter(TERMINATION_REASON=="Layoff" | TERMINATION_REASON=="Not Applicable" & (RECORD_YEAR==2014 | RECORD_YEAR==2015))
ggplotly(ggplot(city_at, aes(x=RECORD_YEAR, fill=TERMINATION_REASON)) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(x="Year", y="Number of Employees", title="Cities of both Active and Terminated Employees") +
  scale_x_continuous(breaks=unique(data$RECORD_YEAR)) +
  facet_wrap(. ~ CITY))
```

Code: Cities of both Active and Terminated Employees

Graph: Bar Chart for the Cities of both Active and Terminated Employees

Since no data looks out of place nor there is any pattern, I think that the employee layoff criteria excludes the city. From this data, we also know that each store is in a different city, so I believe the employee layoff criteria excludes the Stores as well. However, since we know that only the employees in Stores were

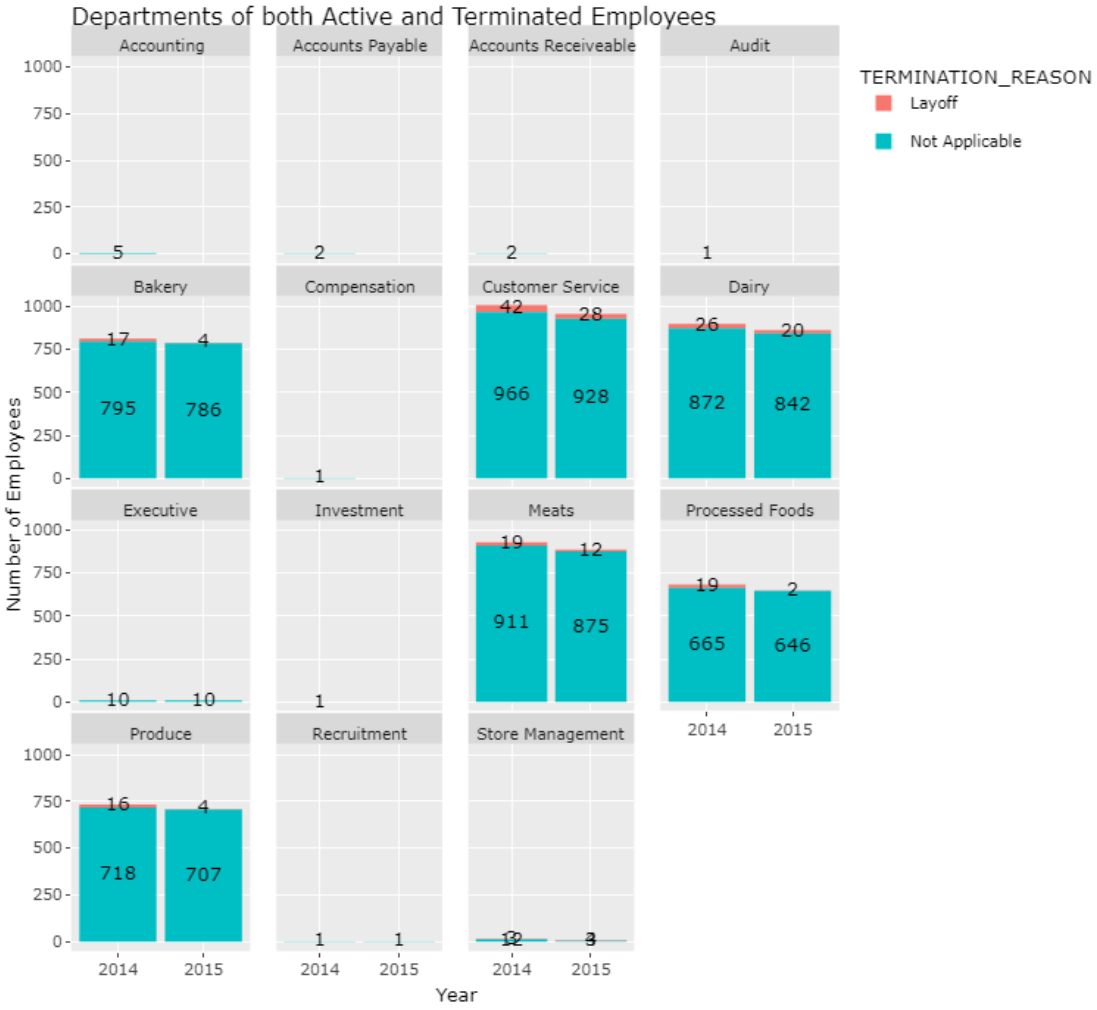
laid-off while the employees in the Head Office are not, we can deduce that the business unit is one of the criteria for employee layoff.

Analysis 6.3.6.4 - Departments of both Active and Terminated employees

This analysis is to determine the layoff criteria of employees by comparing the department of both active and terminated employees.

```
dept_at <- data_full %>%
  select(RECORD_YEAR, DEPARTMENT, TERMINATION_REASON) %>%
  filter(TERMINATION_REASON=="Layoff" | TERMINATION_REASON=="Not Applicable" & (RECORD_YEAR==2014 | RECORD_YEAR==2015))
ggplotly(ggplot(dept_at, aes(x=RECORD_YEAR, fill=TERMINATION_REASON)) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(x="Year", y="Number of Employees", title="Departments of both Active and Terminated Employees") +
  scale_x_continuous(breaks=unique(data$RECORD_YEAR)) +
  facet_wrap(. ~ DEPARTMENT))
```

Code: Department of both Active and Terminated Employees



Graph: Stacked Bar Graph for the Department of both Active and Terminated Employees

Since we know from Analysis 6.3.2.1 that Store 35 (Head Office) has no laid off employees, this means

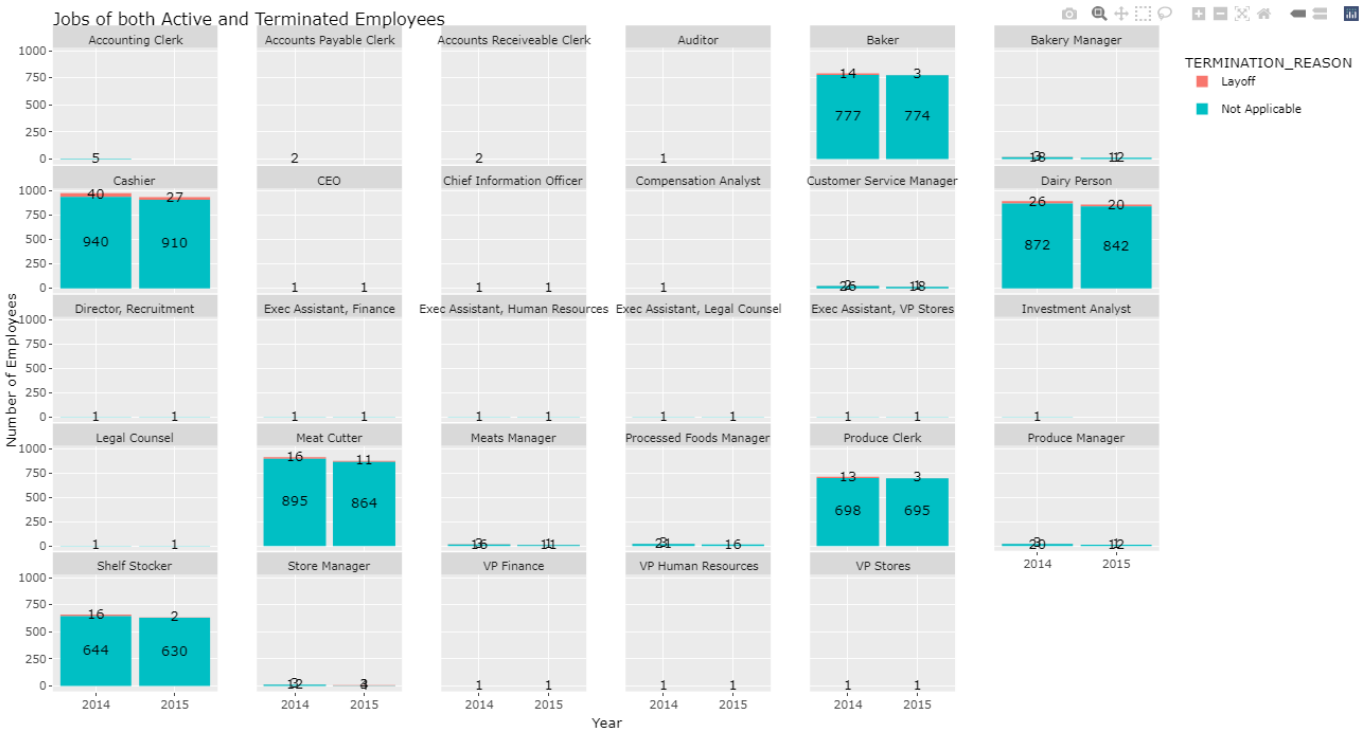
that no one in the Training, Recruitment, Legal, Labor Relations, Investment, Information Technology, HR Technology, Executive, Employee Records, Compensation, Audit, Accounts Receivable, Accounts Payable and Accounting department would have layoff employees. We can see that only the departments found in the stores which are Store Management, Produce, Processed Foods, Meats, Dairy, Customer Service and Bakery departments have layoff employees. Hence, this shows that the department is one of the criteria for employee layoff.

Analysis 6.3.6.5 - Jobs of both Active and Terminated employees

This analysis is to determine the layoff criteria of employees by comparing the jobs of both active and terminated employees.

```
job_at <- data_full %>%
  select(RECORD_YEAR, JOB_TITLE, TERMINATION_REASON) %>%
  filter(TERMINATION_REASON=="Layoff" | TERMINATION_REASON=="Not Applicable" & (RECORD_YEAR==2014 | RECORD_YEAR==2015))
ggplotly(ggplot(job_at, aes(x=RECORD_YEAR, fill=TERMINATION_REASON)) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(x="Year", y="Number of Employees", title="Jobs of both Active and Terminated Employees") +
  scale_x_continuous(breaks=unique(data$RECORD_YEAR)) +
  facet_wrap(. ~ JOB_TITLE))
```

Code: Jobs of both Active and Terminated Employees



Graph: Stacked Bar Graph for the Jobs of both Active and Terminated Employees

Since we know that from Analysis 6.3.6.4, Store 35 (Head Office) has no laid off employees, this means that no jobs in the Training, Recruitment, Legal, Labor Relations, Investment, Information Technology,

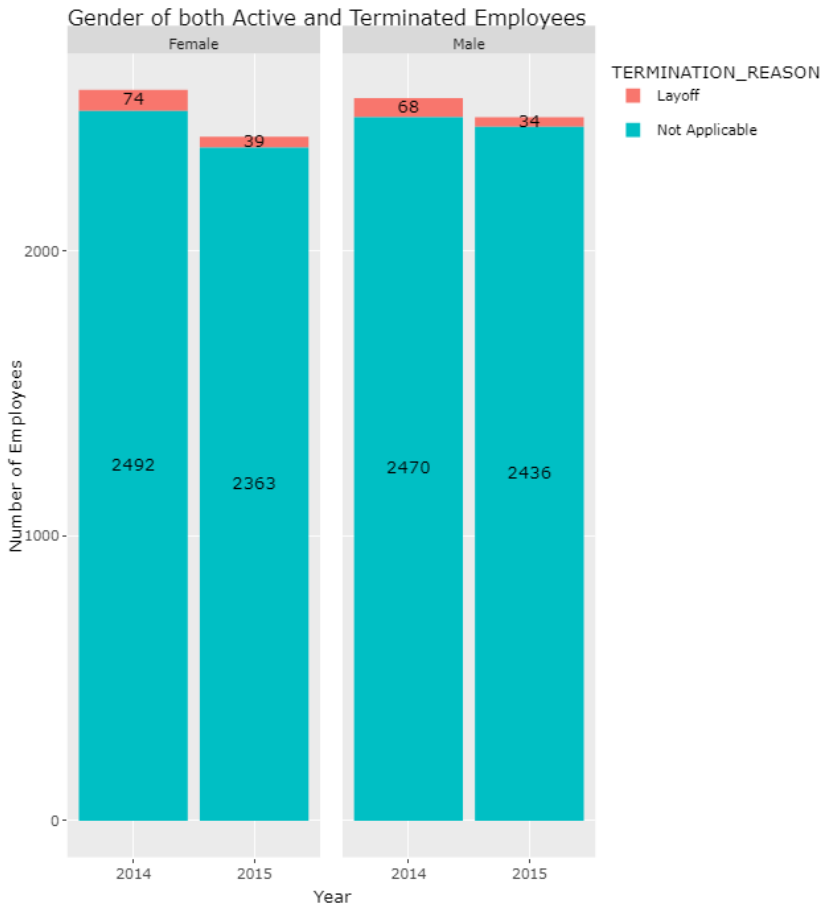
HR Technology, Executive, Employee Records, Compensation, Audit, Accounts Receivable, Accounts Payable and Accounting department would have layoff employees. We can see that only the jobs found in the stores, namely Baker, Bakery Manager, Cashier, Customer Service Manager, Dairy Person, Meat Cutter, Meats Manager, Processed Foods Manager, Produce Clerk, Produce Manager, Shelf Stocker, and Store Manager from the Store Management, Produce, Processed Foods, Meats, Dairy, Customer Service and Bakery departments have layoff employees. Hence, this shows that the job is one of the criteria for employee layoff.

Analysis 6.3.6.6 - Gender of both Active and Terminated employees

This analysis is to determine the layoff criteria of employees by comparing the gender of both active and terminated employees.

```
gender_at <- data_full %>%
  select(RECORD_YEAR, GENDER, TERMINATION_REASON) %>%
  filter(TERMINATION_REASON=="Layoff" | TERMINATION_REASON=="Not Applicable" & (RECORD_YEAR==2014 | RECORD_YEAR==2015))
ggplotly(ggplot(gender_at, aes(x=RECORD_YEAR, fill=TERMINATION_REASON)) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(x="Year", y="Number of Employees", title="Gender of both Active and Terminated Employees") +
  scale_x_continuous(breaks=unique(data$RECORD_YEAR)) +
  facet_wrap(. ~ GENDER))
```

Code: Gender of both Active and Terminated Employees



Graph: Stacked Bar Graph for the Gender of both Active and Terminated Employees

We can see from the graph that the number of male and female layoff employees is fairly balanced, so I believe the employee layoff criteria does not take gender into account.

Conclusion 3

Age

Employee layoff criteria: YES

By comparing the data with active employees, age is one of the employee criteria because laid-off employees were mostly younger in age. More young employees were fired in 2014, the most being 10 employees who are 22 years old. It is reasonable to assume that younger employees are laid off based on their years of experience, which correlates to their age. If we analyse from a generational point of view, most young employees are millennials, and according to (Miller, 2019), the reason why more millennials were laid off was due to their lack of vision and accountability in the workplace. They tend to struggle in appreciating the bigger picture and their role in the workplace, which greatly influences their working performance. Moreover, they also tend to miscommunicate, as they are sometimes overly confident and

does got give respect for the autorities. This is because jobs are often viewed as stepping stones by millennials rather than lifelong careers. This can give them the appearance of being arrogant.

On the other hand, more senior employees were fired in 2015, the most being 18 employees who are 64 years old. Since from a logical point of view, we know that a company would not layoff experienced employees, hence we need to think of why seniors may be laid off from different point of views. Logically, most older employees would be fired as their efficiency in work will greatly decrease once they become older. If we analyse from a generational point of view, most older employees are boomers, and according to (O'Donnell, 2021), the reason why more boomers were laid off was because employers wanted to cut cost by downsizing, as the more experienced one is, the higher the pay. Moreover, boomers' skills and methodical approaches to work are no longer sufficient as workplaces have become extremely adaptable. This could also be because we are approaching a digital era where technology is crucial in a company, and senior employees are more likely to be laid off in 2015 because they are less digitally literate than younger employees, especially Gen Z as they are true digital natives. (Dorsey, 2022)

Length of Service

Employee layoff criteria: YES

By comparing the data with active employees, laid-off employees were mostly those with a shorter length of service. Since the correlation between "AGE" and "LENGTH_OF_SERVICE" is a perfect positive linear correlation, we deduced that more young employees were fired during the year 2014, hence most young employees (12 employees) have a short length of service which is 2 or 7 years. On the other hand, we know more older employees were fired during the year 2015, hence most older employees (13 employees) have a long length of service which is 25 years. Hence, the reason why younger employees are laid off is mostly due to their lack of experience, as indicated by their short length of service. On the other hand, the reason why the seniors are laid off could be due to what was discussed in the "Age section" for Conclusion 3.

City

Employee layoff criteria: NO

The company fired a number of employees in certain cities in 2014, the most of which were 39 employees in Fort Nelson, and then in certain cities in 2015, the most of which were 24 employees in White Rock. The least amount of laid-off employees in 2014 is 1 employee in Blue River, and 5 employees in

Valemount in 2015. It is important that no one was laid-off from the Head Office (Store 35) in Vancouver, but some were laid-off in other stores from Vancouver. Since no data looks out of place nor there is any pattern, I think that the employee layoff criteria excludes the city.

Store

Employee layoff criteria: NO

The company fired a number of employees in certain stores in 2014, the most of which were 39 employees at Fort Nelson (Store 11), then certain stores in 2014, the most of which were 24 employees at White Rock (Store 39). The least amount of laid-off employees in 2014 is 1 employee in Blue River (Store 1), and 5 employees in Valemount (Store 34) in 2015. Since no data looks out of place nor there is any pattern, I think that the employee layoff criteria excludes the Stores.

Business Unit

Employee layoff criteria: YES

Only the employees in Stores were laid-off while the employees in the Head Office are not. This might be due to the fact that the roles of the employees in the Head Office are too significant as they make the majority of the decisions in the company, so it would be harder to find a replacement for these roles.

Department

Employee layoff criteria: YES

Since we know that the Head Office has no laid off employees, this means that no one in the Training, Recruitment, Legal, Labor Relations, Investment, Information Technology, HR Technology, Executive, Employee Records, Compensation, Audit, Accounts Receivable, Accounts Payable and Accounting department would have layoff employees. We can see that only the departments found in the stores which are Store Management, Produce, Processed Foods, Meats, Dairy, Customer Service and Bakery departments have layoff employees. For both years, a large number of employees (42 employees in 2014 and 28 employees in 2015) were laid off from the Customer Service department. This data aligns with (GlowTouch, 2021), as they mentioned that the average turnover rate for inbound customer service centres is high, at 30-45%. This might be due to the fact that the requirements to work in the Customer Service department is lower compared to other departments which require certain technical knowledge. The department that had the fewest layoffs (3 employees) in 2014 is the Store Management department,

while the department that had the fewest layoffs (2 employees) in 2015 is the Processed Foods department.

Job

Employee layoff criteria: YES

Since we know that the Head Office has no laid off employees, this means that no jobs in the Training, Recruitment, Legal, Labor Relations, Investment, Information Technology, HR Technology, Executive, Employee Records, Compensation, Audit, Accounts Receivable, Accounts Payable and Accounting department would have layoff employees. We can see that only the jobs found in the stores, namely Baker, Bakery Manager, Cashier, Customer Service Manager, Dairy Person, Meat Cutter, Meats Manager, Processed Foods Manager, Produce Clerk, Produce Manager, Shelf Stocker, and Store Manager from the Store Management, Produce, Processed Foods, Meats, Dairy, Customer Service and Bakery departments have layoff employees. Hence, this shows that the job is one of the criteria for employee layoff. A lot of employees were laid off from the Customer Service department for both years, therefore, the job that has the highest employee layoffs (40 employees in 2014 and 27 employees in 2015) is cashier, which is under the Customer Service department. Again, this might be due to the fact that the requirements to work as a cashier is lower compared to other departments which require certain technical knowledge. The job that had the least employee layoffs (2 employees) in 2014 is the Customer Service Manager from the Customer Service department, while the department that had the least employee layoff (1 employee) in 2015 is the Bakery Manager from the Bakery department, Customer Service Manager from the Customer Service department, Meats Manager from the Meats department, and Produce Manager from the Produce department.

Gender

Employee layoff criteria: NO

Overall, more female employees are getting laid off in both years than males. But since we know that the majority of employees are all female (3278 employees (52.16%)), while the rest being males (3006 employees (47.84%)), the number of male and female layoff employees seems fairly balanced. Hence, I believe the employee layoff criteria does not take gender into account.

6.4 - Question 4: What is the Nature of Resignation?

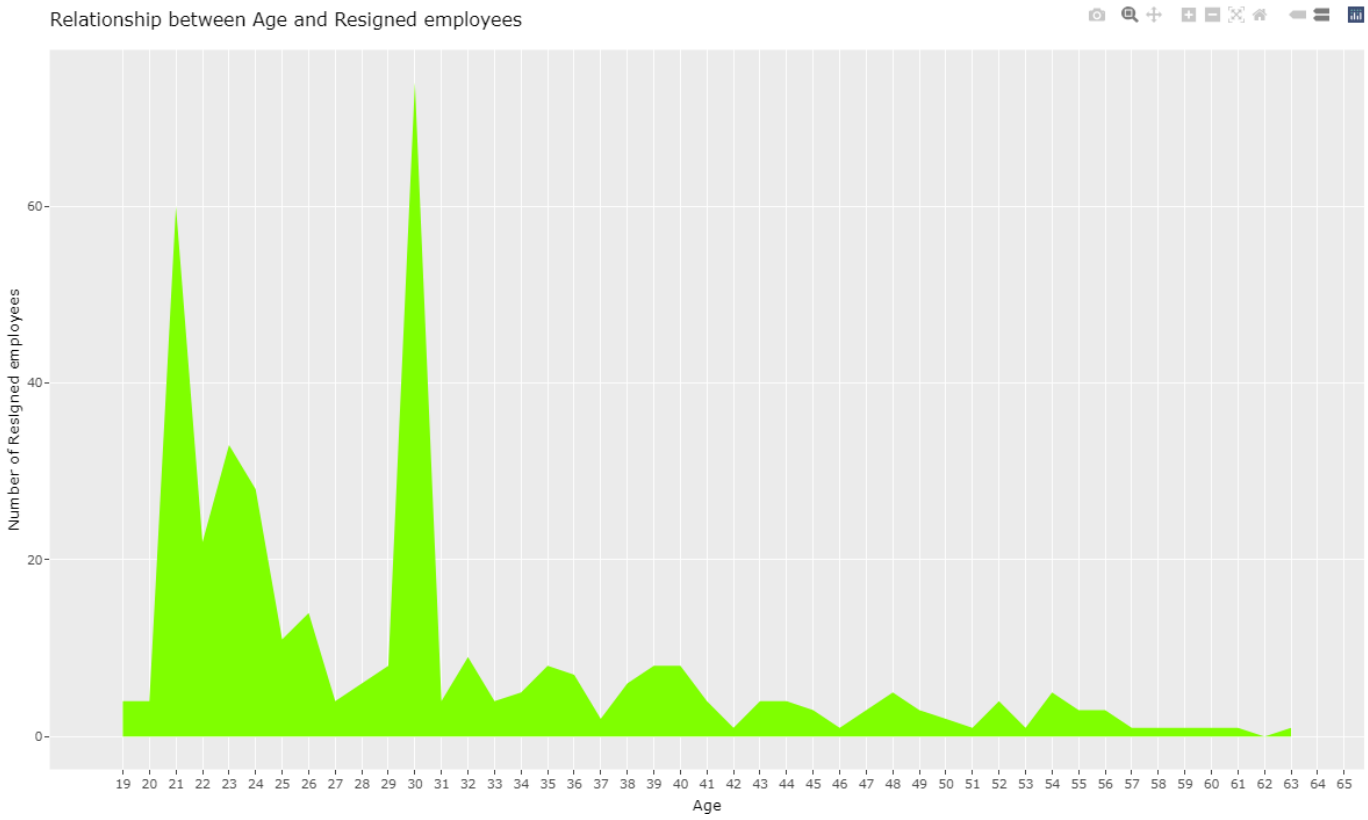
Among all termination reasons, the second majority reason of terminated employees was through resignation (382 employees). Hence, this question is to find out the nature of employee resignation.

Analysis 6.4.1 - Relationship between Age and Resigned employees

This analysis is to determine the relationship between Age and Resigned employees.

```
ggplotly(data %>% filter(TERMINATION_REASON == "Resignation") %>%
  select(RECORD_YEAR, AGE) %>%
  ggplot(aes(x=AGE)) +
  geom_area(aes(y = ..count..), stat ="bin", binwidth=1, fill="chartreuse") +
  labs(colour="YEAR", x="Age", y="Number of Resigned Employees", title="Relationship between Age and Resigned Employees") +
  scale_x_continuous(breaks=unique(data$AGE)))
```

Code: Area Graph for the Relationship between Age and Resigned Employees



Graph: Area Graph for the Relationship between Age and Resigned Employees

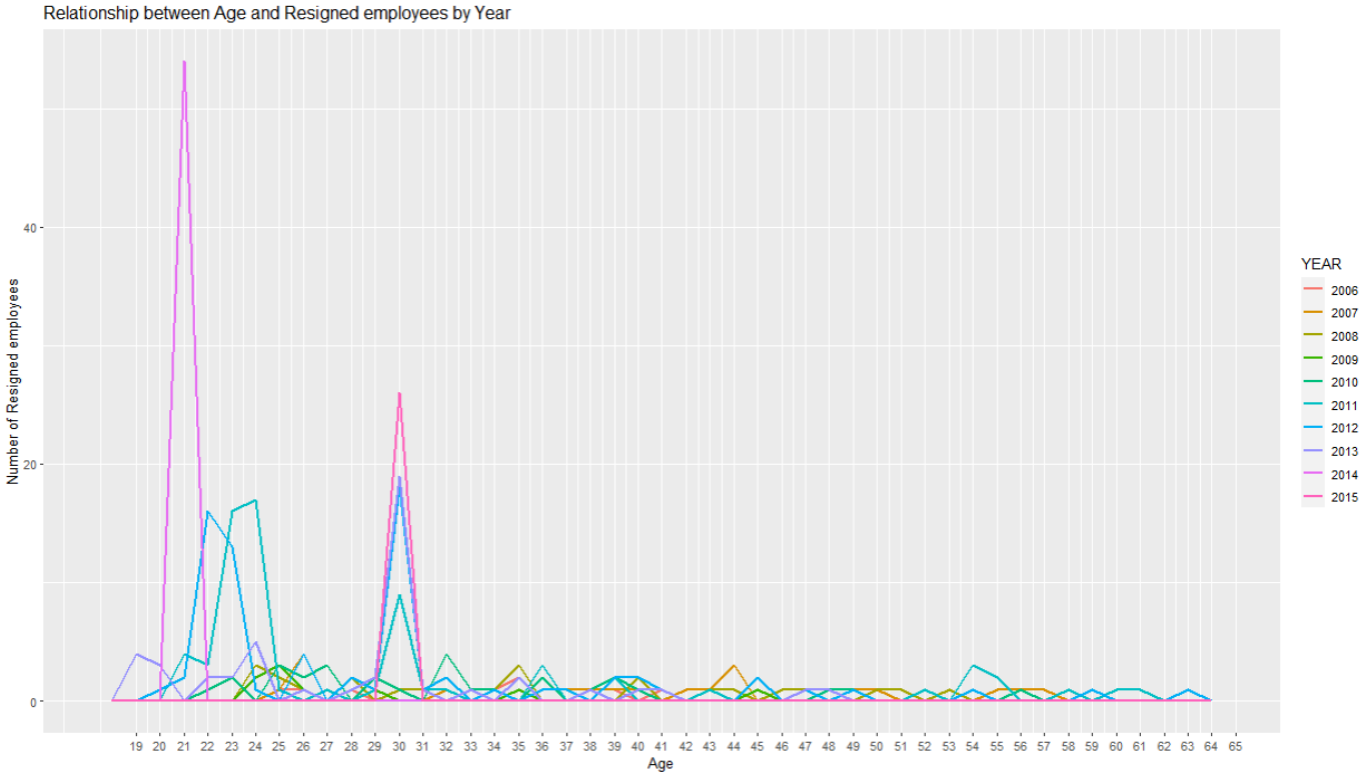
The graph shows that the most employees resigned at age 30 (74 employees) and age 21 (60 employees), while the least employees resigned at an older age, which is age 42, 51, 53, 57, 58,59, 61 and 63 (1 employee).

Analysis 6.4.1.1 - Relationship between Age and Resigned employees by year

This analysis is to determine the relationship between Age and Resigned employees by year.

```
ggplotly(data %>% filter(TERMINATION_REASON == "Resignation") %>%
  select(RECORD_YEAR, AGE) %>%
  ggplot(aes(x=AGE, colour=factor(RECORD_YEAR))) +
  geom_freqpoly(size=1, binwidth=1) +
  labs(colour="YEAR", x="Age", y="Number of Resigned Employees", title="Relationship between Age and Resigned Employees by Year") +
  scale_x_continuous(breaks=unique(data$AGE)))
```

Code: Frequency Polygon Graph for the Relationship between Age and Resigned Employees by Year



Graph: Frequency Polygon Graph for the Relationship between Age and Resigned Employees by Year

According to this graph, most young employees (54 employees) resigned in 2014, especially in age 21. A lot of young employees from the age of 22 to 24 has also resigned in 2011 and 2012. We can also see that a lot of 30 years old employees resigned on different years, which is 9 employees in 2011, 18 employees in 2012, 19 employees in 2013, and 26 employees in 2015.

Analysis 6.4.1.2 - Correlation between Age and Resigned Employees by Year

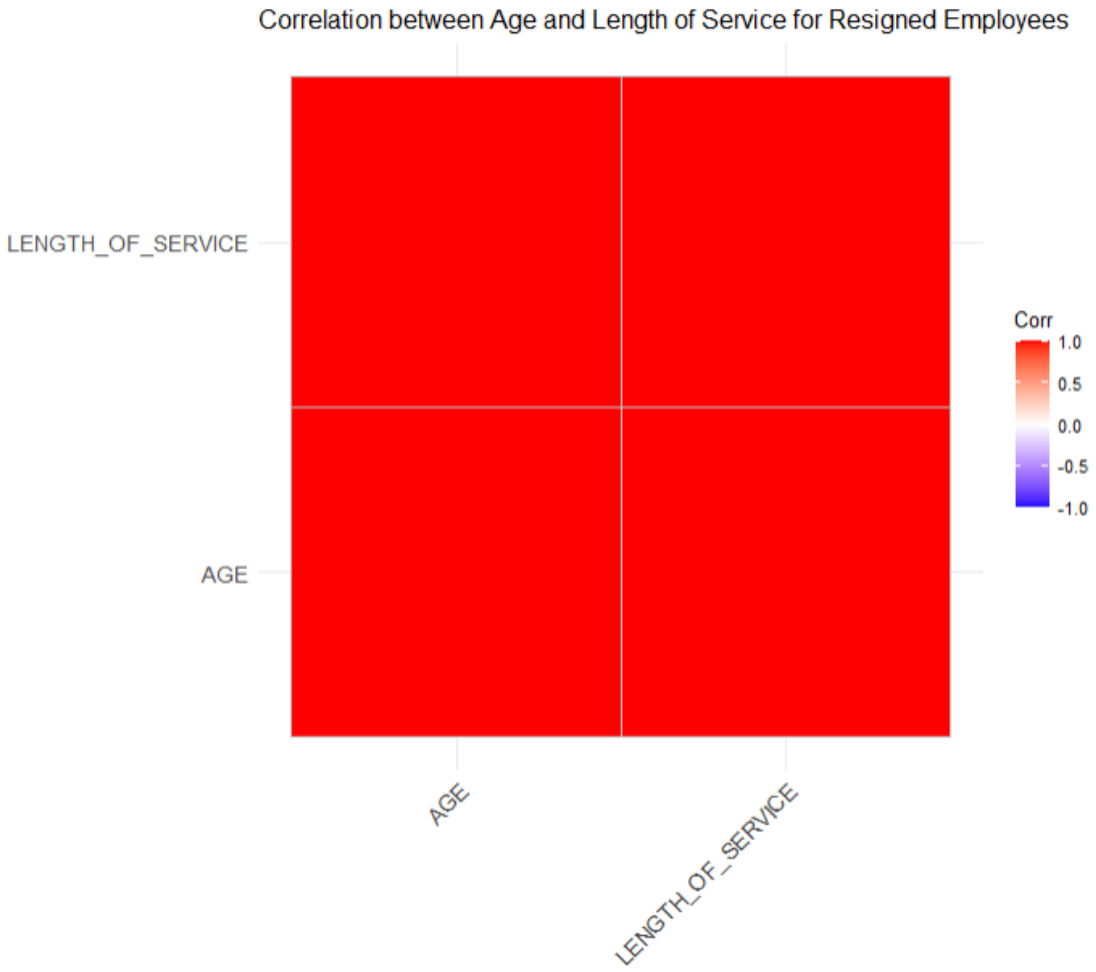
This analysis is to determine if there is any correlation between the age and length of service for resigned employees.

```
age_los <- data %>% filter(TERMINATION_REASON=="Resignation") %>% select(AGE, LENGTH_OF_SERVICE)
cor_matrix <- round(cor(age_los), 1)
cor_matrix
ggcorrplot(cor_matrix) +
  labs(title="Correlation between Age and Length of Service for Resigned Employees")
```

Code: Correlation Matrix between Age and Length of Service

```
                    AGE LENGTH_OF_SERVICE
AGE                   1                 1
LENGTH_OF_SERVICE     1                 1
```

Output: Correlation Matrix between Age and Length of Service

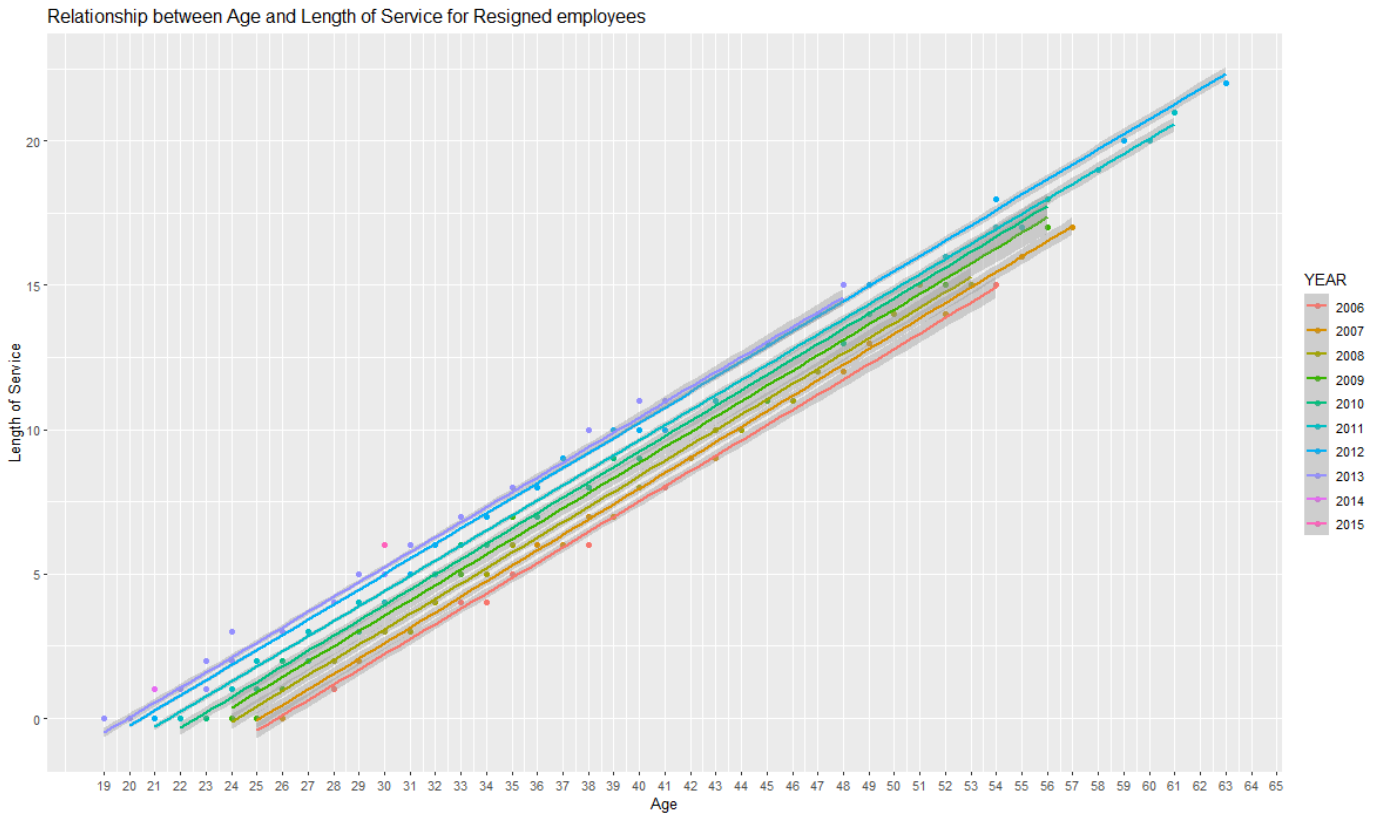


Graph: Correlation Matrix between Age and Length of Service

The Pearson correlation coefficient, which measures the linear association between two variables, is used to quantify the relationship between "AGE" and "LENGTH OF SERVICE." From the output, the correlation coefficients along the diagonal of the table are all equal to 1 because each variable is perfectly correlated with itself. These cells aren't useful for interpretation, so we should focus on the other cells in the matrix. However, the correlation between "AGE" and "LENGTH_OF_SERVICE" is also a 1, indicating that the two variables have a perfectly positive linear correlation. (Zach, 2022)

```
data %>% filter(TERMINATION_REASON == "Resignation") %>%
  select(RECORD_YEAR, AGE, LENGTH_OF_SERVICE) %>%
  ggplot(aes(x=AGE, y=LENGTH_OF_SERVICE, colour=factor(RECORD_YEAR))) +
  geom_point() +
  labs(colour="YEAR", x="Age", y="Length of Service", title="Relationship between Age and Length of Service for Resigned Employees") +
  scale_x_continuous(breaks=unique(data$AGE)) +
  stat_smooth(method=lm)
```

Code: Relationship between Age and Length of Service for Resigned Employees



Graph: Point Graph for Relationship between Age and Length of Service for Resigned Employees

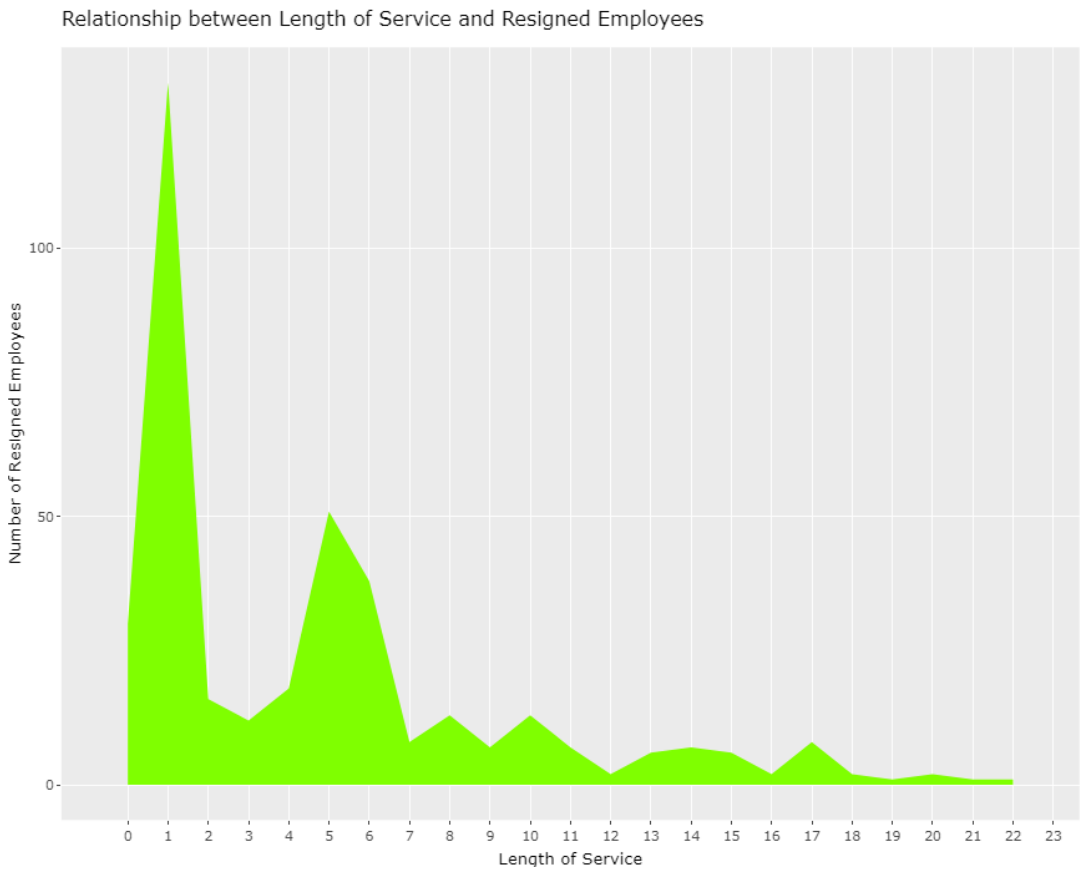
As indicated by both graphs, there is a strong linear relationship between the two variables. As a result, older age is strongly associated with greater length of service. (Zach, 2022)

Analysis 6.4.1.3 - Relationship between Length of Service and Resigned employees

This analysis is to determine the relationship between the length of service and Resigned employees.

```
ggplotly(data %>% filter(TERMINATION_REASON == "Resignation") %>%
  select(RECORD_YEAR, LENGTH_OF_SERVICE) %>%
  ggplot(aes(x=LENGTH_OF_SERVICE)) +
  geom_area(aes(y = ..count..), stat ="bin", binwidth=1, fill="chartreuse") +
  labs(colour="YEAR", x="Length of Service", y="Number of Resigned Employees", title="Relationship between Length of Service and Resigned Employees") +
  scale_x_continuous(breaks=unique(data$LENGTH_OF_SERVICE)))
```

Code: Relationship between Length of Service for Resigned Employees



Graph: Area Graph for the Relationship between Length of Service and Resigned Employees

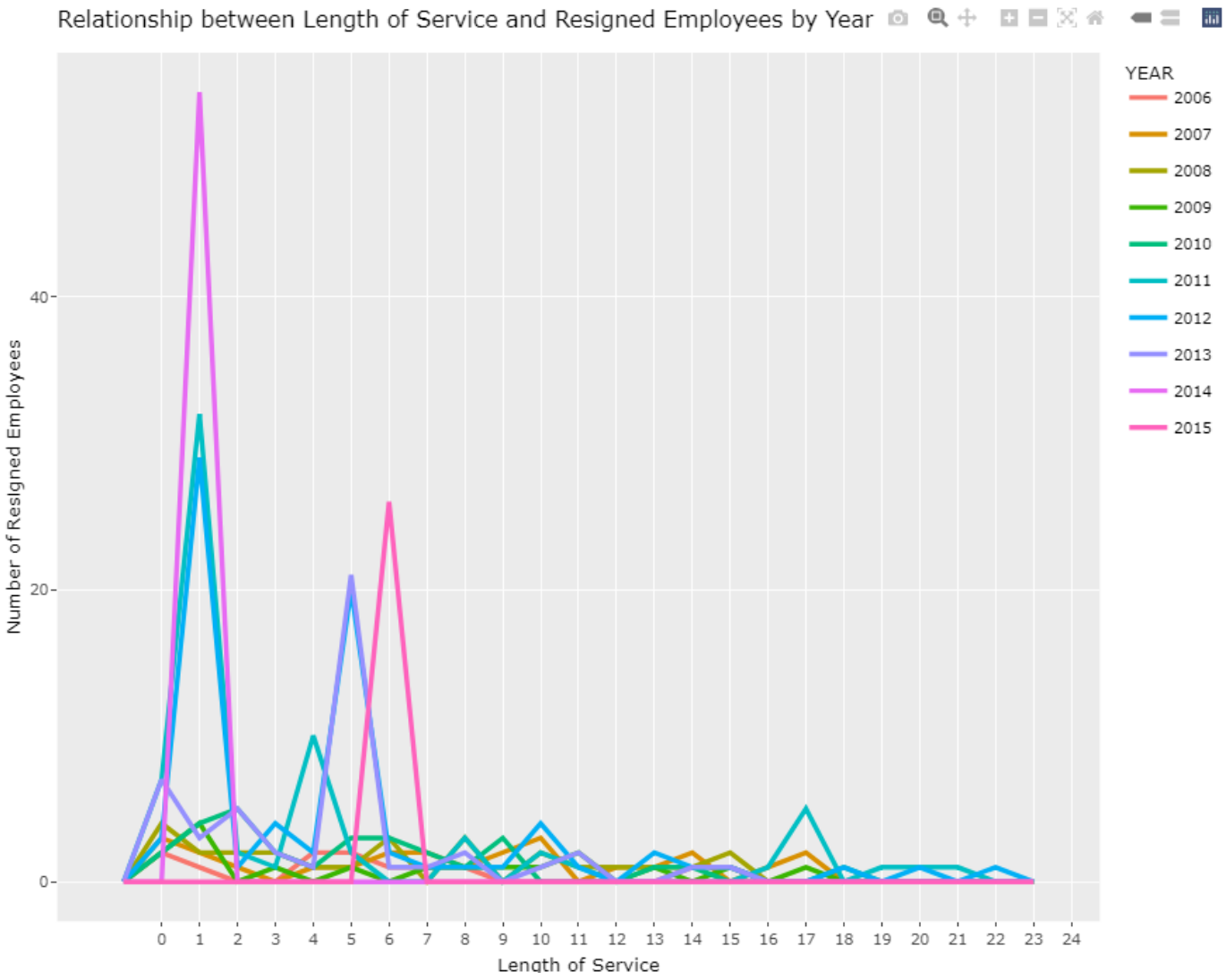
Based on the graph, 131 employees resigned after 1 year of service, and the longest years of service is 22 with only 1 employee. We already knew from Analysis 6.4.1.1 that most young employees (54 employees) resigned in 2014, especially in age 21. A lot of young employees from the age of 22 to 24 has also resigned in 2011 and 2012. We can also see that a lot of 30 years old employees resigned on different years. Because the correlation between "AGE" and "LENGTH OF SERVICE" in Analysis 6.3.1.2 is a perfect positive linear correlation, we can conclude most young employees have less than 5 years of experience in the company, and most young employees (30 employees) resigned without even reaching 1 year of service. We can also conclude that employees that are 30 years old have at least 5 years of experience in the company.

Analysis 6.4.1.4 - Relationship between Length of Service and Resigned employees by year

This analysis is to determine the relationship between the length of service and Resigned employees by year.

```
ggplotly(data %>% filter(TERMINATION_REASON == "Resignation") %>%
  select(RECORD_YEAR, LENGTH_OF_SERVICE) %>%
  ggplot(aes(x=LENGTH_OF_SERVICE, colour=factor(RECORD_YEAR))) +
  geom_freqpoly(size=1, binwidth=1) +
  labs(colour="YEAR", x="Length of Service", y="Number of Resigned Employees", title="Relationship between Length of Service and Resigned Employees by Year") +
  scale_x_continuous(breaks=unique(data$LENGTH_OF_SERVICE)))
```

Code: Relationship between Length of Service and Resigned Employees by Year



Graph: Frequency Polygon Graph for the Relationship between Length of Service and Resigned Employees by Year

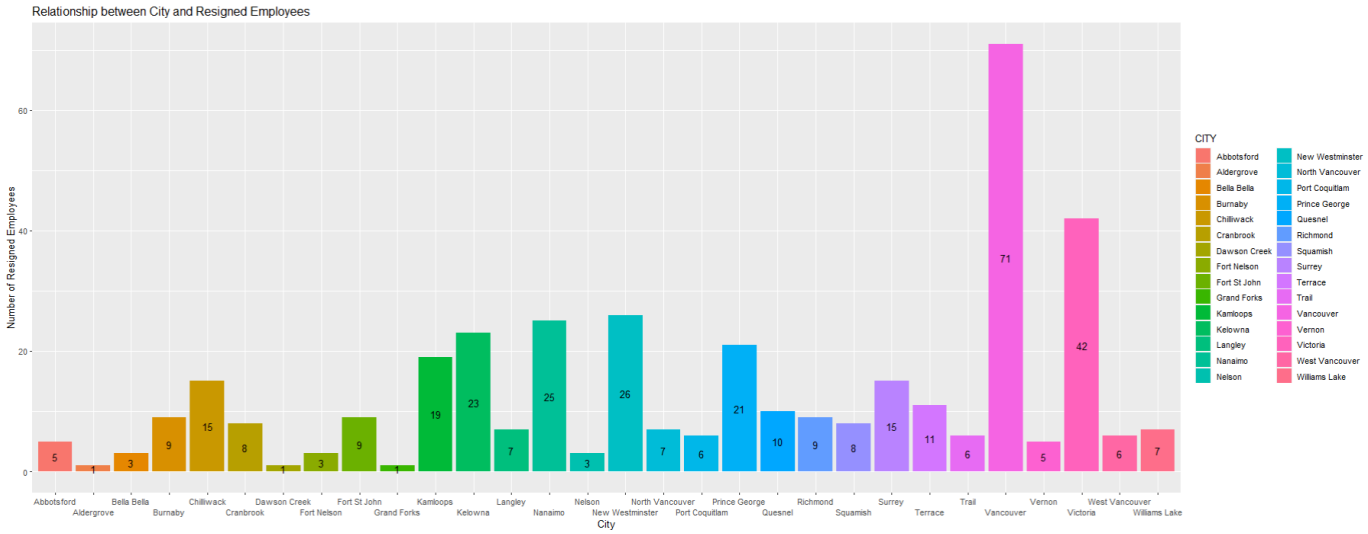
Since the correlation between "AGE" and "LENGTH_OF_SERVICE" is a perfect positive linear correlation from Analysis 6.4.1.2, this graph would definitely resemble Analysis 6.4.1.1. We know that most young employees resigned in 2011, 2012 and 2014, hence they have a shorter length of service which is less than 5 years. We can also conclude that 30 years old employees resigned on different years in 2011, 2012, 2013, and 2015, hence they have a longer length of service which is at least 5 years.

Analysis 6.4.2 - Relationship between City and Resigned employees

This analysis is to determine the relationship between the city and Resigned employees.

```
data %>% filter(TERMINATION_REASON == "Resignation") %>%
  select(CITY) %>%
  ggplot(aes(x=CITY, fill=CITY)) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(x="City", y="Number of Resigned Employees", title="Relationship between City and Resigned Employees") +
  scale_x_discrete(guide=guide_axis(n.dodge = 2))
```

Code: Relationship between City and Resigned Employees



Graph: Bar Graph of the Relationship between City and Resigned Employees

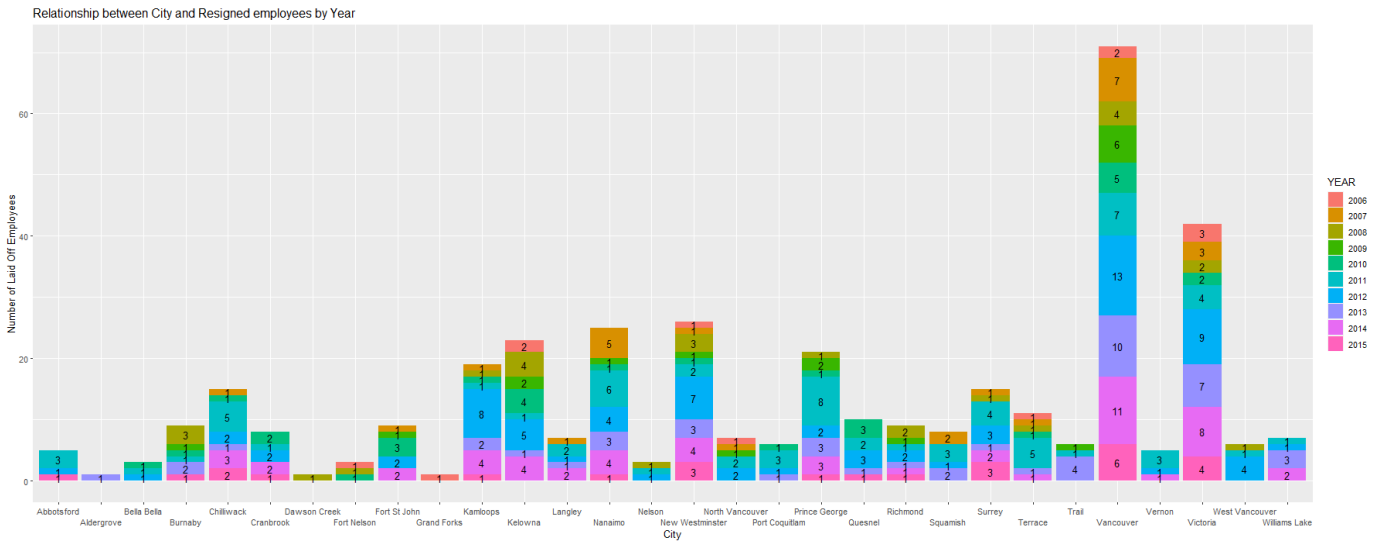
From the graph, we could see that a lot of people resigned in Vancouver (71 employees) and Victoria (42 employees), which is reasonable since they have the most stores in their location. The city that had the least resigned employees is Aldergrove, Dawson Creek and Grand Forks with only 1 employee.

Analysis 6.4.2.1 - Relationship between City and Resigned employees by Year

This analysis is to determine the relationship between the city and Resigned employees by year.

```
data %>% filter(TERMINATION_REASON == "Resignation") %>%
  select(RECORD_YEAR, CITY) %>%
  ggplot(aes(x=CITY, fill=factor(RECORD_YEAR))) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="YEAR", x="City", y="Number of Resigned Employees", title="Relationship between City and Resigned Employees by Year") +
  scale_x_discrete(guide=guide_axis(n.dodge = 2))
```

Code: Relationship between City and Resigned Employees by Year



Graph: Stacked Bar Graph of the Relationship between City and Resigned Employees by Year

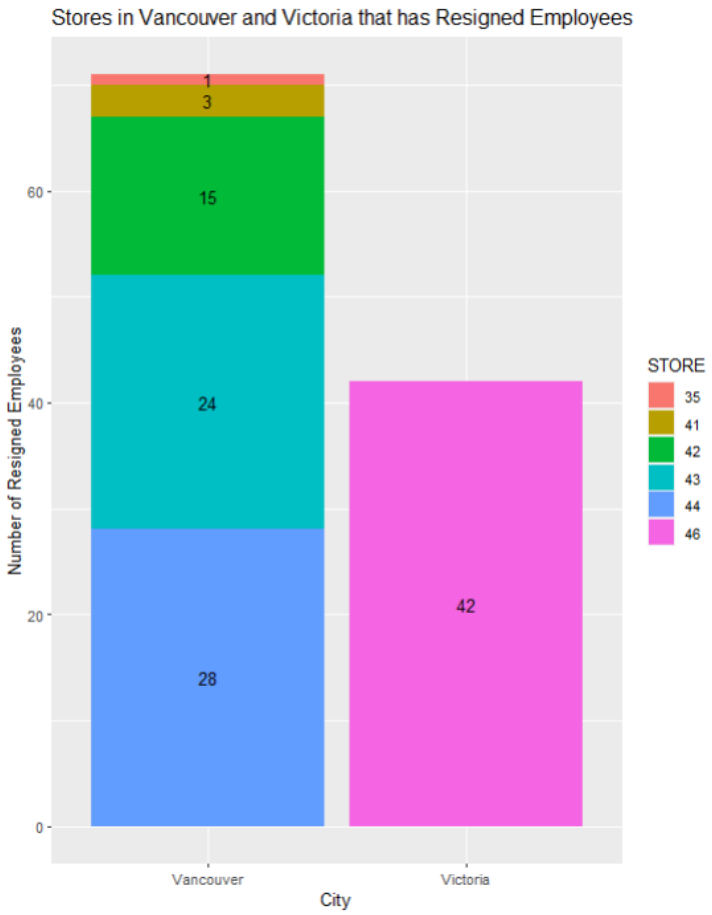
From the graph, we can see that most employees resigned during 2012 especially in Vancouver and Victoria. Since Vancouver and Victoria are in the list, we need to check which Stores in Vancouver and Victoria has resigned employees.

Analysis 6.4.2.2 - Stores in Vancouver and Victoria that has Resigned employees

This analysis is to determine which stores in Vancouver and Victoria have Resigned employees.

```
data %>% filter(TERMINATION_REASON == "Resignation", (CITY=="Vancouver" | CITY=="Victoria")) %>%
  select(STORE, CITY) %>%
  ggplot(aes(x=CITY, fill=STORE)) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="STORE", x="City", y="Number of Resigned Employees", title="Stores in Vancouver and Victoria that has Resigned Employees")
```

Code: Stores in Vancouver and Victoria that has Resigned Employees



Graph: Stacked Bar Graph of Stores in Vancouver and Victoria that has Resigned Employees

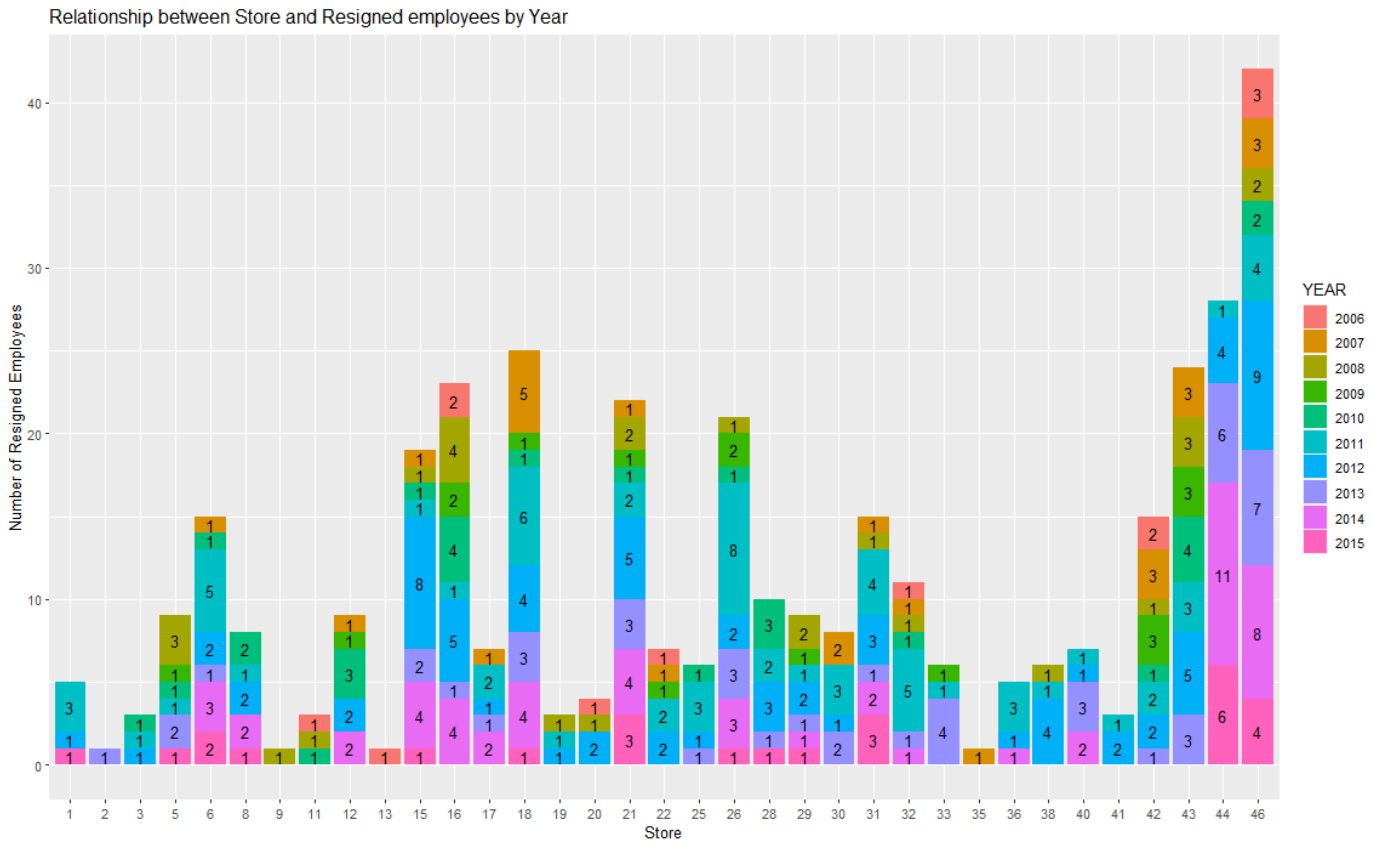
From the output, we know that the most employees that resigned (28 employees) in Vancouver came from Store 44, while the employees that resigned (42 employees) in Victoria came from Store 46. In the Head Office (Store 35), only 1 employee resigned.

Analysis 6.4.2.3 - Relationship between Store and Resigned employees by Year

This analysis is to determine the relationship between Store and Resigned employees by year.

```
data %>% filter(TERMINATION_REASON == "Resignation") %>%
  select(RECORD_YEAR, STORE) %>%
  ggplot(aes(x=STORE, fill=factor(RECORD_YEAR))) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="YEAR", x="Store", y="Number of Resigned Employees", title="Relationship between Store and Resigned Employees by Year")
```

Code: Relationship between Store and Resigned employees by Year



Graph: Stacked Bar Graph of the Relationship between Store and Resigned employees by Year

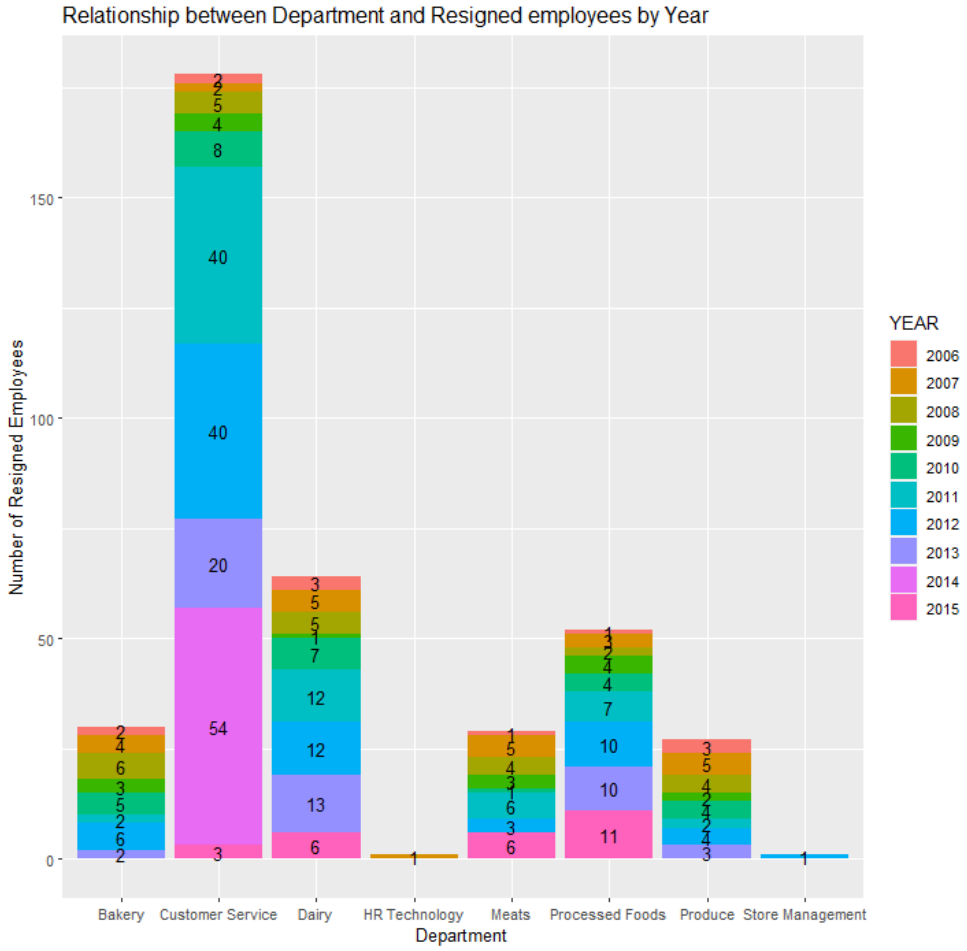
This graph aligns with Analysis 6.4.2. From the graph, we could see that Store 46 from Victoria has the most employees that resigned, most employees (9 employees) resigned during 2012. This is followed by Store 44 from Vancouver, most employees (11 employees) resigned during 2014. The least amount of resigned employees in every year per store is 1 employee.

Analysis 6.4.3 - Relationship between Department and Resigned employees

This analysis is to determine the relationship between Department and Resigned employees.

```
data %>% filter(TERMINATION_REASON == "Resignation") %>%
  select(RECORD_YEAR, DEPARTMENT) %>%
  ggplot(aes(x=DEPARTMENT, fill=factor(RECORD_YEAR))) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="YEAR", x="Department", y="Number of Resigned Employees", title="Relationship between Department and Resigned Employees by Year")
```

Code: Relationship between Department and Resigned Employees by Year



Graph: Stacked Bar Graph of the Relationship between Department and Resigned Employees by Year

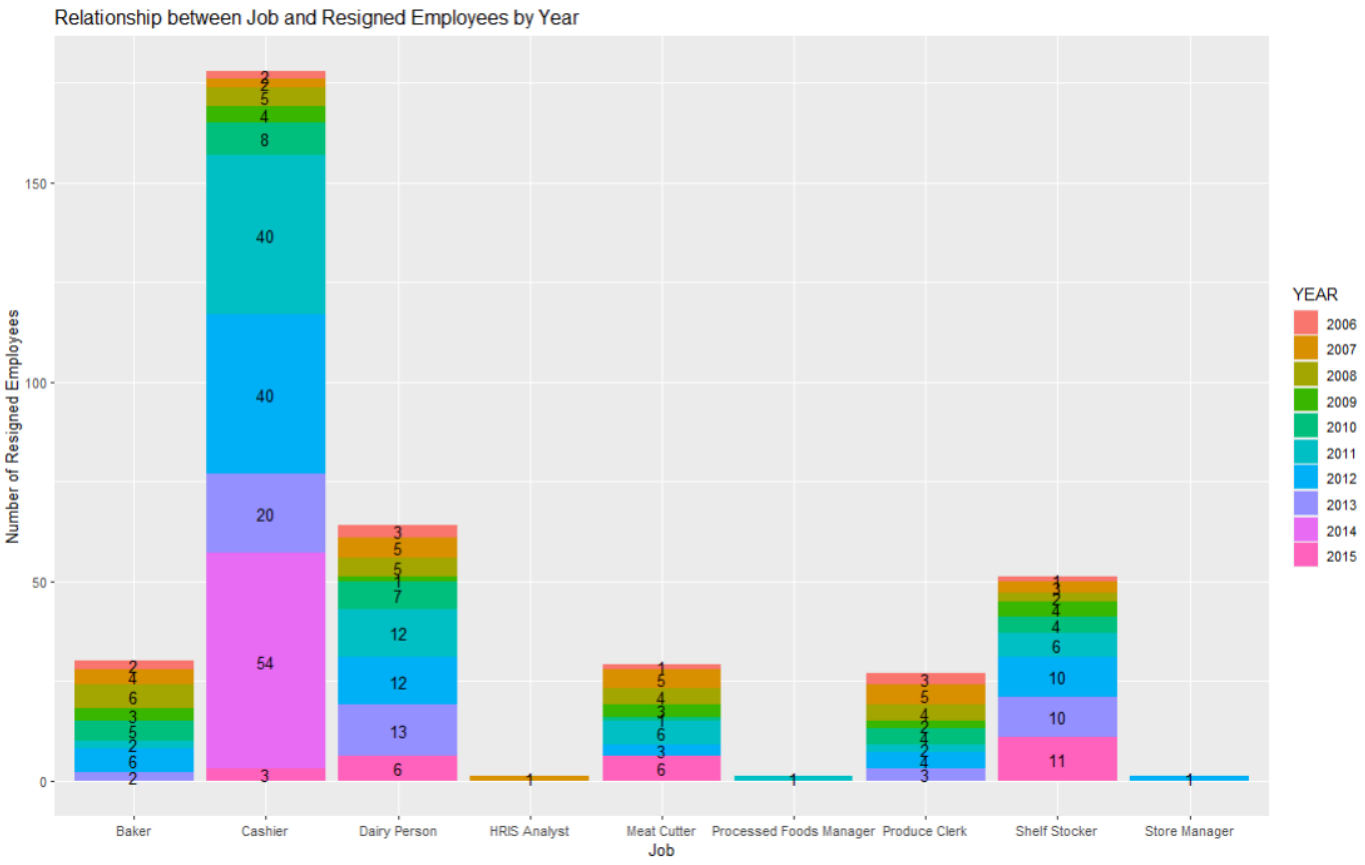
From 2011 to 2014, a large number of employees (40 employees in 2011, 40 employees in 2012, 20 employees in 2013, and 54 employees in 2014) resigned from the Customer Service department. The departments that had the fewest resignations (1 employees) are the HR Technology department and Store Management department.

Analysis 6.4.4 - Relationship between Job and Resigned employees

This analysis is to determine the relationship between Job and Resigned employees.

```
data %>% filter(TERMINATION_REASON == "Resignation") %>%
  select(RECORD_YEAR, JOB_TITLE) %>%
  ggplot(aes(x=JOB_TITLE, fill=factor(RECORD_YEAR))) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="YEAR", x="Job", y="Number of Resigned Employees", title="Relationship between Job and Resigned Employees by Year")
```

Code: Relationship between Job and Resigned Employees by Year



Graph: Stacked Bar Chart of the Relationship between Job and Resigned Employees by Year

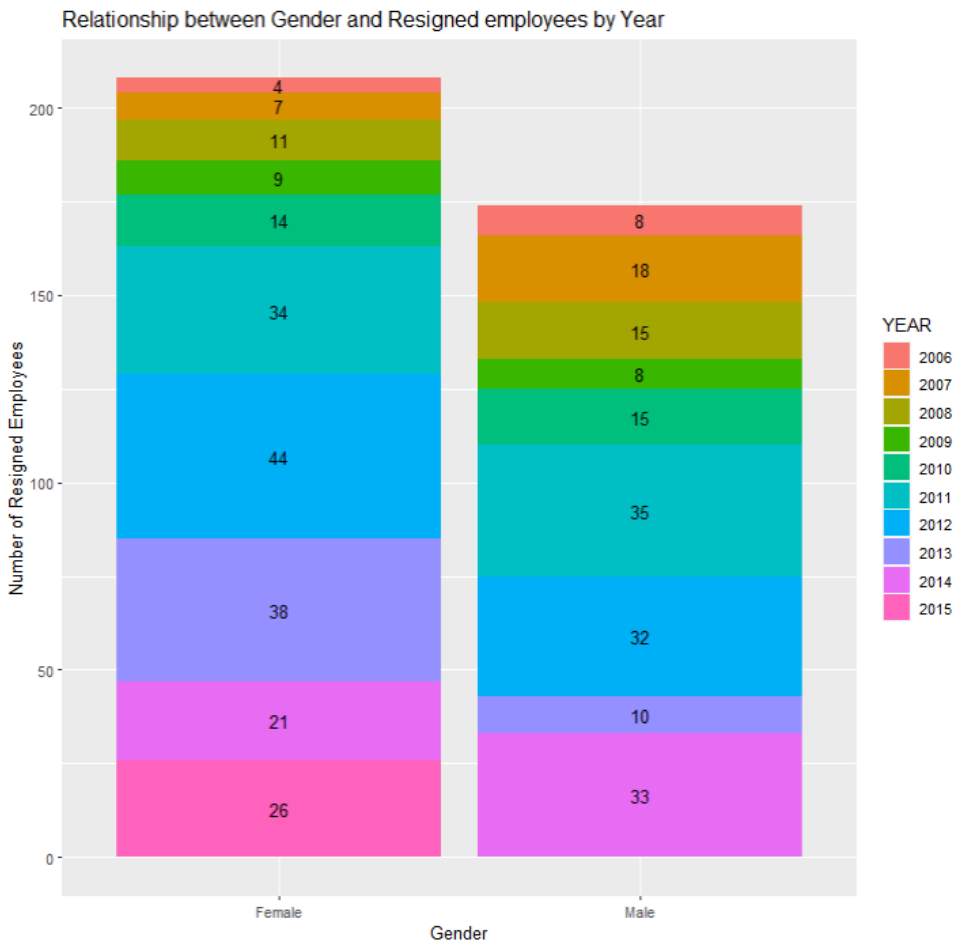
Since we know that from Analysis 6.4.3, a lot of employees resigned from the Customer Service department from 2011 to 2014, therefore, the job that has the highest employee resignment (40 employees in 2011, 40 employees in 2012, 20 employees in 2013, and 54 employees in 2014) is cashier, which is under the Customer Service department. The departments that had the fewest resignations (1 employees) are the HRIS Analyst under the HR Technology department, Processed Foods Manager under the Processed Foods department, and Store Manager from the Store Management department.

Analysis 6.4.5 - Relationship between Gender and Resigned employees

This analysis is to determine the relationship between Gender and Resigned employees by year.

```
data %>% filter(TERMINATION_REASON == "Resignation") %>%
  select(RECORD_YEAR, GENDER) %>%
  ggplot(aes(x=GENDER, fill=factor(RECORD_YEAR))) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="YEAR", x="Gender", y="Number of Resigned Employees", title="Relationship between Gender and Resigned Employees by Year")
```

Code: Relationship between Gender and Resigned Employees by Year



Graph: Stacked Bar Chart of the Relationship between Gender and Resigned Employees by Year

Overall, more female employees are resigning than males. The only information that stood out was in 2015, only females employee resigned. But since we know from Section 5.13 that the majority of employees are all female (3278 employees (52.16%)), while the rest being males (3006 employees (47.84%)), the number of male and female layoff employees seems fairly balanced.

Conclusion 4

Age

Most employees resigned at age 30 (74 employees) in year 2011 (9 employees), 2012 (18 employees), 2013 (19 employees) and 2015 (26 employees). Most young employees (54 employees) resigned in 2014, especially in age 21. A lot of young employees from the age of 22 to 24 has also resigned in 2011 and 2012. The least employees that resigned resigned at an older age, which is age 42, 51, 53, 57, 58, 59, 61 and 63 (1 employee). If we analyse from a generational point of view, a lot of employees that resigned are millennials. Hence, this resigning behaviour can be explained due to the trend of job hopping among

millennials. According to (Adkins, 2019), millennials are the most likely generation to switch jobs. This may be due the fact that they are the least engaged generation in the workplace, the salary provided, company culture and more. However according to the same source, many millennials may not want to change jobs, but their employers aren't providing compelling reasons for them to stay. The reality is that they simply want a job that feels meaningful to them – and they will keep looking until they find it. Moreover, the reason why majority of employees resign at age 30 is because the average age of marriage in Canada is 30 (Institute for Family Studies, 2022).

Length of Service

Since the correlation between "AGE" and "LENGTH_OF_SERVICE" is a perfect positive linear correlation, we deduced that most young employees have less than 5 years of experience in the company, and most young employees (30 employees) resigned without even reaching 1 year of service. Employees that are 30 years old have a longer length of service which is at least 5 years. To put it into picture, 131 employees resigned after 1 year of service, and the longest years of service is 22 with only 1 employee. The reason why the length of service is so short among the employees could be due to what was discussed in the "Age section" for Conclusion 4.

City

A lot of people resigned in Vancouver (71 employees) and Victoria (42 employees), mostly in 2012, which is reasonable since they have the most stores in their location. The city that had the least resigned employees is Aldergrove, Dawson Creek and Grand Forks with only 1 employee.

Store

The least amount of resigned employees in every year per store is 1 employee. Most employees that resigned (42 employees) in Victoria came from Store 46, and most of them (9 employees) resigned during 2012. This is followed by 28 resigned employees in Vancouver that came from Store 44, where most employees (11 employees) resigned during 2014. Compared to the Head Office (Store 35) in Vancouver, only 1 employee resigned. This indicates that the company might have been focusing on the wellbeing of the employees in the Head Office too much to the point that other stores in Vancouver was neglected as a result, hence the big difference of the resignment amount between Store 35 and other stores in Vancouver.

Department

From 2011 to 2014, a large number of employees (40 employees in 2011, 40 employees in 2012, 20 employees in 2013, and 54 employees in 2014) resigned from the Customer Service department. I think that this may be due to the stress in working under the Customer Service department, as mentioned by (CommBox, 2020) as they argue that customer service representatives are frequently under extreme time constraints, and they has to deal with complex problems and rude customers constantly. But I think that this could also be due to the job hopping trend of millennials, as discussed in the "Age" section for Conclusion 4. The departments that had the fewest resignations (1 employees) are the HR Technology department and Store Management department.

Job

A lot of employees resigned from the Customer Service department from 2011 to 2014, therefore, the job that has the highest employee resignment (40 employees in 2011, 40 employees in 2012, 20 employees in 2013, and 54 employees in 2014) is cashier. This may be due to the stress and job hopping trend mentioned in the "Department" section for Conclusion 4. To add on, this might be caused by the nature of the job itself as cashier is indeed a boring job (Moran, 2022), which might not please the behaviour of millennials. The departments that had the fewest resignations (1 employees) are the HRIS Analyst under the HR Technology department, Processed Foods Manager under the Processed Foods department, and Store Manager from the Store Management department.

Gender

Overall, more female employees are resigning than males. The only information that stood out was in 2015, only females employee resigned. But since we know from Section 5.13 that the majority of employees are all female (3278 employees (52.16%)), while the rest being males (3006 employees (47.84%)), the number of male and female layoff employees seems fairly balanced.

6.5 - Question 5: What is the Nature of Retirement?

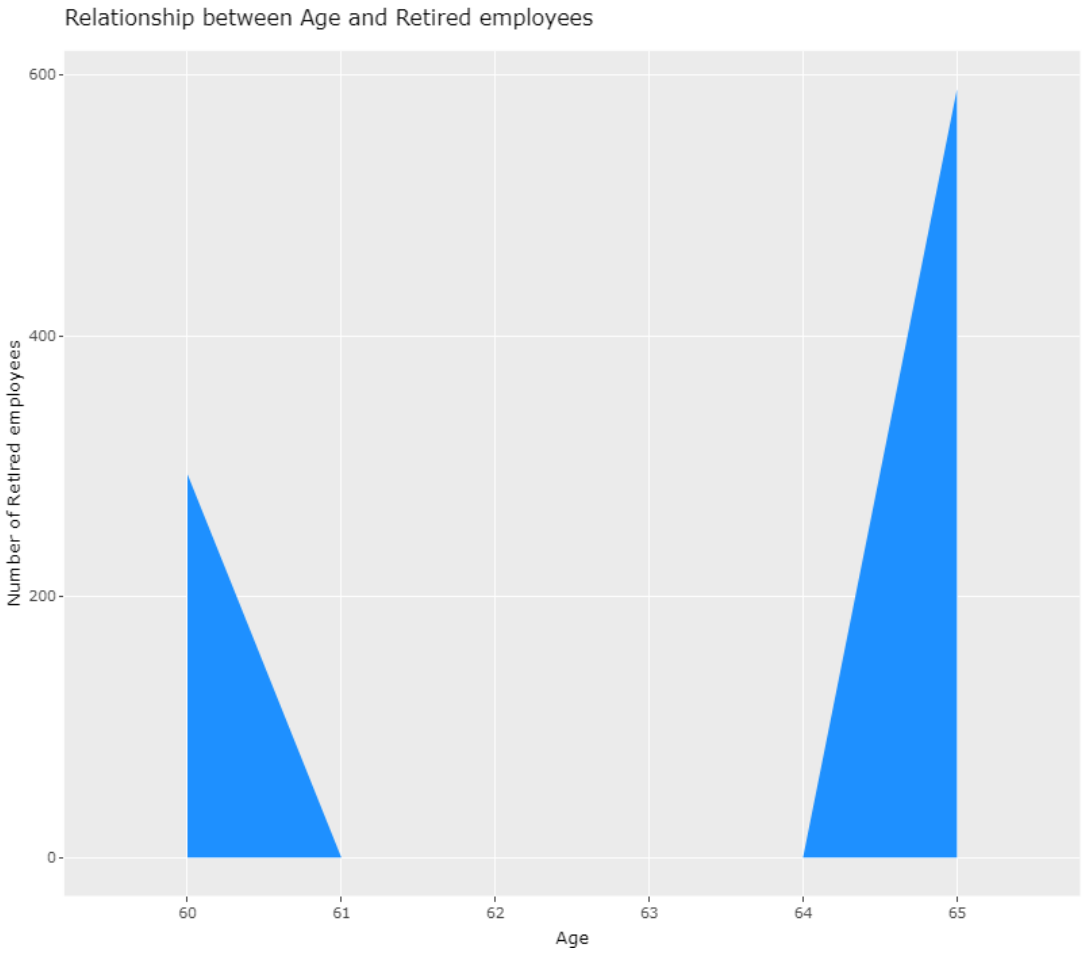
Among all termination reasons, the majority reason of terminated employees was through retirement (883 employees). Hence, this question is to find out the nature of employee resignation.

Analysis 6.5.1 - Relationship between Age and Retired employees

This analysis is to determine the relationship between Age and Retired employees.

```
ggplotly(data %>% filter(TERMINATION_REASON == "Retirement") %>%
  select(RECORD_YEAR, AGE) %>%
  ggplot(aes(x=AGE)) +
  geom_area(aes(y = ..count..), stat ="bin", binwidth=1, fill="dodgerblue") +
  labs(colour="YEAR", x="Age", y="Number of Retired Employees", title="Relationship between Age and Retired Employees") +
  scale_x_continuous(breaks=unique(data$AGE)))
```

Code: Area Graph for the Relationship between Age and Retired Employees



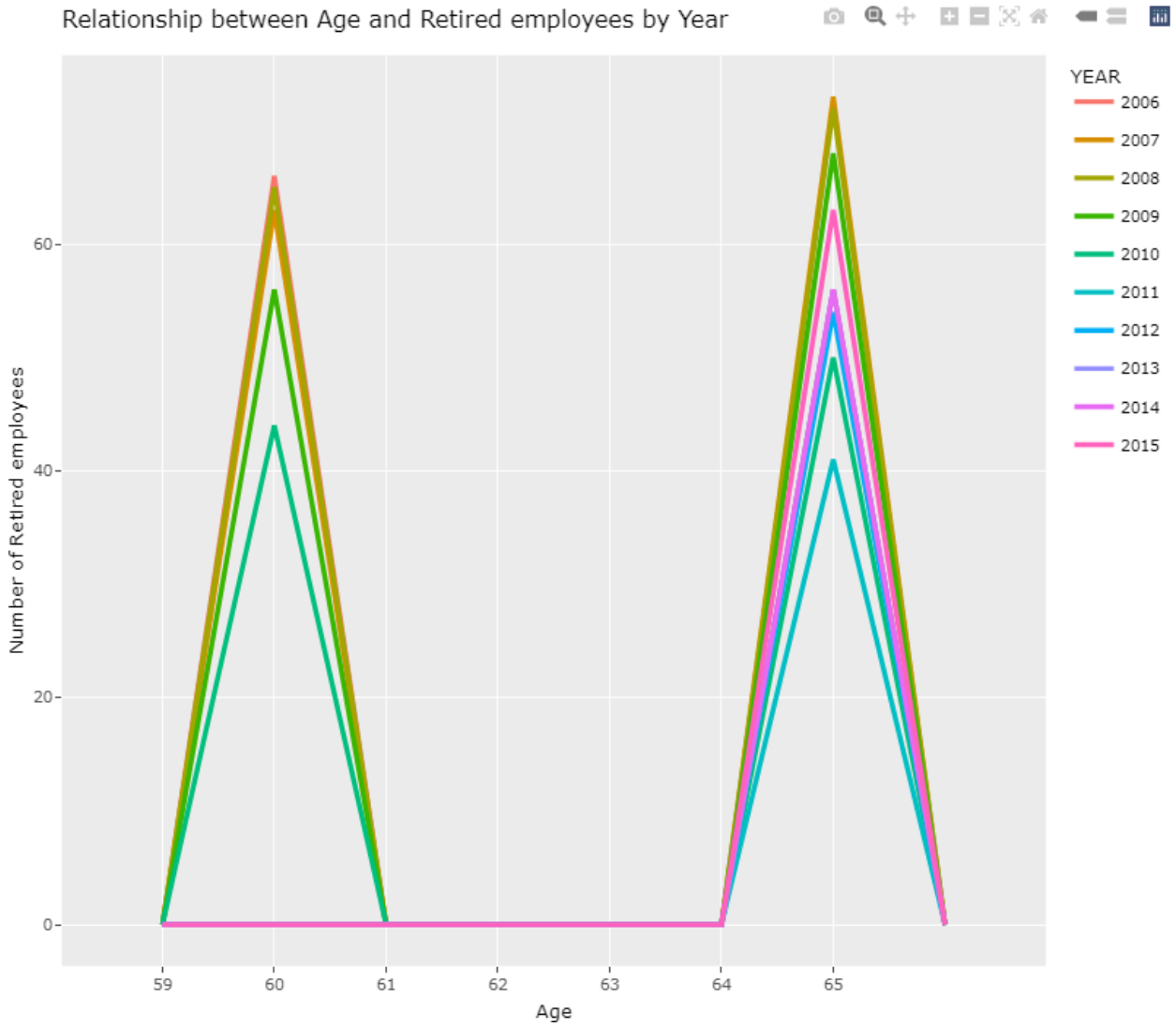
Graph: Area Graph for the Relationship between Age and Retired Employees

The graph shows that the employees only retire when they are 60 years old or 65 years old. The most employees retired at age 65 (589 employees), while the least employees retired at age 60 (294 employee).

Analysis 6.5.1.1 - Relationship between Age and Retired employees by year

This analysis is to determine the relationship between Age and Retired employees by year.

Code: Relationship between Age and Retired Employees by Year



Graph: Frequency Polygon Graph for the Relationship between Age and Retired Employees by Year

According to this graph, most employees that retired at age 65 retired in 2007 (73 employees), which was closely followed by the ones that retired in 2006 (72 employees). Most employees that retired at age 60 retired in 2006 (66 employees), which was closely followed by the ones that retired in 2008 (65 employees).

Analysis 6.5.1.2 - Correlation between Age and Retired Employees by Year

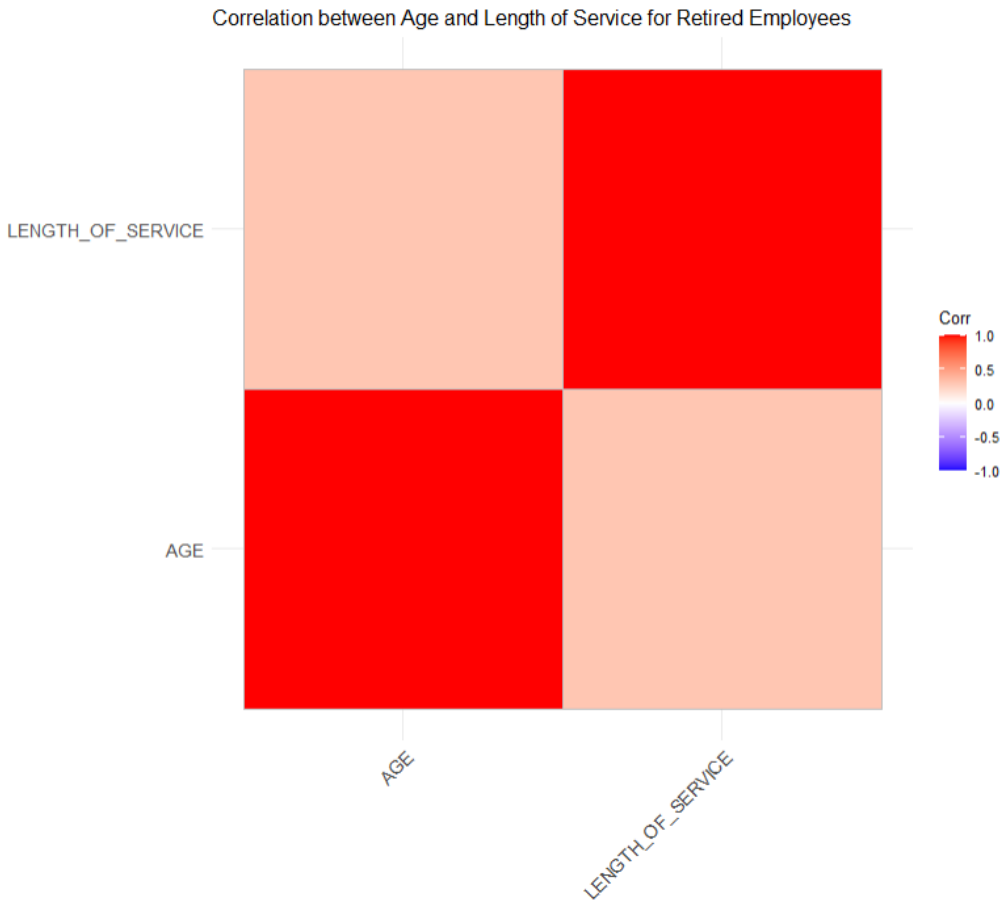
This analysis is to determine if there is any correlation between the age and length of service for Retired employees.

```
age_los <- data %>% filter(TERMINATION_REASON=="Retirement") %>% select(AGE, LENGTH_OF_SERVICE)
cor_matrix <- round(cor(age_los), 1)
cor_matrix
ggcorrplot(cor_matrix) +
  labs(title="Correlation between Age and Length of Service for Retired Employees")
```

Code: Correlation Matrix between Age and Length of Service

```
> cor_matrix
                  AGE LENGTH_OF_SERVICE
AGE               1.0               0.3
LENGTH_OF_SERVICE 0.3               1.0
```

Output: Correlation Matrix between Age and Length of Service

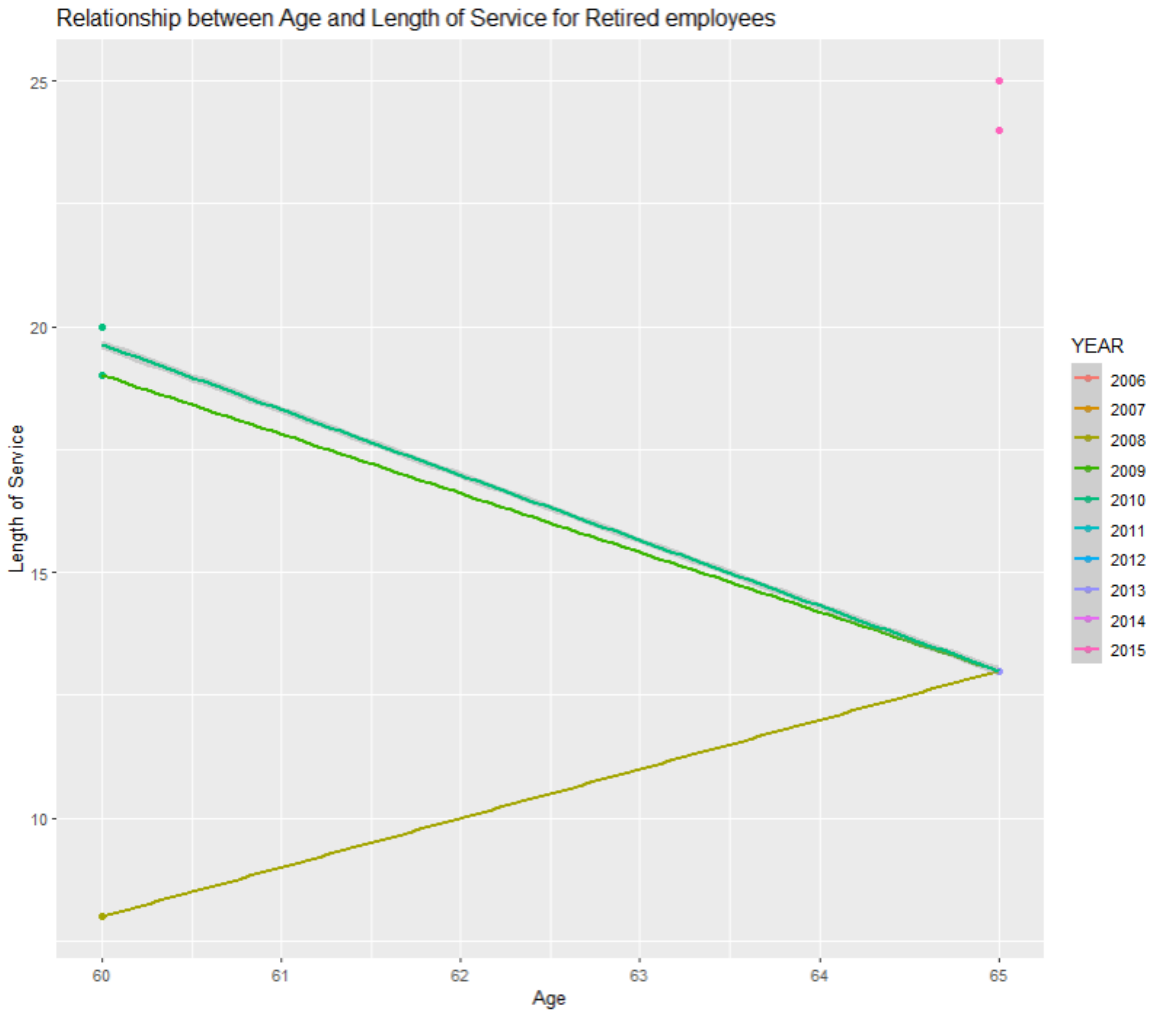


Graph: Correlation Matrix between Age and Length of Service

The Pearson correlation coefficient, which measures the linear association between two variables, is used to quantify the relationship between "AGE" and "LENGTH OF SERVICE." From the output, the correlation coefficients along the diagonal of the table are all equal to 1 because each variable is perfectly correlated with itself. These cells aren't useful for interpretation, so we should focus on the other cells in the matrix. However, the correlation between "AGE" and "LENGTH_OF_SERVICE" is 0.3, indicating that the two variables have a weak positive linear correlation. (Zach, 2022)

```
ggplotly(data %>% filter(TERMINATION_REASON == "Retirement") %>%
  select(RECORD_YEAR, AGE, LENGTH_OF_SERVICE) %>%
  ggplot(aes(x=AGE, y=LENGTH_OF_SERVICE, colour=factor(RECORD_YEAR))) +
  geom_point() +
  labs(colour="YEAR", x="Age", y="Length of Service", title="Relationship between Age and Length of Service for Retired Employees") +
  scale_x_continuous(breaks=unique(data$AGE)) +
  stat_smooth(method=lm))
```

Code: Relationship between Age and Length of Service for Retired Employees



Graph: Point Graph for Relationship between Age and Length of Service for Retired Employees

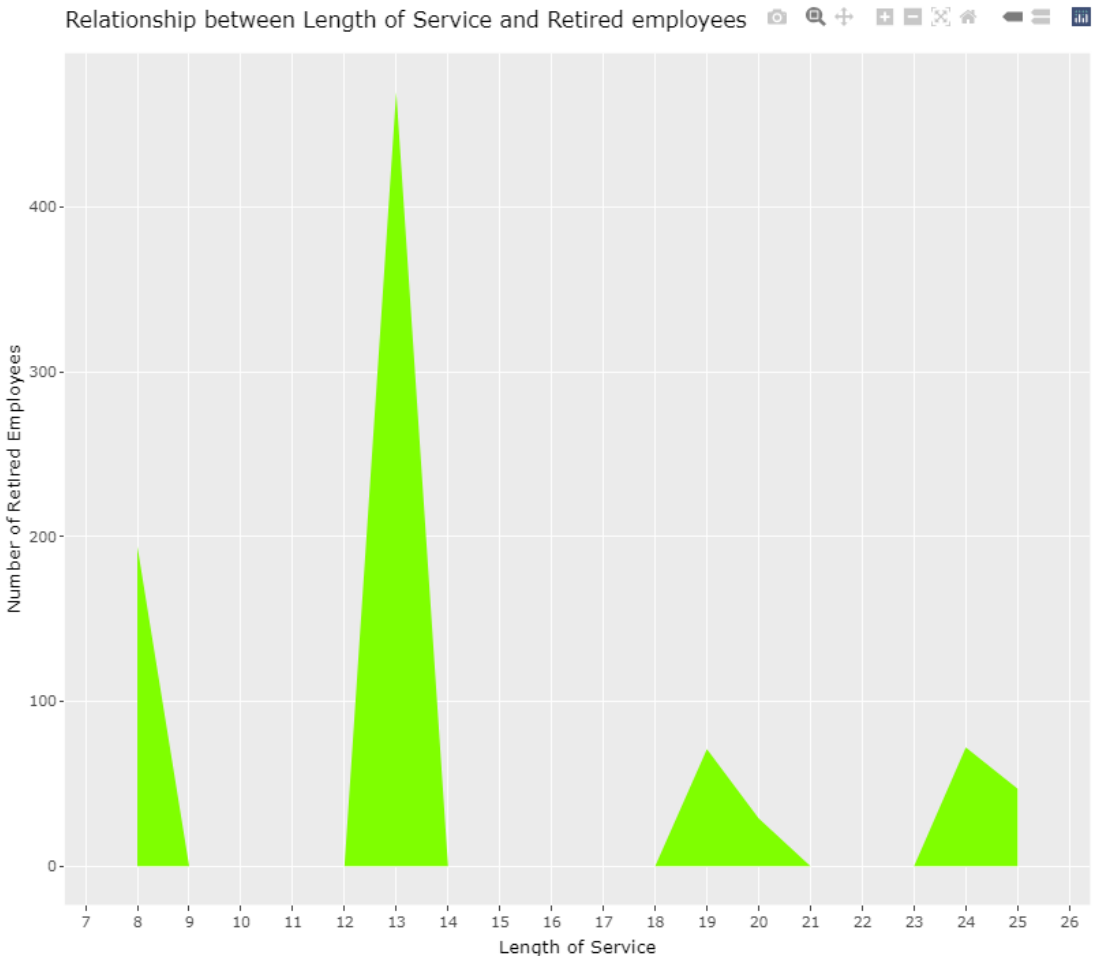
As indicated by both graphs, there is very little association between the age and length of service for retired employees. In 2009 and 2010, employees that retired on age 65 did not had as much length of service compared to the ones that retired on age 60. In 2006, 2007, 2008 the employees that retired on age 65 have more length of service compared to the ones that retired on age 60.

Analysis 6.5.2 - Relationship between Length of Service and Retired employees

This analysis is to determine the relationship between the length of service and Retired employees.

```
ggplotly(data %>% filter(TERMINATION_REASON == "Retirement") %>%
  select(RECORD_YEAR, LENGTH_OF_SERVICE) %>%
  ggplot(aes(x=LENGTH_OF_SERVICE)) +
  geom_area(aes(y = ..count..), stat ="bin", binwidth=1, fill="chartreuse") +
  labs(colour="YEAR", x="Length of Service", y="Number of Retired Employees", title="Relationship between Length of Service and Retired Employees") +
  scale_x_continuous(breaks=unique(data$LENGTH_OF_SERVICE)))
```

Code: Relationship between Length of Service for Retired Employees



Graph: Area Graph for the Relationship between Length of Service and Retired Employees

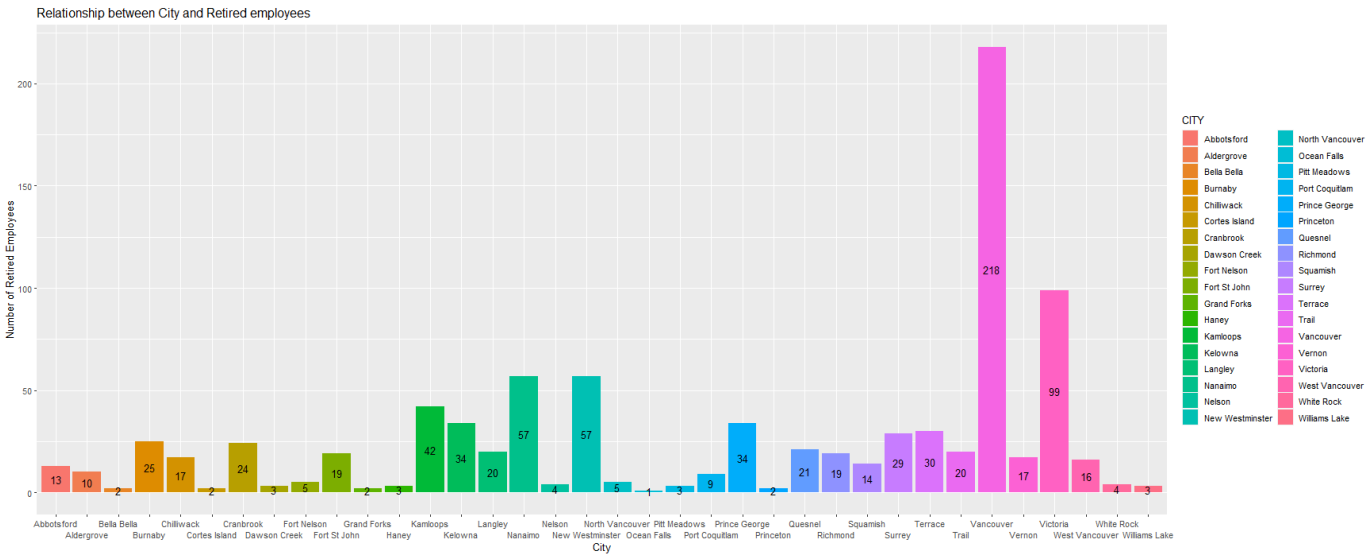
Based on the graph, the most employees (470 employees) retired after working for 13 years. 47 employees retired after working for the longest length of service which is 25 years, while 194 employees retired after working for the shortest length of service which is 8 years.

Analysis 6.5.3 - Relationship between City and Retired employees

This analysis is to determine the relationship between the city and Retired employees.

```
data %>% filter(TERMINATION_REASON == "Retirement") %>%
  select(CITY) %>%
  ggplot(aes(x=CITY, fill=CITY)) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(x="City", y="Number of Retired Employees", title="Relationship between City and Retired Employees") +
  scale_x_discrete(guide=guide_axis(n.dodge = 2))
```

Code: Relationship between City and Retired Employees



Graph: Bar Graph of the Relationship between City and Retired Employees

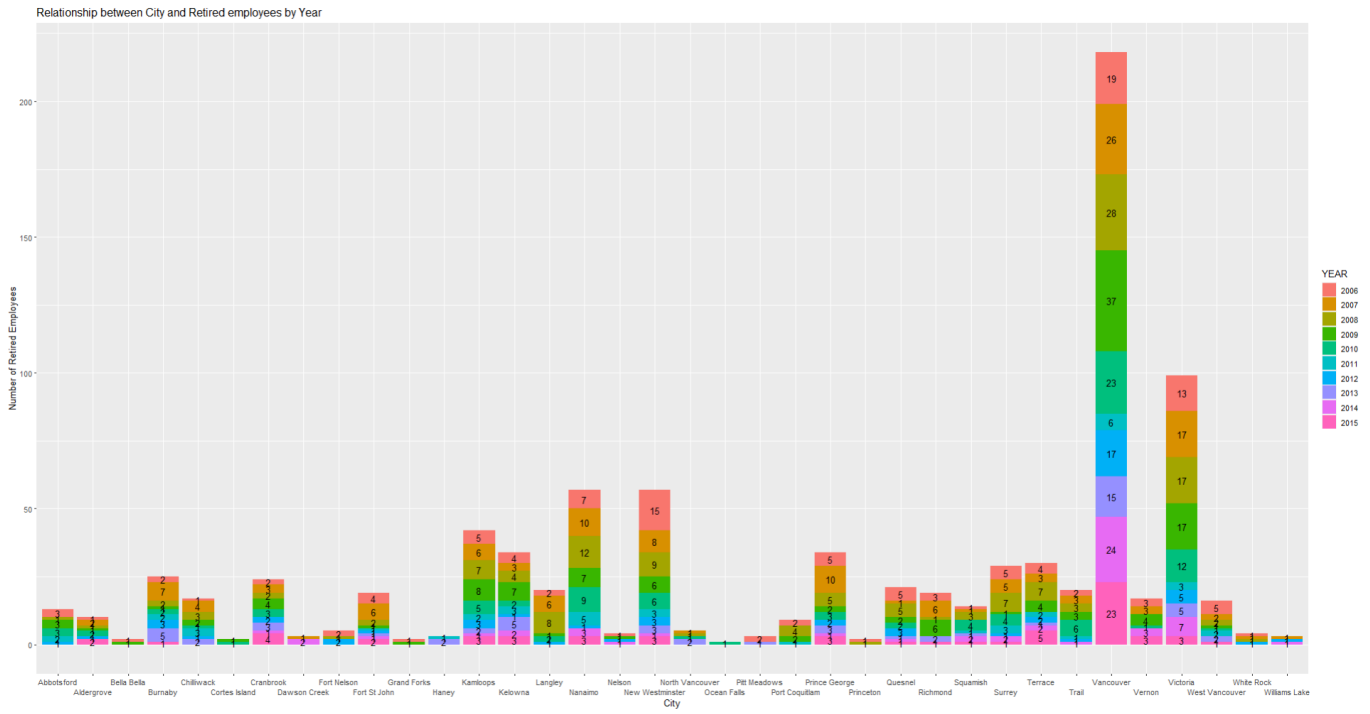
From the graph, we could see that a lot of people retired in Vancouver (218 employees) and Victoria (99 employees), which is reasonable since they have the most stores in their location. The city that had the least retired employees is Ocean Falls with only 1 employee.

Analysis 6.5.3.1 - Relationship between City and Retired employees by year

This analysis is to determine the relationship between the city and Retired employees by year.

```
data %>% filter(TERMINATION_REASON == "Retirement") %>%
  select(RECORD_YEAR, CITY) %>%
  ggplot(aes(x=CITY, fill=factor(RECORD_YEAR))) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="YEAR", x="City", y="Number of Retired Employees", title="Relationship between City and Retired Employees by Year") +
  scale_x_discrete(guide=guide_axis(n.dodge = 2))
```

Code: Relationship between City and Retired Employees by Year



Graph: Stacked Bar Graph of the Relationship between City and Retired Employees by Year

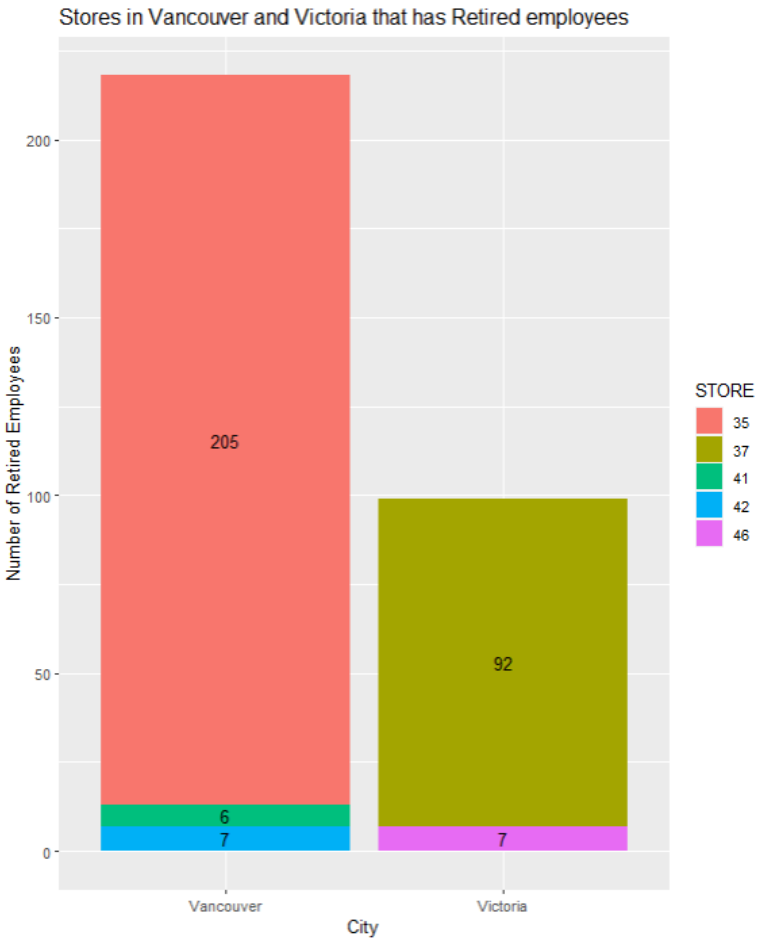
From the graph, we can see that most employees in Vancouver retired in 2009 (37 employees), and most employees in Victoria retired from 2007 to 2009 (17 employees). Since Vancouver and Victoria are in the list, we need to check which Stores in Vancouver and Victoria has retired employees.

Analysis 6.5.3.2 - Stores in Vancouver and Victoria that has Retired employees

This analysis is to determine which stores in Vancouver and Victoria have Retired employees.

```
data %>% filter(TERMINATION_REASON == "Retirement", (CITY=="Vancouver" | CITY=="Victoria")) %>%
  select(STORE, CITY) %>%
  ggplot(aes(x=CITY, fill=STORE)) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="STORE", x="City", y="Number of Retired Employees", title="Stores in Vancouver and Victoria that has Retired Employees")
```

Code: Stores in Vancouver and Victoria that has Retired Employees



Graph: Stacked Bar Graph of Stores in Vancouver and Victoria that has Retired Employees

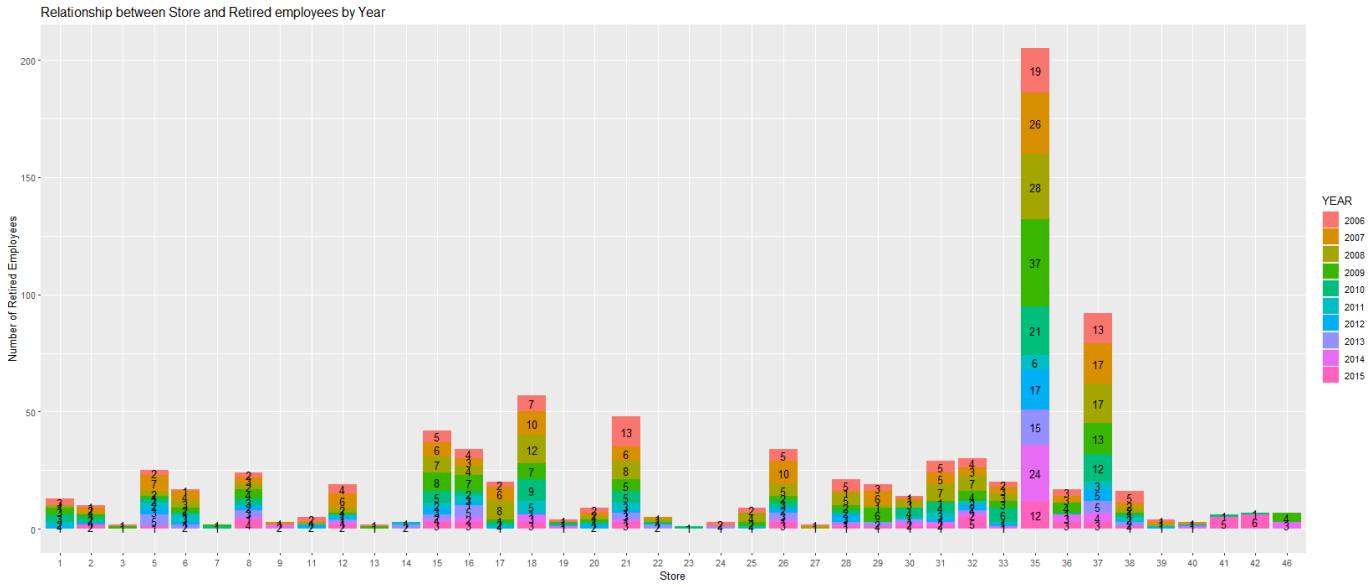
From the output, we know that the most employees that retired (205 employees) in Vancouver came from the Head Office (Store 35), while the most employees that retired (92 employees) in Victoria came from Store 37.

Analysis 6.5.3.3 - Relationship between Store and Retired employees

This analysis is to determine the relationship between Store and Retired employees by year.

```
data %>% filter(TERMINATION_REASON == "Retirement") %>%
  select(RECORD_YEAR, STORE) %>%
  ggplot(aes(x=STORE, fill=factor(RECORD_YEAR))) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="YEAR", x="Store", y="Number of Retired Employees", title="Relationship between Store and Retired Employees by Year")
```

Code: Relationship between Store and Retired employees by Year



Graph: Stacked Bar Graph of the Relationship between Store and Retired employees by Year

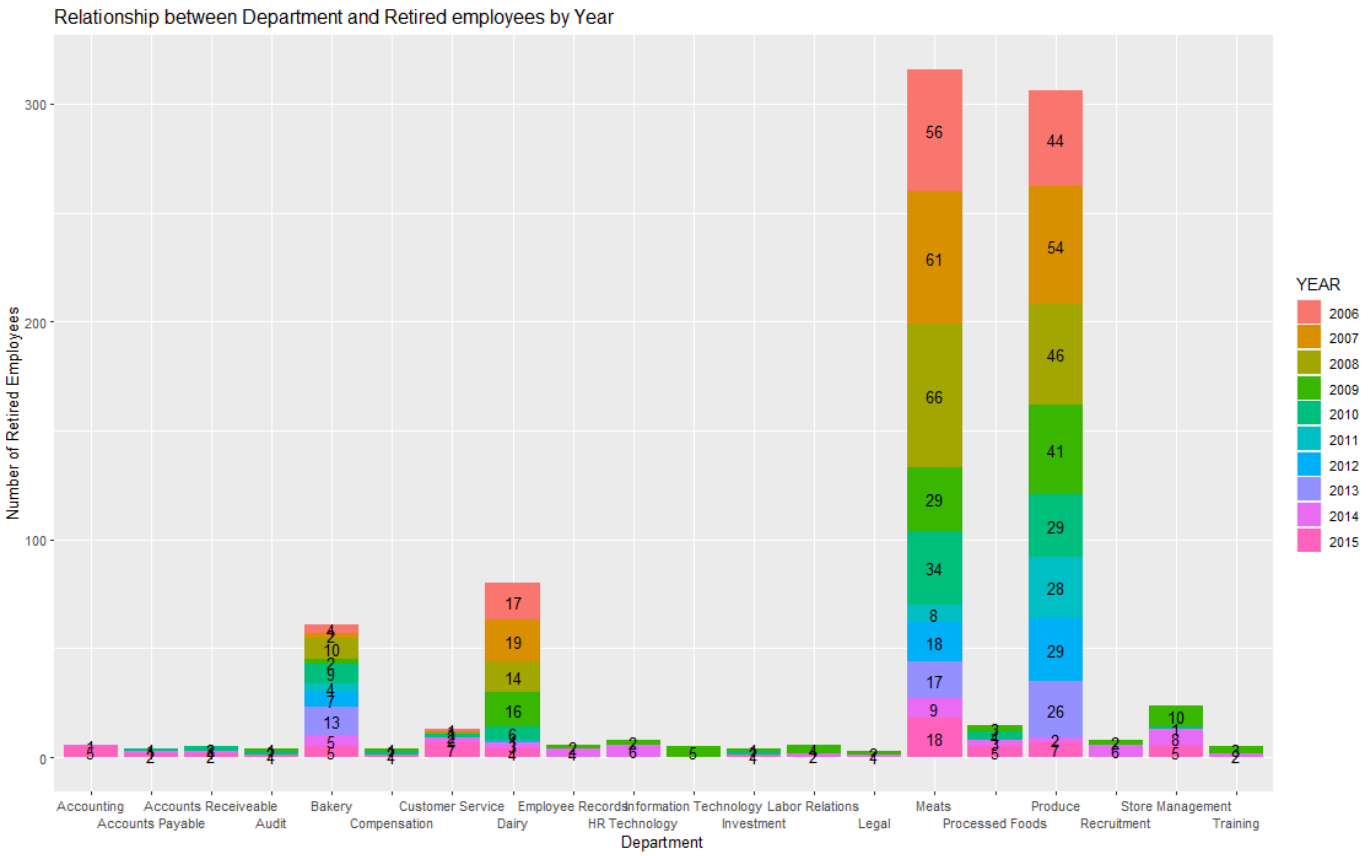
This graph aligns with Analysis 6.5.3. From the graph, we could see that Store 35 from Vancouver has the most employees that retired, most employees (37 employees) retired during 2009. This is followed by Store 37 from Vancouver, most employees (17 employees) retired during 2007 and 2008. The least amount of retired employees in every year per store is 1 employee.

Analysis 6.5.4 - Relationship between Department and Retired employees

This analysis is to determine the relationship between Department and Retired employees.

```
data %>% filter(TERMINATION_REASON == "Retirement") %>%
  select(RECORD_YEAR, DEPARTMENT) %>%
  ggplot(aes(x=DEPARTMENT, fill=factor(RECORD_YEAR))) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="YEAR", x="Department", y="Number of Retired Employees", title="Relationship between Department and Retired Employees by Year") +
  scale_x_discrete(guide=guide_axis(n.dodge = 2))
```

Code: Relationship between Department and Retired Employees by Year



Graph: Stacked Bar Graph of the Relationship between Department and Retired Employees by Year

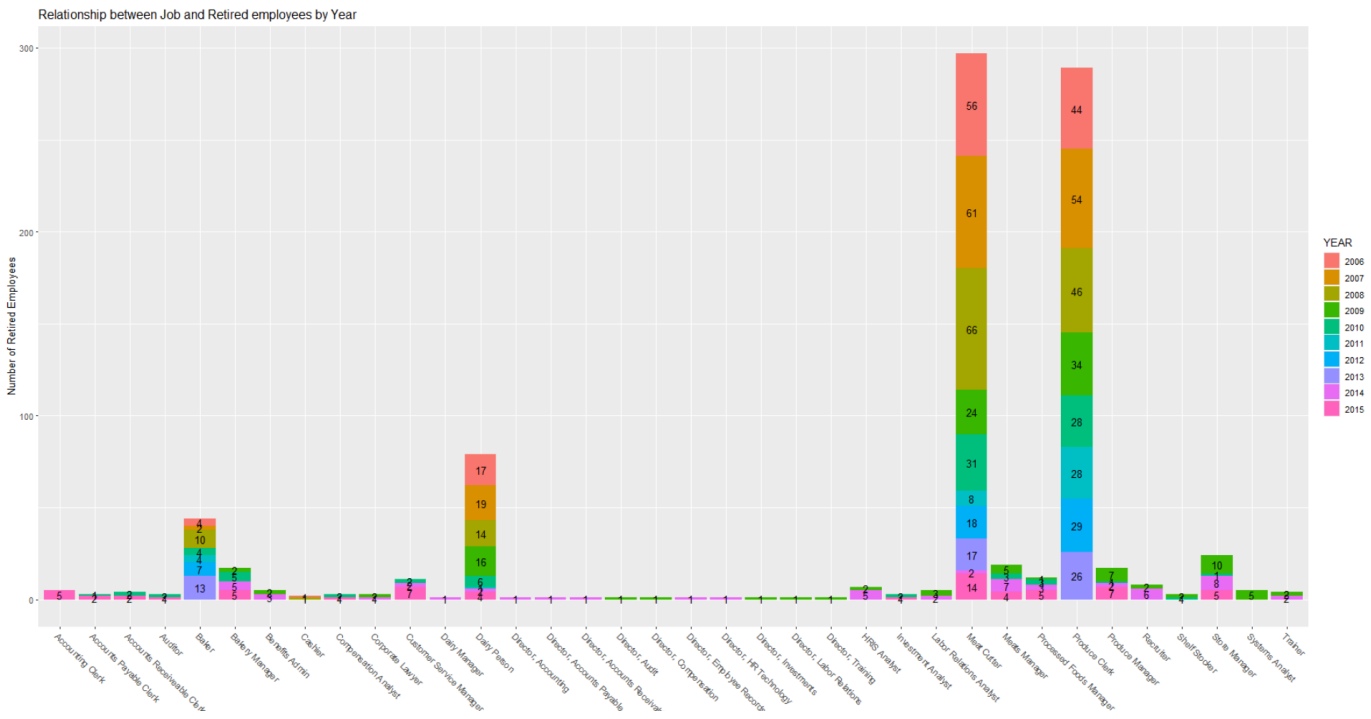
Since a lot of employees in Vancouver retired, there is a lot of new departments in the data. However, the department with the most retirements (316 employees) is the Meats department, where most retirements (66 employees) are from 2008. The produce also had a lot of retirements (306 employees), where most retirements (54 employees) are from 2007. The department that had the fewest retirements (3 employees) is the Legal department.

Analysis 6.5.5 - Relationship between Job and Retired employees

This analysis is to determine the relationship between Job and Retired employees.

```
data %>% filter(TERMINATION_REASON == "Retirement") %>%
  select(RECORD_YEAR, JOB_TITLE) %>%
  ggplot(aes(x=JOB_TITLE, fill=factor(RECORD_YEAR))) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="YEAR", x="Job", y="Number of Retired Employees", title="Relationship between Job and Retired Employees by Year") +
  theme(axis.text.x=element_text(angle=-45, hjust=0))
```

Code: Relationship between Job and Retired Employees by Year



Graph: Stacked Bar Chart of the Relationship between Job and Retired Employees by Year

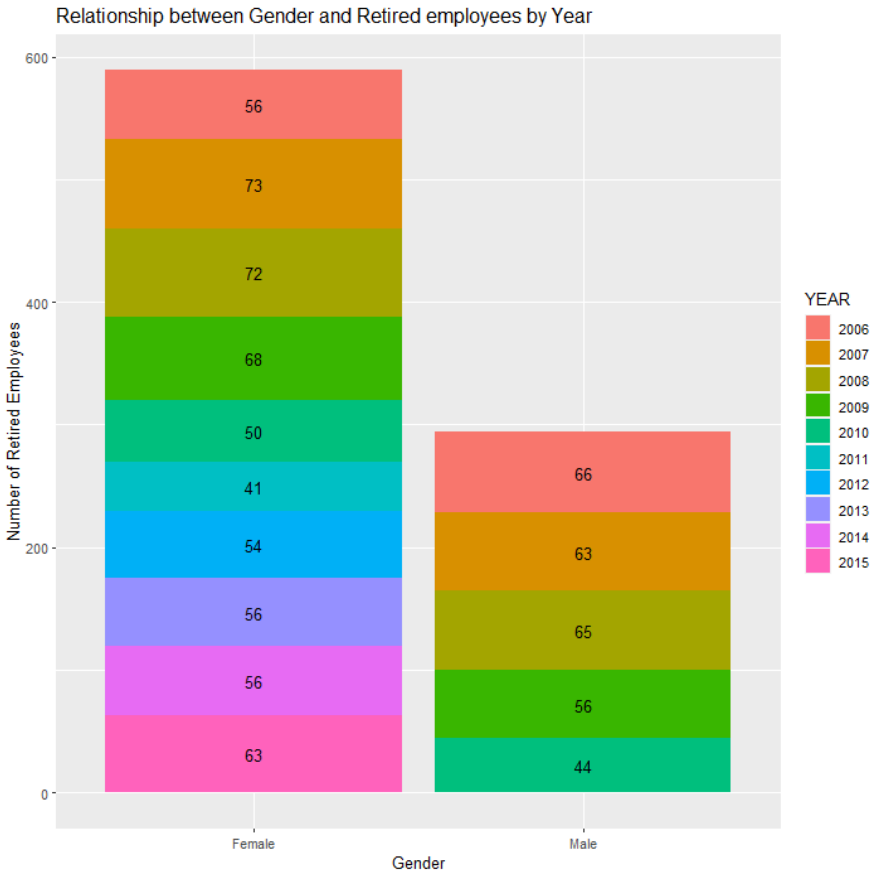
Since we know that from Analysis 6.5.4, a lot of employees resigned from the Meats department, where most retirements (66 employees) are from 2008. Therefore, the job that has the highest employee resignment is the Meat Cutter. The produce also had a lot of retirements (306 employees), where most retirements (54 employees) are from 2007, which aligns with Produce Clerk. The departments that had the fewest resignations (1 employee) are mostly the directors of different departments.

Analysis 6.5.5 - Relationship between Gender and Retired employees

This analysis is to determine the relationship between Gender and Retired employees by year.

```
data %>% filter(TERMINATION_REASON == "Retirement") %>%
  select(RECORD_YEAR, GENDER) %>%
  ggplot(aes(x=GENDER, fill=factor(RECORD_YEAR))) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="YEAR", x="Gender", y="Number of Retired Employees", title="Relationship between Gender and Retired Employees by Year")
```

Code: Relationship between Gender and Retired Employees by Year



Graph: Stacked Bar Chart of the Relationship between Gender and Retired Employees by Year

Overall, more female employees are retiring than males. From 2012 to 2015 however, only females employee retired.

Conclusion 5

Age

Employees only retire when they are 60 years old or 65 years old. The most employees retired at age 65 (589 employees), where most retired in 2007 (73 employees). This is because the normal age for retirement is 65 years old in Canada (CBC News, 2009). The least employees retired at age 60 (294 employee), where most retired in 2006 (66 employees).

Length of Service

The correlation between "AGE" and "LENGTH_OF_SERVICE" is 0.3, indicating that the two variables have a weak positive linear correlation. In 2009 and 2010, employees that retired on age 65 did not had as much length of service compared to the ones that retired on age 60. In 2006, 2007, 2008 the employees that retired on age 65 have more length of service compared to the ones that retired on age 60. Most

employees (470 employees) retired after working for 13 years. 47 employees retired after working for the longest length of service which is 25 years, while 194 employees retired after working for the shortest length of service which is 8 years.

City

A lot of people retired in Vancouver (218 employees), mostly in 2009 (37 employees). A lot of people also retired in Victoria (99 employees), mostly from 2007 to 2009 (17 employees). This is reasonable since they have the most stores in their location. The city that had the least retired employees is Ocean Falls with only 1 employee.

Store

Most employees that retired (205 employees) in Vancouver came from the Head Office (Store 35), and most employees (37 employees) retired during 2009. I think the reason why so many employees from Store 35 retired instead of resigned or laid-off was because the pay is sufficient as most jobs in the Head Office are high paying jobs. It could also mean that they are happy and satisfied with their jobs. The most employees that retired (92 employees) in Victoria came from Store 37, and most employees (17 employees) retired during 2007 and 2008. The least amount of retired employees in every year per store is 1 employee.

Department

The department with the most retirements (316 employees) is the Meats department, where most retirements (66 employees) are from 2008. The Produce department also had a lot of retirements (306 employees), where most retirements (54 employees) are from 2007. The department that had the fewest retirements (3 employees) is the Legal department.

Job

A lot of employees resigned from the Meats department, where most retirements (66 employees) are from 2008. Therefore, the job that has the highest employee resignment is the Meat Cutter. The produce also had a lot of retirements (306 employees), where most retirements (54 employees) are from 2007, which aligns with Produce Clerk. The departments that had the fewest resignations (1 employee) are mostly the directors of different departments. I think the reason why so many employees from the Meats and Produce

department retired instead of resigned or laid-off was because the pay is sufficient. It could also mean that they are happy and satisfied with their jobs.

Gender

More female employees are retiring than males. From 2012 to 2015 however, only females employee retired.

7.0 EXTRA FEATURES

7.1 - list.functions.in.file()

This is to check exactly what packages and functions have I used throughout the code.

```
library(NCmisc)
list.functions.in.file("C:\\Users\\shiau\\OneDrive - Asia Pacific University\\DEGREE\\Y2\\PFDA\\ASSIGNMENT\\CHANG SHIAU HUEI_TP060322.R", alphabetic = TRUE)
```

Code: list.functions.in.file()

```
> list.functions.in.file("C:\\Users\\shiau\\OneDrive - Asia Pacific University\\DEGREE\\Y2\\PFDA\\ASSIGNMENT\\CHANG SHIAU HUEI_TP060322.R", alphabetic = TRUE)
$`c("package:plotly", "package:dplyr")`
[1] "arrange"  "distinct" "group_by" "mutate"   "select"

$`c("package:plotly", "package:dplyr", "package:stats")`
[1] "filter"

$`c("package:viridis", "package:viridisLite")`
[1] "magma"   "viridis"

$`character(0)`
[1] "map_df"

$`package:base`
 [1] "as.Date"    "c"          "cbind"      "class"      "colnames"   "colSums"    "data.frame" "factor"
 [9] "format"     "grepl"      "ifelse"     "library"    "names"      "nlevels"    "nrow"       "paste0"
[17] "round"      "split"      "strptime"   "subset"     "summary"    "unique"

$`package:dplyr`
[1] "desc"       "glimpse"    "n_distinct"

$`package:ggcorrplot`
[1] "ggcorrplot"

$`package:ggplot2`
 [1] "aes"                "after_stat"         "coord_flip"         "element_text"       "facet_grid"
 [6] "facet_wrap"         "geom_area"          "geom_bar"           "geom_boxplot"       "geom_count"
[11] "geom_density"       "geom_freqpoly"      "geom_histogram"     "geom_line"          "geom_path"
[16] "geom_point"         "geom_text"          "ggplot"             "ggtitle"            "guide_axis"
[21] "guides"             "labs"               "position_stack"     "scale_fill_gradient" "scale_fill_manual"
[26] "scale_x_continuous" "scale_x_discrete"   "stat_smooth"        "theme"

$`package:graphics`
[1] "pie"

$`package:grDevices`
[1] "cm.colors"   "heat.colors"

$`package:janitor`
[1] "adorn_pct_formatting" "adorn_totals"         "clean_names"          "get_dupes"
[5] "tabyl"

$`package:NCmisc`
[1] "list.functions.in.file"

$`package:plotly`
[1] "ggplotly"

$`package:plotrix`
[1] "pie3D"

$`package:stats`
[1] "cor"

$`package:utils`
[1] "read.csv" "str"      "tail"     "View"
```

Output: list.functions.in.file()

7.2 - get_dupes()

To check for duplicate records.

```
get_dupes(data)
```

Code: get_dupes()

```
> get_dupes(data)
No variable names specified - using all columns.

No duplicate combinations found of: EMPLOYEE_ID, RECORD_DATE, BIRTH_DATE, ORI_HIRE_DATE, TERMINATION_DATE, AGE, LENGTH_OF_SERVICE, CITY, DEPARTMENT, ... and 8 other variables
 [1] EMPLOYEE_ID        RECORD_DATE        BIRTH_DATE         ORI_HIRE_DATE      TERMINATION_DATE   AGE                LENGTH_OF_SERVICE  CITY               DEPARTMENT
[10] JOB_TITLE          STORE              GENDER             TERMINATION_REASON TERMINATION_TYPE   RECORD_YEAR        STATUS             BUSINESS_UNIT      dupe_count
<0 rows> (or 0-length row.names)
```

Output:get_dupes()

7.3 - colSums()

To form a column sum for data frames, while providing a condition.

```
colSums(data == "")
```

Code: colSums()

```
> colSums(data == "")
       EmployeeID     recorddate_key      birthdate_key    orighiredate_key terminationdate_key               age   length_of_service          city_name     department_name
                0                  0                  0                  0                  0                  0                  0                  0                  0
        job_title         store_name        gender_full     termreason_desc       termtype_desc        STATUS_YEAR             STATUS      BUSINESS_UNIT
                0                  0                  0                  0                  0                  0                  0                  0
```

Output: colSums()

7.4 - clean_names()

To clean and set the column names of a dataframe all in one go by providing the casing required.

```
data <- clean_names(data, case="all_caps")
```

Code: clean_names()

```
> colnames(data) # View column headings
 [1] "EmployeeID"         "recorddate_key"     "birthdate_key"      "orighiredate_key"   "terminationdate_key" "age"                "length_of_service"  "city_name"
 [9] "department_name"    "job_title"          "store_name"         "gender_full"        "termreason_desc"     "termtype_desc"      "STATUS_YEAR"        "STATUS"
[17] "BUSINESS_UNIT"
> data <- clean_names(data, case="all_caps")
>
> # 4.1.2 Change Column Names
> colnames(data)[2] <- "RECORD_DATE"
> colnames(data)[3] <- "BIRTH_DATE"
> colnames(data)[4] <- "ORI_HIRE_DATE"
> colnames(data)[5] <- "TERMINATION_DATE"
> colnames(data)[8] <- "CITY"
> colnames(data)[9] <- "DEPARTMENT"
> colnames(data)[11] <- "STORE"
> colnames(data)[12] <- "GENDER"
> colnames(data)[13] <- "TERMINATION_REASON"
> colnames(data)[14] <- "TERMINATION_TYPE"
> colnames(data)[15] <- "RECORD_YEAR"
> colnames(data)
 [1] "EMPLOYEE_ID"        "RECORD_DATE"        "BIRTH_DATE"         "ORI_HIRE_DATE"      "TERMINATION_DATE"   "AGE"                "LENGTH_OF_SERVICE"  "CITY"
 [9] "DEPARTMENT"         "JOB_TITLE"          "STORE"              "GENDER"             "TERMINATION_REASON" "TERMINATION_TYPE"   "RECORD_YEAR"        "STATUS"
[17] "BUSINESS_UNIT"
```

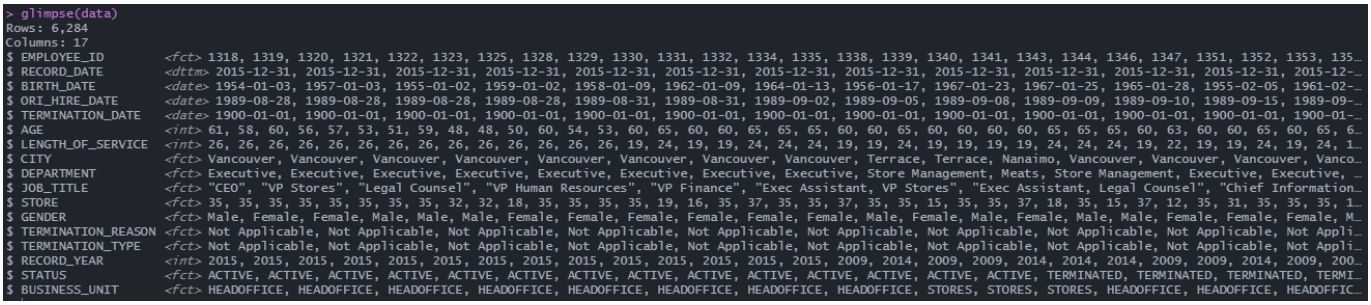
Output: clean_names()

7.5 - glimpse()

To show more information at once according to console size during data exploration, as it displays the names of all column headings, datatype of the column, number of columns, and number of rows.

```
glimpse(data)
```

Code: glimpse()



Output: glimpse()

7.6, 7.7, 7.8 - tabyl(), map_df(), adorn_totals()

Tabyl() is used to generate a frequency table with less code, map_df() makes sures that the data returned is a data frame, and adorn_totals() is to count the total of a column or row from a tabyl() output.

```
## Create a table that displays the frequency of the number of occurrence for each employee ID
exp_emp_id <- tabyl(data_full, EMPLOYEE_ID) %>%
  dplyr::arrange( n ) %>% # Arrange "n" in order
  split( .,"n"] ) %>%     # Splits "n" into different dimensions
  purrr::map_df(., janitor::adorn_totals) %>%
  # :: is to get this 1 function from a certain package
  # . is to get all variable
  filter(EMPLOYEE_ID == "Total")

## Appending the frequency number column to the data since the min & max frequency is known
Times_Repeated = factor(c(1:10))
exp_emp_id = cbind(exp_emp_id, Times_Repeated)

## Deleting unecessary columns
exp_emp_id["EMPLOYEE_ID"] = NULL

## Reorder column
exp1_emp_id <- exp_emp_id[, c(3,1,2)]

## TABLE: Frequency of the number of occurrence for EMPLOYEE_ID
exp1_emp_id %>%
    adorn_totals() %>%
    adorn_pct_formatting(digits=2, affix_sign=TRUE, rounding="half to even", percent)
```

Code: tabyl(), map_df(), adorn_totals()

Times_Repeated	n	percent
1	146	0.29%
2	512	1.03%
3	1194	2.40%
4	1536	3.09%
5	1690	3.40%
6	1806	3.64%
7	2205	4.44%
8	2296	4.62%
9	2898	5.84%
10	35370	71.23%
Total	49653	100.00%

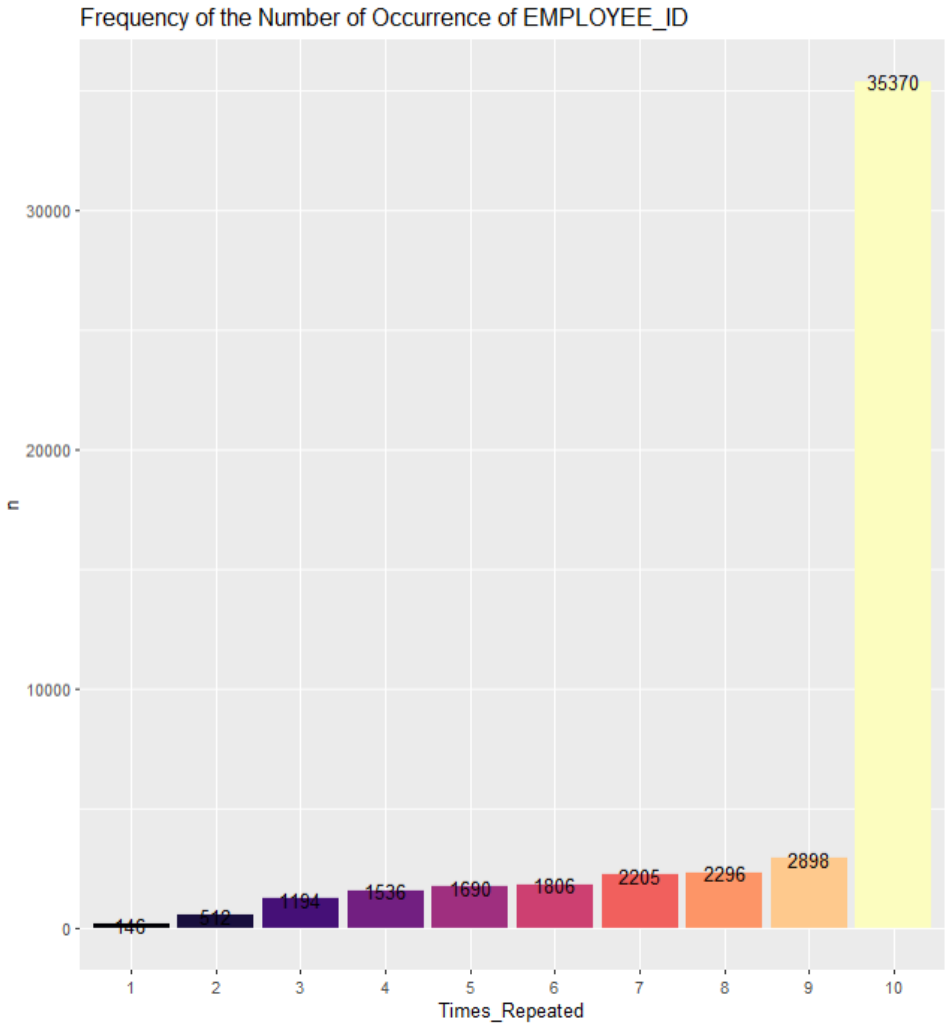
Output: tabyl(), map_df(), adorn_totals()

7.9, 7.10, 7.11, 7.12 - magma(), viridis(), heat.colors(), cm.colors()

These are colour palettes and maps for data visualisation to improve graph readability for readers who suffer from common forms of colour blindness or colour vision deficiency.

```
ggplot(exp_emp_id, aes(x=Times_Repeated, y=n)) +
  geom_bar(stat="identity", fill=magma(10)) +
  geom_text(aes(label=n)) +
  ggtitle("Frequency of the Number of Occurrence of EMPLOYEE_ID")
```

Code: magma()



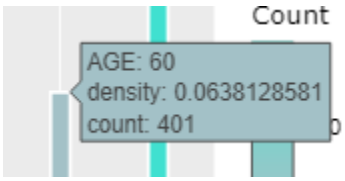
Output: magma()

7.13 - ggplotly()

This allows you to plot a variety of graphs, while having the ability to display certain information when hovered over, zooming in on certain properties, hiding attributes and values on the click of a button, and more.

```
ggplotly(ggplot(data, aes(x=AGE)) +
   geom_histogram(colour="white", aes(y=..density.., fill=..count..), binwidth=1) +
   scale_fill_gradient("Count", low="hotpink", high="turquoise") +
   geom_density(alpha=.1, fill="black") +
   ggtitle("Frequency and Density of AGE"))
```

Code: ggplotly()



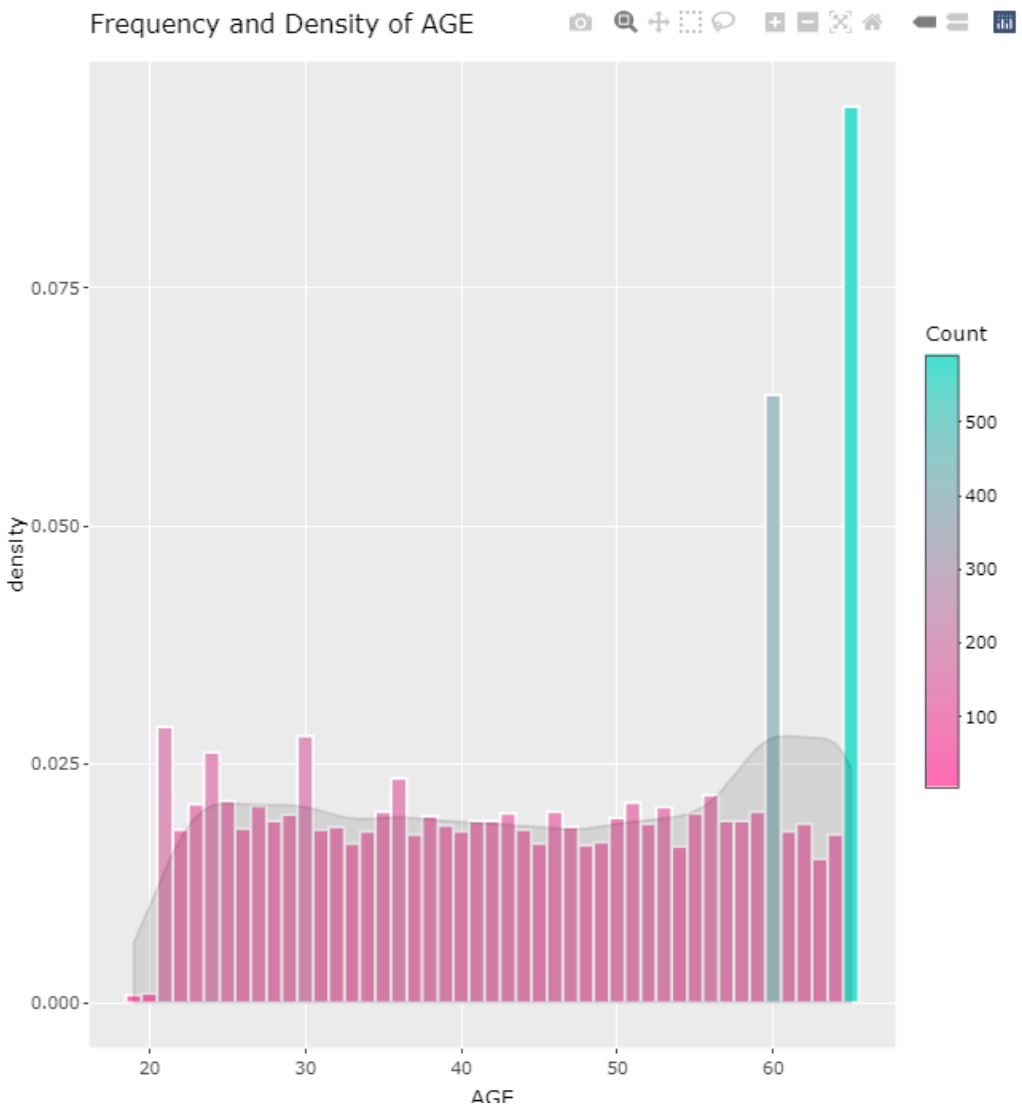
Output: ggplotly()

7.14 - geom_density()

To create a density plot. Density plots are good in a way that they are better at determining the shape of a distribution, which makes it easier to find patterns of data.

```
ggplotly(ggplot(data, aes(x=AGE)) +
   geom_histogram(colour="white", aes(y=..density.., fill=..count..), binwidth=1) +
   scale_fill_gradient("Count", low="hotpink", high="turquoise") +
   geom_density(alpha=.1, fill="black") +
   ggtitle("Frequency and Density of AGE"))
```

Code: geom_density()



Output: geom_density()

7.15 - grepl()

This stands for “grep logical”, and it is used to search for matches of a string vector or string.

```
select(exp_job, c(JOB_TITLE,n)) %>%
  dplyr::filter(grepl('Director', JOB_TITLE)) %>%
  adorn_totals()
```

Code: grepl()

```
> select(exp_job, c(JOB_TITLE,n)) %>%
+   dplyr::filter(grepl('Director', JOB_TITLE)) %>%
+   adorn_totals()
                      JOB_TITLE  n
          Director, Accounting  1
    Director, Accounts Payable  1
 Director, Accounts Receivable  1
              Director, Audit  1
        Director, Compensation  1
    Director, Employee Records  1
       Director, HR Technology  1
         Director, Investments  1
     Director, Labor Relations  1
         Director, Recruitment  1
            Director, Training  1
                         Total 11
```

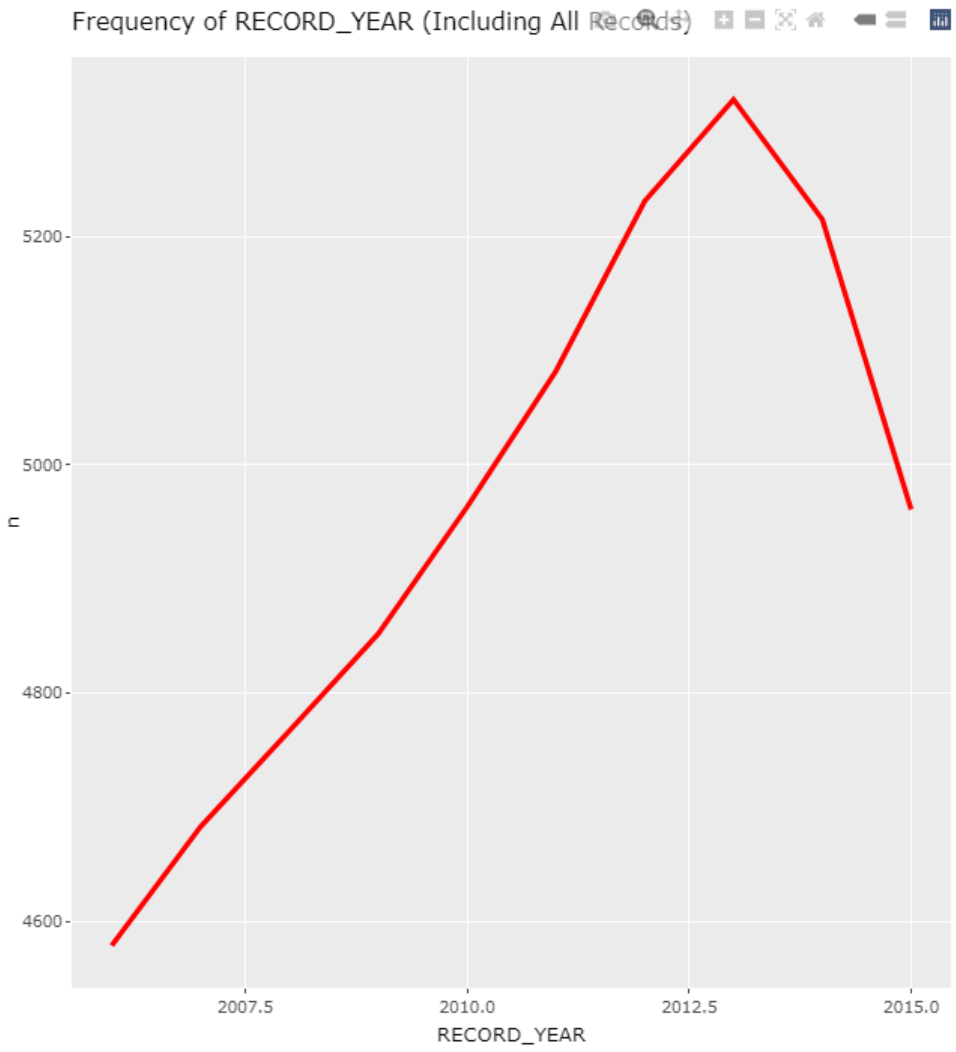
Output: grepl()

7.16 - scale_x_continuous()

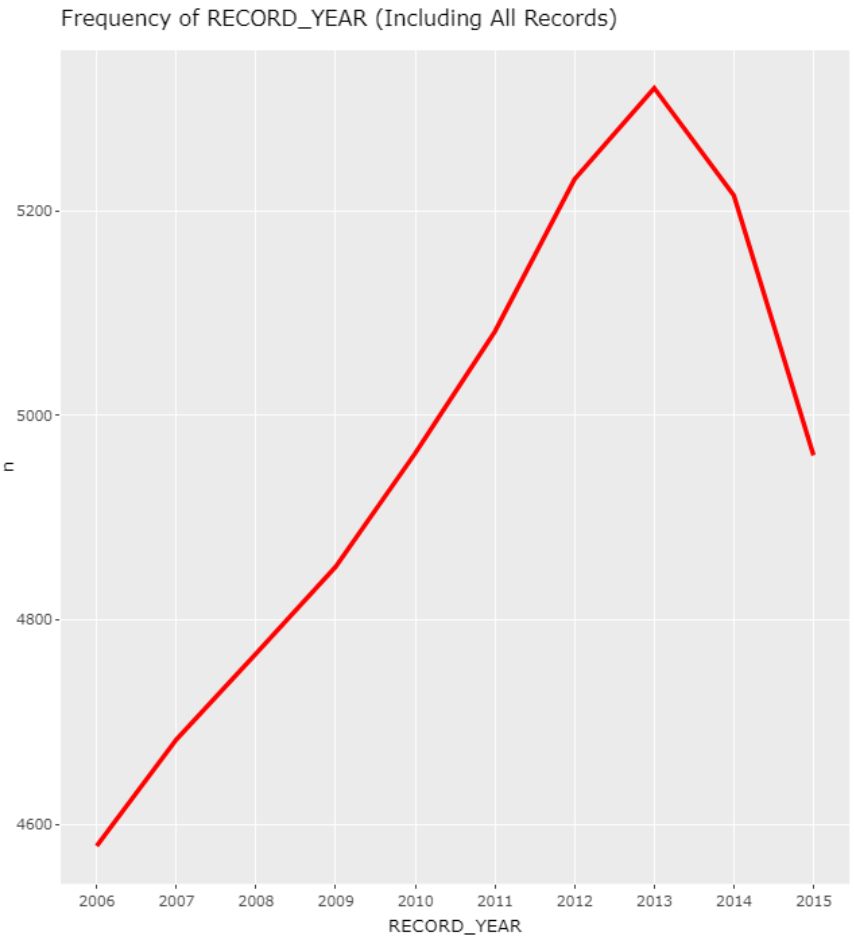
Instead of letting R generate the x-axis, I used this function to break it down into the x-axis values I wanted.

```
exp_record_year_vis <- ggplot(exp_record_year, aes(x=RECORD_YEAR, y=n)) +
  geom_line(col="red", size=1) +
  ggtitle("Frequency of RECORD_YEAR (Including All Records)") +
  scale_x_continuous(breaks=unique(data_full$RECORD_YEAR))
ggplotly(exp_record_year_vis)
```

Code: scale_x_continuous()



Output: Without scale_x_continuous()



Output: With scale_x_continuous()

7.17 - cor()

To create a correlation matrix.

```
age_los <- data %>% select(AGE, LENGTH_OF_SERVICE)
cor_matrix <- round(cor(age_los), 1)
cor_matrix
```

Code: cor()

```
> cor_matrix
                   AGE LENGTH_OF_SERVICE
AGE                1.0                0.8
LENGTH_OF_SERVICE  0.8                1.0
```

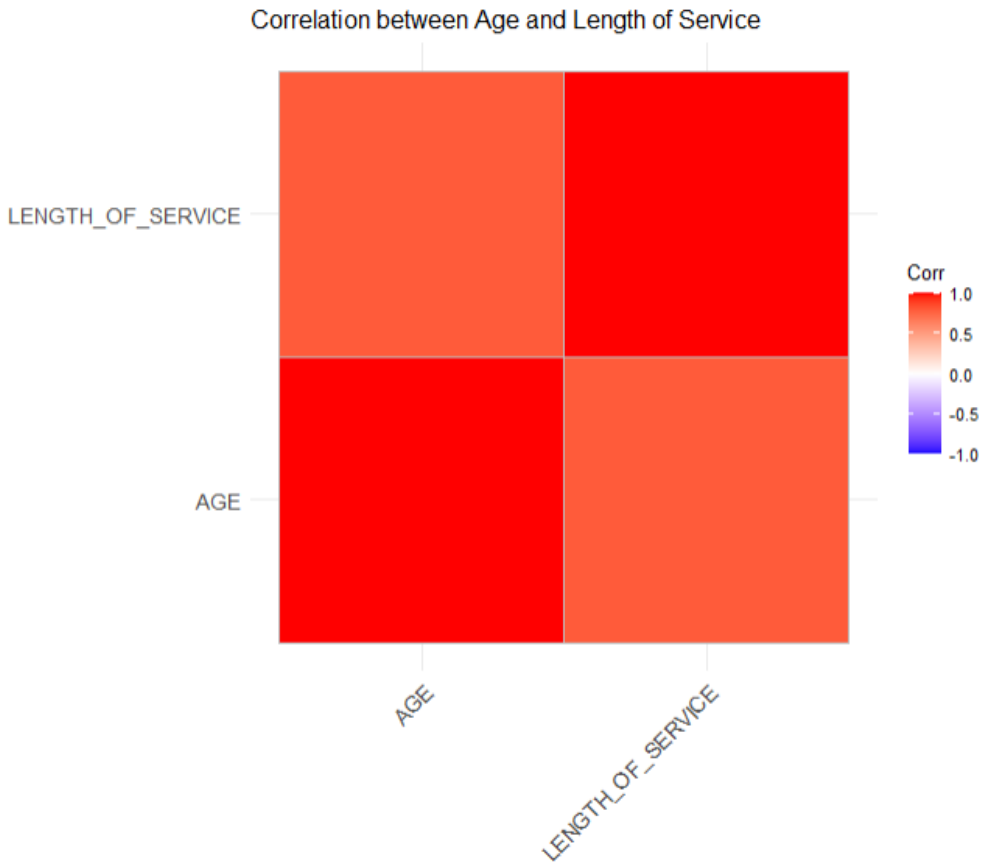
Output: cor()

7.18 - ggcorrplot()

To reorder, display significant level, and visualize a correlation matrix.

```
ggcorrplot(cor_matrix) + labs(title="Correlation between Age and Length of Service")
```

Code: ggcorrplot()



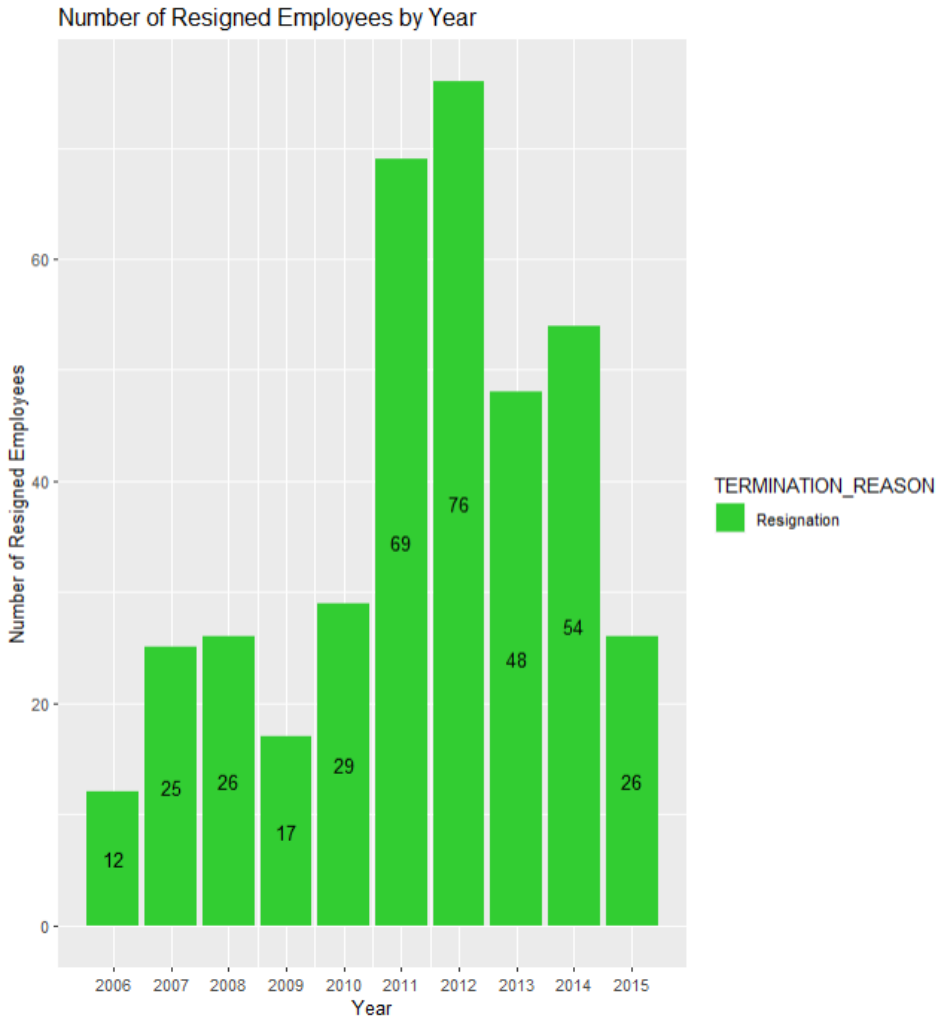
Output: ggcorrplot()

7.19 - scale_fill_manual()

Instead of letting R generate the fill colour, I used this function to specify the fill colour that I wanted.

```
data %>% filter(TERMINATION_REASON == "Resignation") %>%
  select(RECORD_YEAR, TERMINATION_REASON) %>%
  ggplot(aes(x=RECORD_YEAR, fill=TERMINATION_REASON)) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(x="Year", y="Number of Resigned Employees", title="Number of Resigned Employees by Year") +
  scale_x_continuous(breaks=unique(data$RECORD_YEAR)) +
  scale_fill_manual(values = "limegreen")
```

Code: scale_fill_manual()



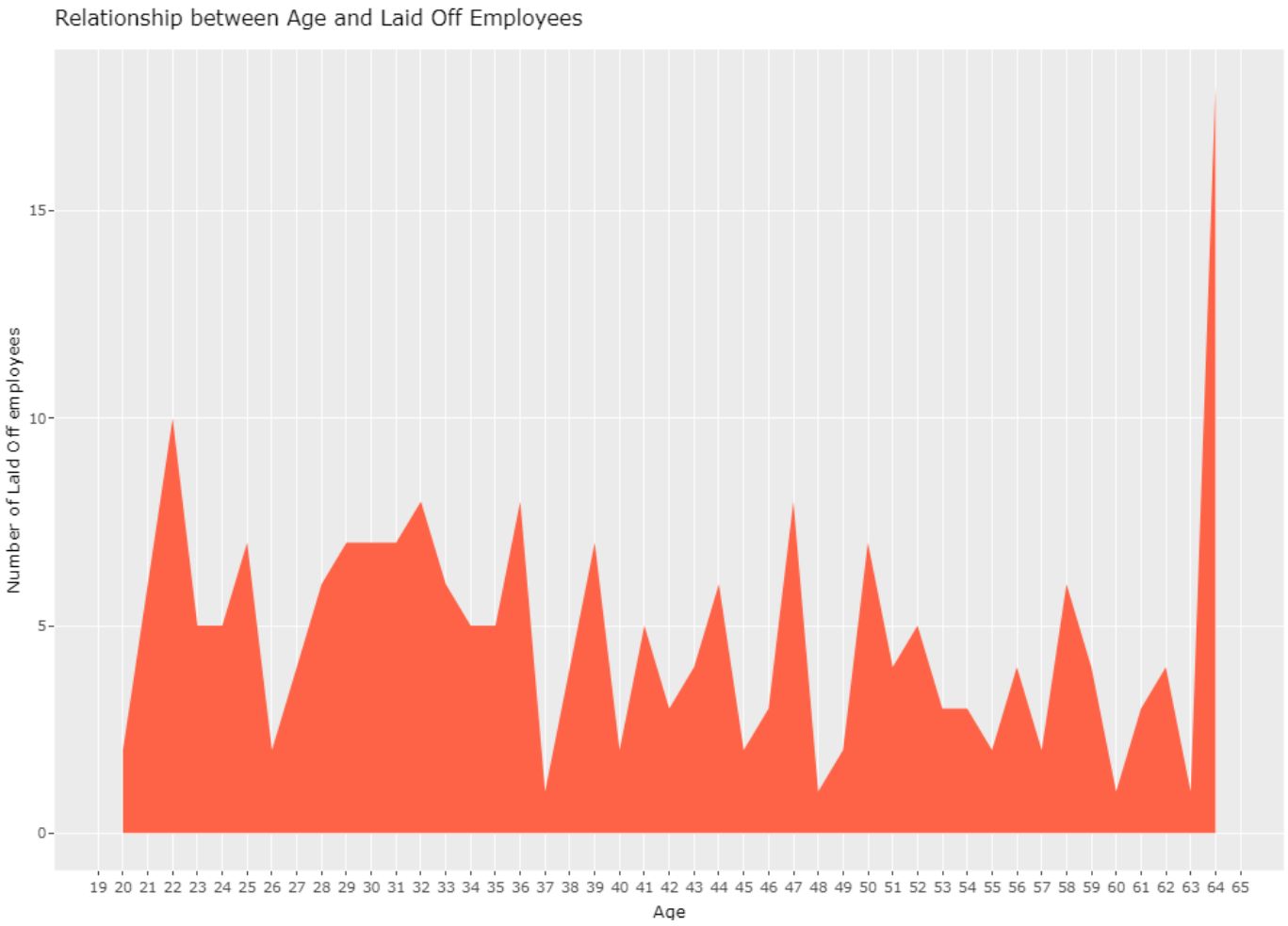
Output: scale_fill_manual()

7.20 - geom_area()

To create an area graph. This makes visualization more readable.

```
ggplotly(data %>% filter(TERMINATION_REASON == "Layoff") %>%
   select(RECORD_YEAR, AGE) %>%
   ggplot(aes(x=AGE)) +
   geom_area(aes(y = ..count..), stat ="bin", binwidth=1, fill="tomato") +
   labs(colour="YEAR", x="Age", y="Number of Laid Off employees", title="Relationship between Age and Laid Off Employees") +
   scale_x_continuous(breaks=unique(data$AGE)))
```

Code: geom_area()



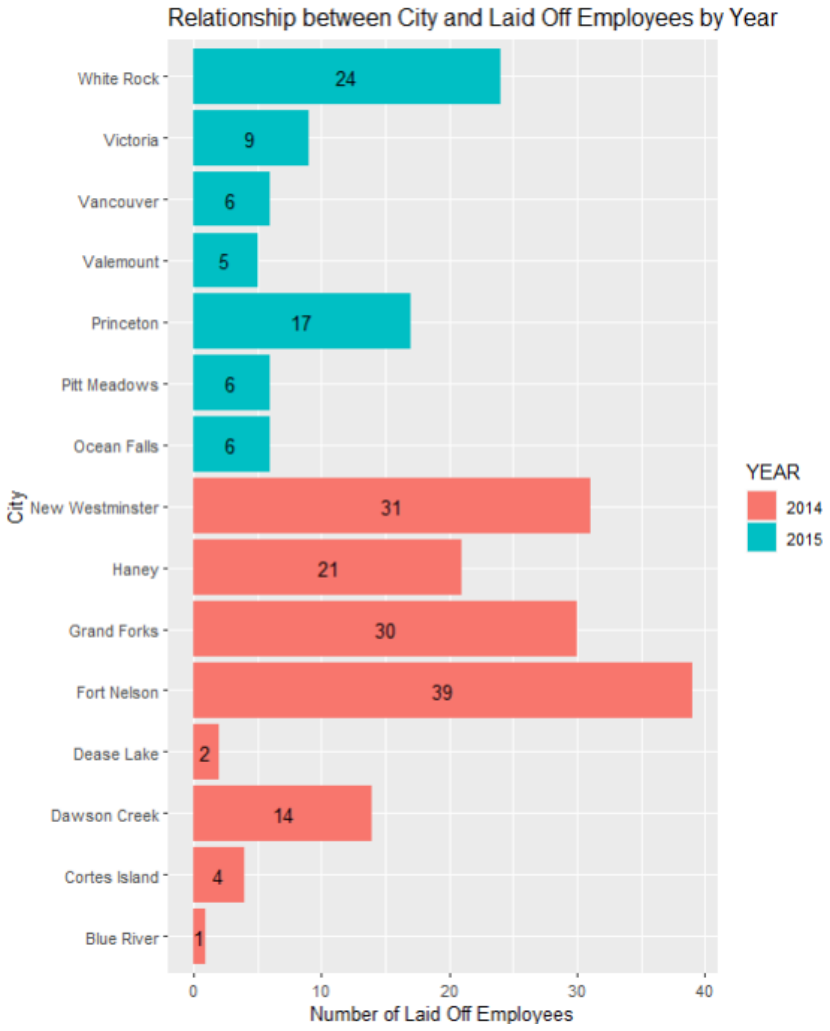
Output: geom_area()

7.21 - coord_flip()

This is used to flip the cartesian coordinates so that horizontal becomes vertical and vertical becomes horizontal, and vice versa. This is convenient when the axis is not long enough for values to be readable.

```
data %>% filter(TERMINATION_REASON == "Layoff") %>%
  select(RECORD_YEAR, CITY) %>%
  ggplot(aes(x=CITY, fill=factor(RECORD_YEAR))) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="YEAR", x="City", y="Number of Laid Off Employees", title="Relationship between City and Laid Off Employees by Year") +
  coord_flip()
```

Code: coord_flip()



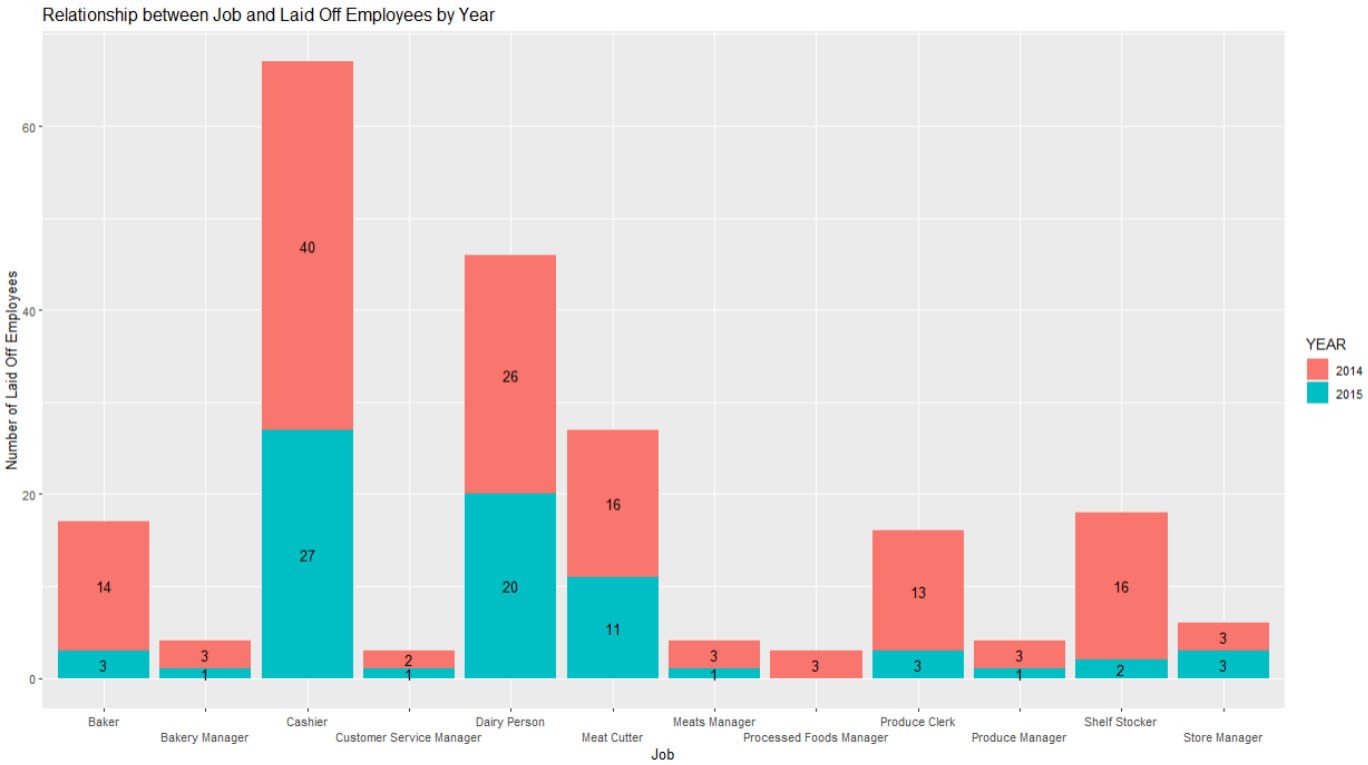
Output: coord_flip()

7.22 - scale_x_discrete()

I used this function to display the values in the x axis on 2 separate lines, so they don't overlap, which become unreadable.

```
data %>% filter(TERMINATION_REASON == "Layoff") %>%
  select(RECORD_YEAR, JOB_TITLE) %>%
  ggplot(aes(x=JOB_TITLE, fill=factor(RECORD_YEAR))) +
  geom_bar() +
  geom_text(aes(label = after_stat(count)), stat="count", position=position_stack(vjust = 0.5)) +
  labs(fill="YEAR", x="Job", y="Number of Laid Off Employees", title="Relationship between Job and Laid Off Employees by Year") +
  scale_x_discrete(guide=guide_axis(n.dodge = 2))
```

Code: scale_x_discrete()



Output: scale_x_discrete()

8.0 CONCLUSION

This assignment consists of a total number of 8 questions, 58 analysis, 22 extra features, and 13 different types of charts. The types of chart used are namely Pie Chart, Line Graph, Path Graph, Point Graph, Bar Chart, Histogram, Frequency Polygon, Scatter Plot, Count Plot, Box Plot, Density Graph, Area Graph and Correlation matrix.

9.0 REFERENCES

Adkins, A. (2019). *Millennials: The Job-Hopping Generation*. Gallup.Com. https://www.gallup.com/workplace/231587/millennials-job-hopping-generation.aspx

CBC News. (2009). *Mandatory retirement fades in Canada*. https://www.cbc.ca/news/business/mandatory-retirement-fades-in-canada-1.799697

CommBox. (2020). *10 Reasons for Customer Service Agents Burnout and How to Fix It*. https://www.commbox.io/10-reasons-for-customer-service-agents-burnout-and-how-to-fix-it/

Dorsey, J. (2021). *Generations Birth Years - Gen Z, Millennials, Gen X, and Baby Boomers*. JasonDorsey. https://jasondorsey.com/about-generations/generations-birth-years/

Dorsey, J. (2022). *Gen Z and tech dependency: How the youngest generation interacts differently with the digital world*. Jason Dorsey. https://jasondorsey.com/blog/gen-z-and-tech-dependency-how-the-youngest-generation-interacts-differently-with-the-digital-world/

Garnier, S. (2021). *CRAN - Package viridis*. CRAN. https://cran.r-project.org/web/packages/viridis/index.html

GlowTouch. (2021). *Mastering Agent Turnover in Your Customer Support Center*. GlowTouch LLC. https://www.glowtouch.com/mastering-agent-turnover-in-your-customer-support-center/

Institute for Family Studies. (2022). *Why Young Adults in Canada Embrace Marriage—or Reject It*. https://ifstudies.org/blog/why-young-adults-in-canada-embrace-marriageor-reject-it

Lablogatory. (2018). *Why is it Important to Learn About Generations?* https://labmedicineblog.com/2018/03/30/why-is-it-important-to-learn-about-generations/

Miller, L. (2019). *9 Reasons Talented Millennials Get Fired*. Entrepreneur. https://www.entrepreneur.com/article/305852

Moran, A. (2022). *The 30 Most Boring Jobs in the World*. CareerAddict.

https://www.careeraddict.com/top-10-most-boring-jobs-2015

O'Donnell, J. T. (2021). *3 Reasons Baby Boomers Are Getting Fired*. Inc.Com. https://www.inc.com/jt-odonnell/3-reasons-baby-boomers-are-getting-fired.html

stat.berkeley.edu. (2006a). *Dates and Times in R*. https://www.stat.berkeley.edu/%7Es133/dates.html

stat.berkeley.edu. (2006b). *Factors in R*. https://www.stat.berkeley.edu/%7Es133/factors.html

STHDA. (2022). *ggcorrplot: Visualization of a correlation matrix using ggplot2 - Easy Guides - Wiki - STHDA*. http://www.sthda.com/english/wiki/ggcorrplot-visualization-of-a-correlation-matrix-using-ggplot2

Zach. (2022). *How to Read a Correlation Matrix*. Statology. https://www.statology.org/how-to-read-a-correlation-matrix/