practice_exercise

Ekta Chaudhary

28/06/2020

#Reading the dataset

```
practice_data = read_excel("./data/Practice_exercise.xlsx", sheet = "Data") %>%
  janitor::clean_names() %>%
  select(observation_number,quarter,employee_id, sex = sex_male_1, race, age, hospital_visit = hospital_
  mutate(
    age_cat = case_when(
      age < 30 ~ 1,
      age <= 45 ~ 2,
      age > 45 ~ 3
    )
  )
```

#Checking for missing data

```
sapply(practice_data, function(x) sum(is.na(x)))
```

```
## observation_number            quarter        employee_id
##                  0                  0                  0
##                sex               race                age
##                 71               2123                  0
##     hospital_visit             salary       health_score
##                  0                  0                  0
##            age_cat
##                  0
```

```
practice_data %>%
  select(everything()) %>%
  summarise_all(funs(sum(is.na(.))))
```

```
## # A tibble: 1 x 10
##   observation_num~ quarter employee_id   sex  race   age hospital_visit
##              <int>   <int>       <int> <int> <int> <int>          <int>
## 1                0       0           0    71  2123     0              0
## # ... with 3 more variables: salary <int>, health_score <int>,
## #   age_cat <int>
```

#Finding the minimum and maximum values of each variable

```
sapply(practice_data, function(x) min(x))
```

```
## observation_number            quarter        employee_id
##       1.000000e+00       1.000000e+00       1.000000e+00
##                sex               race                age
```

```
##                    NA                    NA          7.000000e+00
##        hospital_visit                salary          health_score
##          0.000000e+00          2.835070e+04          6.265991e-01
##               age_cat
##          1.000000e+00
```

```
sapply(practice_data, function(x) max(x))
```

```
## observation_number              quarter          employee_id
##            19103.00                12.00              2000.00
##                 sex                 race                  age
##                  NA                   NA               172.00
##      hospital_visit               salary         health_score
##                1.00             68826.34                10.00
##             age_cat
##                3.00
```

#Checking the number of employees with health score outside the range of data

```
practice_data %>%
  count(
   health_sc_6 = ifelse(health_score > 6, 1, 0)
  )
```

```
## # A tibble: 2 x 2
##   health_sc_6     n
##         <dbl> <int>
## 1           0 17865
## 2           1  1238
```

#Calculating the number of quarters for which the employees have missing data on sex

```
practice_data %>%
  select(
    employee_id, sex
  ) %>%
  filter(
  is.na(sex)
  ) %>%
  group_by(
    employee_id
  ) %>%
  summarise(
    missing = sum(is.na(sex))
  )
```

```
## # A tibble: 7 x 2
##   employee_id missing
##         <dbl>   <int>
## 1        1994      10
## 2        1995       9
## 3        1996      12
```

```
## 4         1997       11
## 5         1998       12
## 6         1999        7
## 7         2000       10
```

#Calculating the number of quarters for which the employees have missing data on race

```
practice_data %>%
  select(
    employee_id, race
  ) %>%
  filter(
     is.na(race)
  ) %>%
  group_by(
    employee_id
  ) %>%
  summarise(
    miss = sum(is.na(race))
  )
```

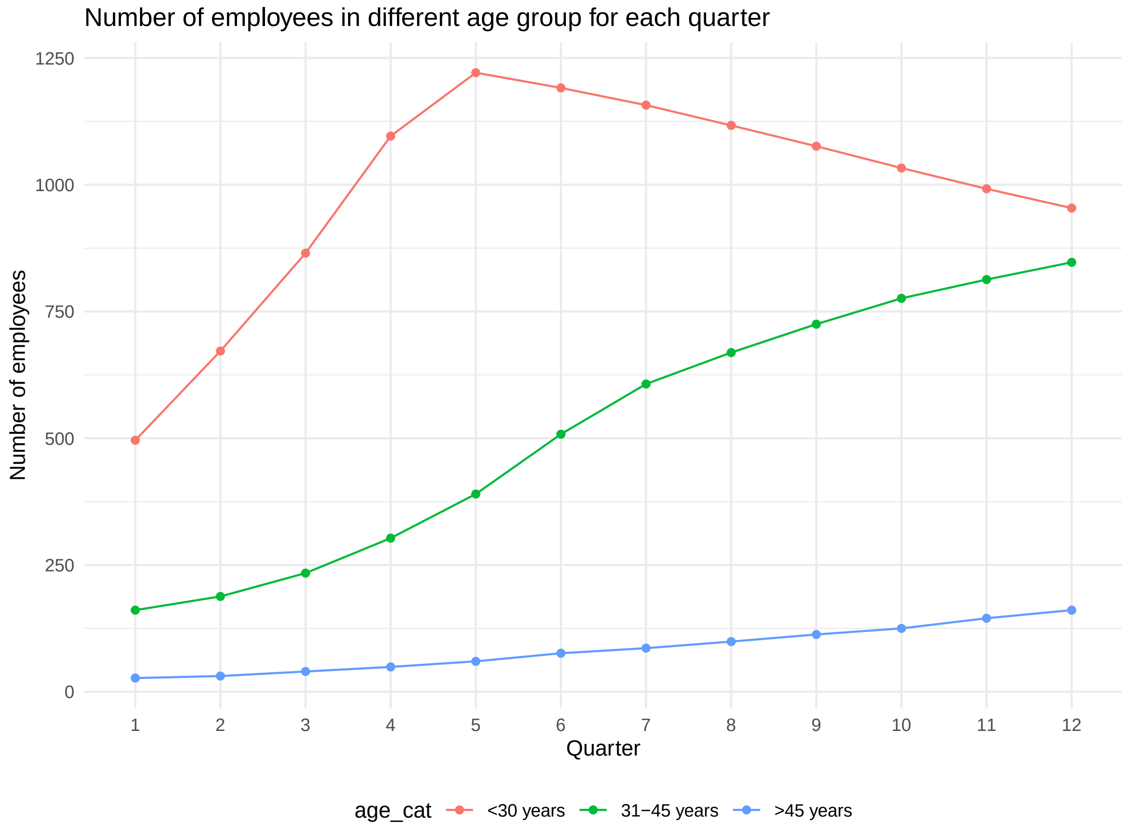
```
## # A tibble: 220 x 2
##    employee_id  miss
##          <dbl> <int>
##  1           8    10
##  2          10    12
##  3          13     9
##  4          22     9
##  5          36    12
##  6          38    12
##  7          48    10
##  8          49     7
##  9          51     8
## 10          55     9
## # ... with 210 more rows
```

#Calculating the number of employees in each age group for each quarter

```
emp_data = practice_data %>%
  mutate(
    quarter = factor(
    quarter),
    age_cat = factor(age_cat)
  )
```

```
emp_data = emp_data %>%
  select(
    employee_id, quarter, age_cat
  ) %>%
  group_by(
    quarter, age_cat
  ) %>%
  tally()
```

```r
e <- ggplot(emp_data, aes(x = quarter, y = n, group = age_cat)) +
  geom_line(aes(color = age_cat)) +
  geom_point(aes(color = age_cat)) + labs(x = "Quarter", y = "Number of employees", title = "Number of e
e
```



#Checking the trend in average salary over time

```r
practice_data %>%
  select(
    salary, quarter
  ) %>%
  group_by(
    quarter
  ) %>%
  summarise(
    avg_salary = mean(salary)
  )
```

```
## # A tibble: 12 x 2
##    quarter avg_salary
##      <dbl>      <dbl>
##  1       1     43628.
##  2       2     44274.
##  3       3     45021.
##  4       4     45531.
##  5       5     46133.
##  6       6     46948.
```

```
## 7       7     47780.
## 8       8     48667.
## 9       9     49562.
## 10     10     50498.
## 11     11     51433.
## 12     12     52376.
```

```
salary_data = practice_data %>%
  mutate(
    age_cat = factor(age_cat),
quarter = factor(quarter))
```

#Checking the trend in average salary over time by age group

```
salary_data = salary_data %>%
  select(
    salary, quarter, age_cat
  ) %>%
  group_by(
    quarter, age_cat
  ) %>%
  summarise(
    avg_salary = mean(salary)
  ) #%>%
  #pivot_wider(
    #names_from = age_cat, values_from = avg_salary
  #)
```

```
p <- ggplot(salary_data, aes(x = quarter, y = avg_salary, group = age_cat)) +
  geom_line(aes(color = age_cat)) +
  geom_point(aes(color = age_cat)) + labs(x = "Quarter", y = "Average Salary", title = "Trend in averag
p
```

#Checking the trend in mean health score over time

```
hc = practice_data %>%
  select(
    health_score, quarter
  ) %>%
  group_by(
    quarter
  ) %>%
  summarise(
    avg_score = mean(health_score)
  )
```

#Mean health score over time by age group

```
practice_data %>%
  select(
    health_score, quarter, age_cat
  ) %>%
  group_by(
    quarter, age_cat
  ) %>%
  summarise(
    avg_score = mean(health_score)
  )
```

```
## # A tibble: 36 x 3
```

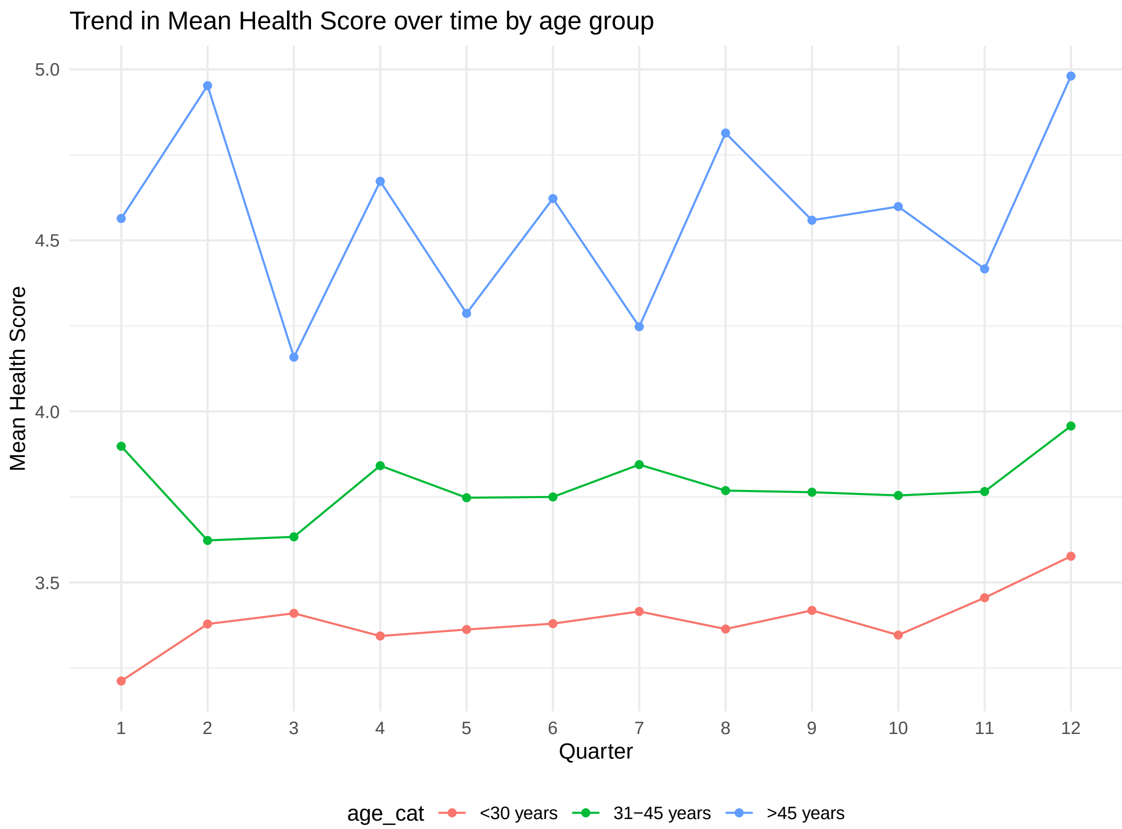
```
## # Groups:   quarter [12]
##    quarter age_cat avg_score
##      <dbl>   <dbl>     <dbl>
##  1       1       1      3.21
##  2       1       2      3.90
##  3       1       3      4.56
##  4       2       1      3.38
##  5       2       2      3.62
##  6       2       3      4.95
##  7       3       1      3.41
##  8       3       2      3.63
##  9       3       3      4.16
## 10       4       1      3.34
## # ... with 26 more rows
```

```r
health_sc = practice_data %>%
  mutate(
    age_cat = factor(age_cat),
quarter = factor(quarter))
```

```r
health_sc = health_sc %>%
  select(
    health_score, quarter, age_cat
  ) %>%
  group_by(
    quarter, age_cat
  ) %>%
  summarise(
    avg_score = mean(health_score)
  )
```

In group by and summarise, the number of rows is the number of groups, and the columns are the values you have summarised.

```r
p1 <- ggplot(health_sc, aes(x = quarter, y = avg_score, group = age_cat)) +
  geom_line(aes(color = age_cat)) +
  geom_point(aes(color = age_cat)) + labs(x = "Quarter", y = "Mean Health Score", title = "Trend in Mea
p1
```

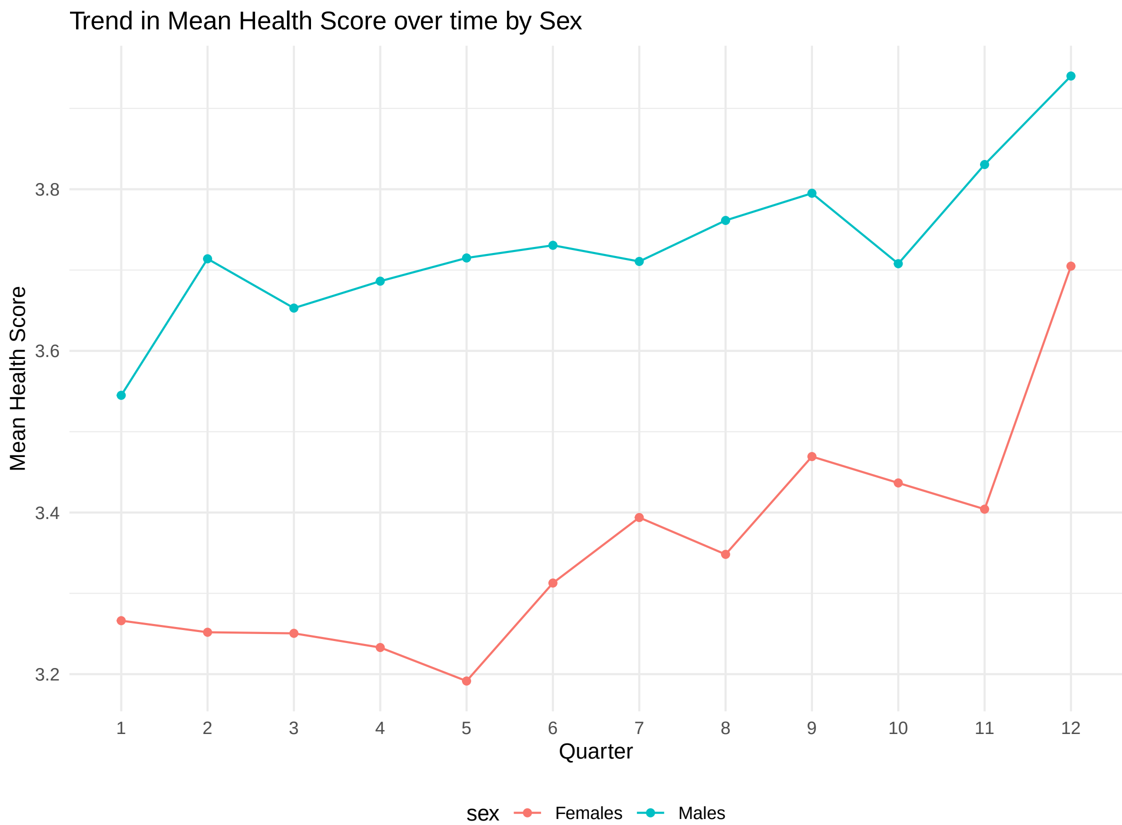


#Mean health score over time by sex

```
health_sex = practice_data %>%
  mutate(
    sex = factor(sex),
quarter = factor(quarter))
```

```
health_sex = health_sex %>%
  drop_na() %>%
  select(
    health_score, quarter, sex
  ) %>%
  group_by(
    quarter, sex
  ) %>%
  summarise(
    avg_score = mean(health_score)
  )
```

```
pq <- ggplot(health_sex, aes(x = quarter, y = avg_score, group = sex)) +
  geom_line(aes(color = sex)) +
  geom_point(aes(color = sex)) + labs(x = "Quarter", y = "Mean Health Score", title = "Trend in Mean Hea
pq
```

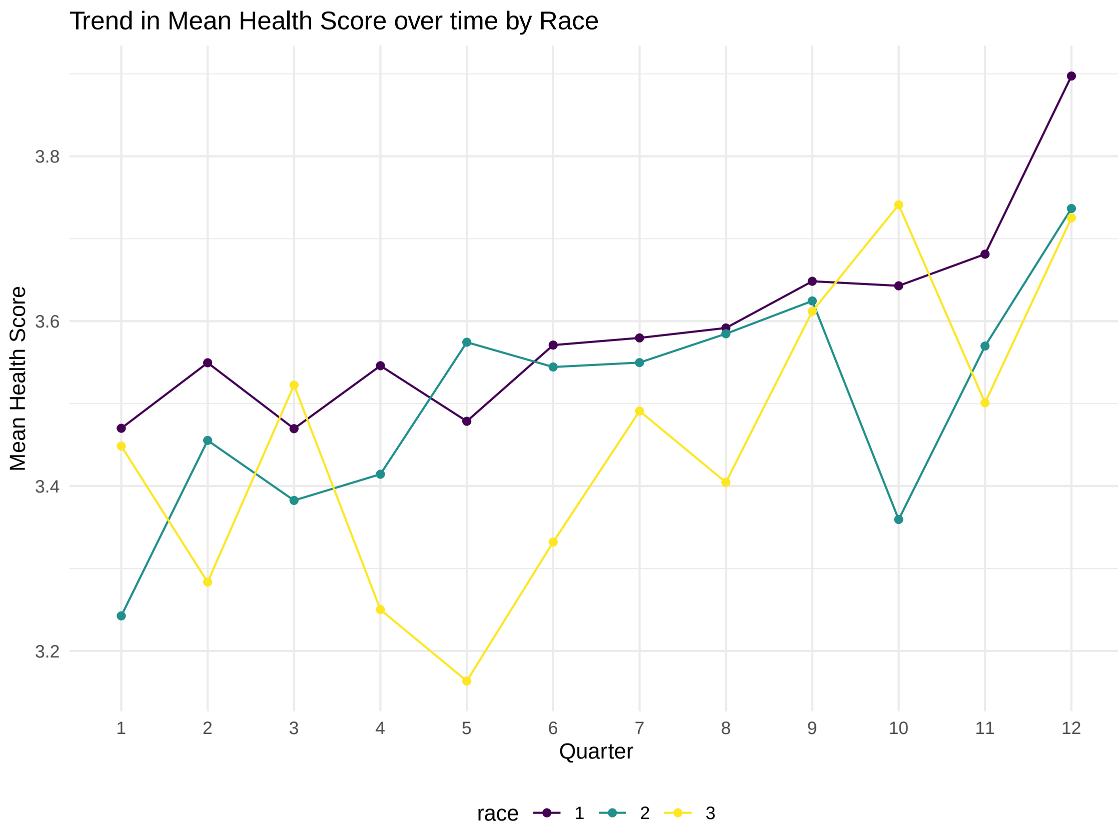


#Mean health score over time by race

```
health_race = practice_data %>%
  mutate(
    race = factor(race),
quarter = factor(quarter))
```

```
health_race = health_race %>%
  drop_na() %>%
  select(
    health_score, quarter, race
  ) %>%
  group_by(
    quarter, race
  ) %>%
  summarise(
    avg_score = mean(health_score)
  )
```

```
pe <- ggplot(health_race, aes(x = quarter, y = avg_score, group = race)) +
  geom_line(aes(color = race)) +
  geom_point(aes(color = race)) + labs(x = "Quarter", y = "Mean Health Score", title = "Trend in Mean He
pe
```



#Correcting the data quality issues

```
new_data = practice_data %>%
  drop_na() %>%
  filter(
    health_score <= 6,
    age >= 14, age <= 75
  )
```

#Checking the trend in mean health score over time in the corrected data

```
new_data %>%
  select(
    health_score, quarter
  ) %>%
  group_by(
    quarter
  ) %>%
  summarise(
    avg_score = mean(health_score)
  ) %>%
  knitr::kable()
```

quarter	avg_score
1	2.957220
2	3.062881

quarter	avg_score
3	3.053617
4	3.082009
5	3.080370
6	3.121949
7	3.192630
8	3.135304
9	3.177808
10	3.147629
11	3.234287
12	3.298995

#Mean health score over time by age group

```
new_data %>%
  select(
    health_score, quarter, age_cat
  ) %>%
  group_by(
    quarter, age_cat
  ) %>%
  summarise(
    avg_score = mean(health_score)
  ) %>%
knitr::kable()
```

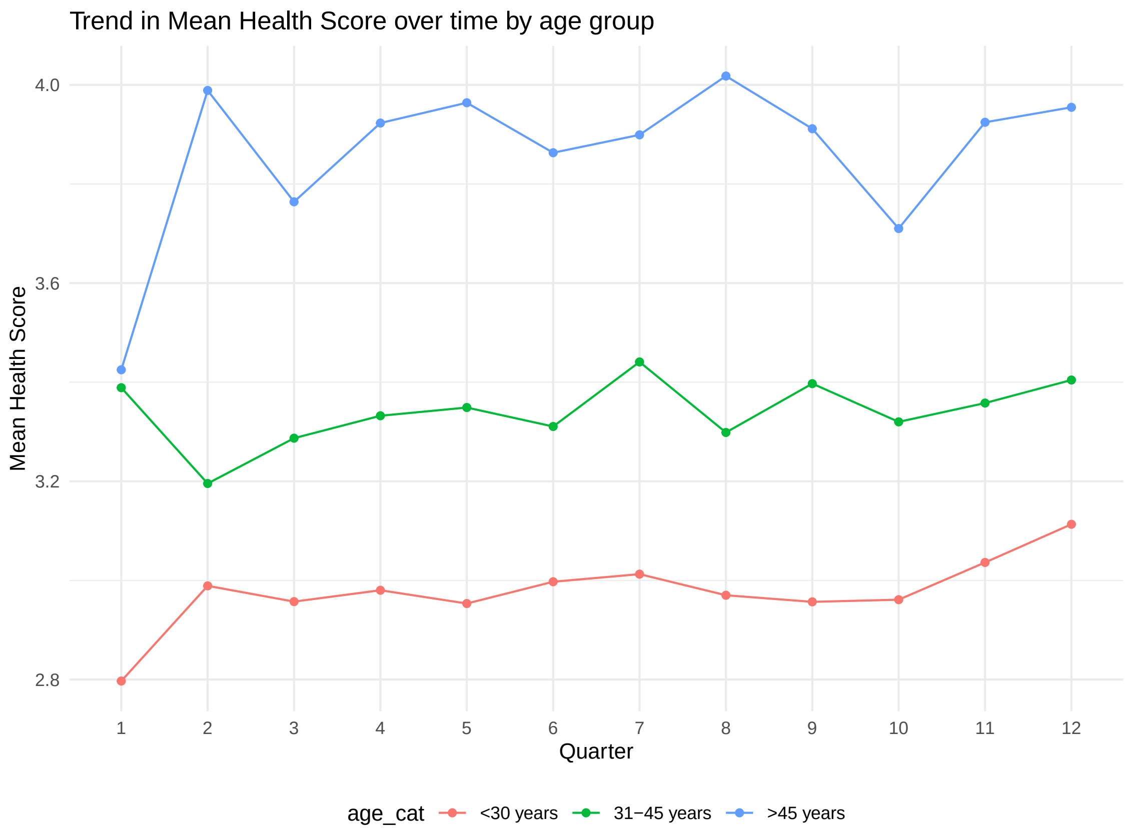
quarter	age_cat	avg_score
1	1	2.796936
1	2	3.388951
1	3	3.425028
2	1	2.989041
2	2	3.195572
2	3	3.988790
3	1	2.957211
3	2	3.287013
3	3	3.764033
4	1	2.980156
4	2	3.332257
4	3	3.922990
5	1	2.953362
5	2	3.348889
5	3	3.963946
6	1	2.997311
6	2	3.310626
6	3	3.863035
7	1	3.012671
7	2	3.440885
7	3	3.899116
8	1	2.970070
8	2	3.298552
8	3	4.017854

quarter	age_cat	avg_score
9	1	2.956768
9	2	3.397101
9	3	3.911494
10	1	2.961097
10	2	3.319968
10	3	3.710119
11	1	3.036243
11	2	3.357914
11	3	3.924474
12	1	3.113301
12	2	3.404411
12	3	3.954721

```
health_score = new_data %>%
  mutate(
    age_cat = factor(age_cat),
quarter = factor(quarter))
```

```
health_score = health_score %>%
  select(
    health_score, quarter, age_cat
  ) %>%
  group_by(
    quarter, age_cat
  ) %>%
  summarise(
    avg_score = mean(health_score)
  )
```

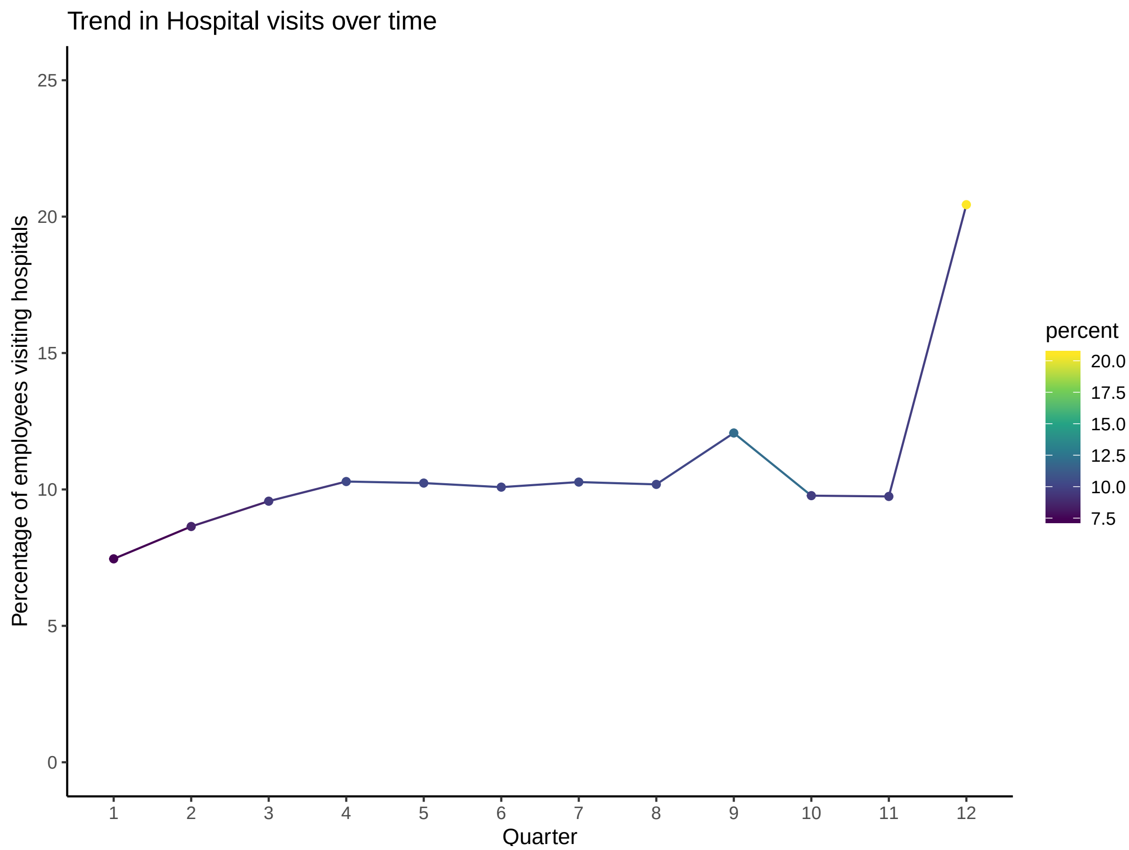
```
pn <- ggplot(health_score, aes(x = quarter, y = avg_score, group = age_cat)) +
  geom_line(aes(color = age_cat)) +
  geom_point(aes(color = age_cat)) + labs(x = "Quarter", y = "Mean Health Score", title = "Trend in Mea
pn
```



#Calculating the trend in hospital visits over time

```
hosp = practice_data %>%
  select(
    employee_id, hospital_visit, quarter
  ) %>%
  group_by(
    quarter
  ) %>%
  summarise(
    percent = (sum(hospital_visit)/n())*100
  )
```

```
s <- ggplot(hosp, aes(x = quarter, y = percent, color = percent)) + theme_classic() + geom_line() + geor
   scale_x_discrete(name = "Quarter", limits = c("1","2","3","4","5","6","7","8","9","10","11","12")) +
  scale_y_continuous(name = "Percentage of employees visiting hospitals",
                         breaks = seq(0, 25, 5),
                         limits = c(0, 25)) + labs(x = "Quarter", y = "Percentage of employees visitir
s
```



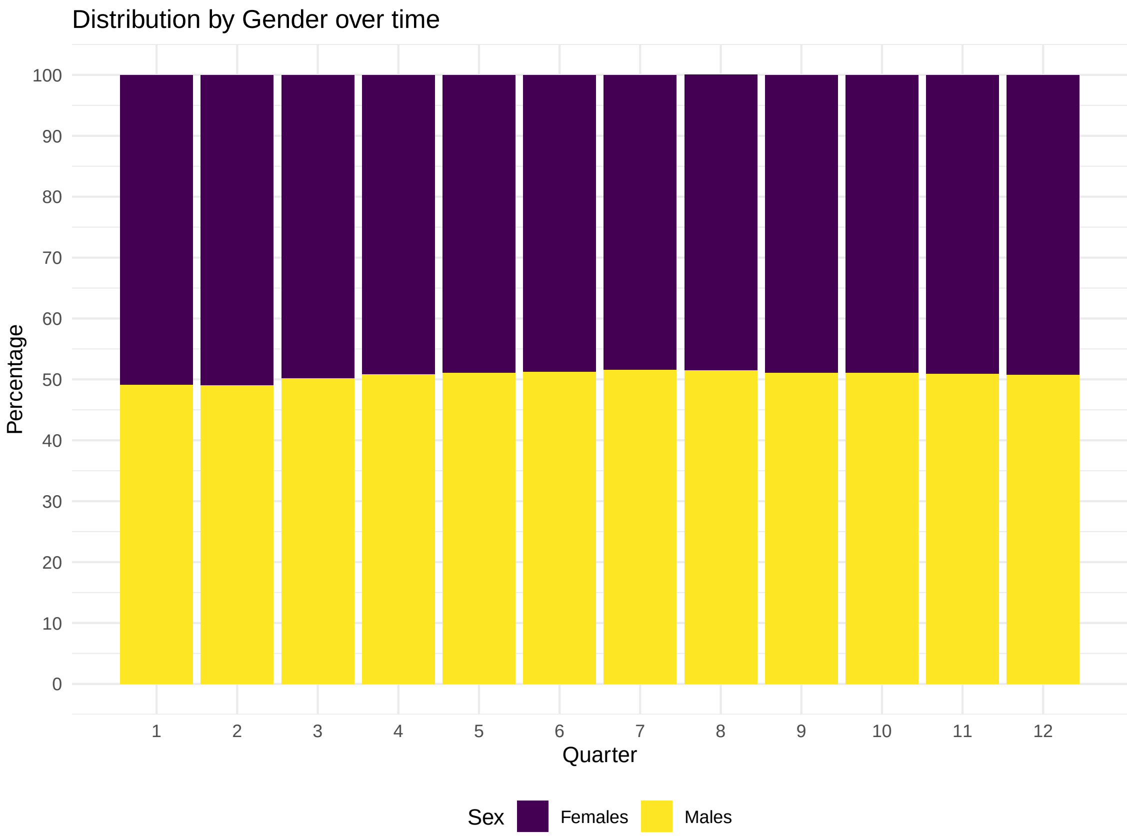
#Calculating the distribution by gender over time

```
w = practice_data %>%
  drop_na() %>%
  mutate(
    sex = factor(sex)
  ) %>%
  select(
     employee_id, quarter, sex
  ) %>%
  group_by(
    quarter, sex
  ) %>%
summarise(n = n()) %>%
mutate(freq = n / sum(n)*100)
w
```

```
## # A tibble: 24 x 4
## # Groups:   quarter [12]
##     quarter sex       n  freq
##       <dbl> <fct> <int> <dbl>
##   1       1 0       305  50.8
##   2       1 1       295  49.2
##   3       2 0       399  50.9
##   4       2 1       385  49.1
##   5       3 0       503  49.8
##   6       3 1       507  50.2
```

```
## 7         4 0        632   49.1
## 8         4 1        655   50.9
## 9         5 0        726   48.9
## 10        5 1        759   51.1
## # ... with 14 more rows
```

```
t = w %>%
  ggplot(aes(x = quarter, y = freq, fill = sex)) + geom_bar(stat = "identity", legend = c("Female", "Mal
  scale_y_continuous(name = "Percentage",
                         breaks = seq(0, 100, 10),
                         limits = c(0, 100)) + labs(x = "Quarter", y = "Percentage" , title = "Distril
t
```



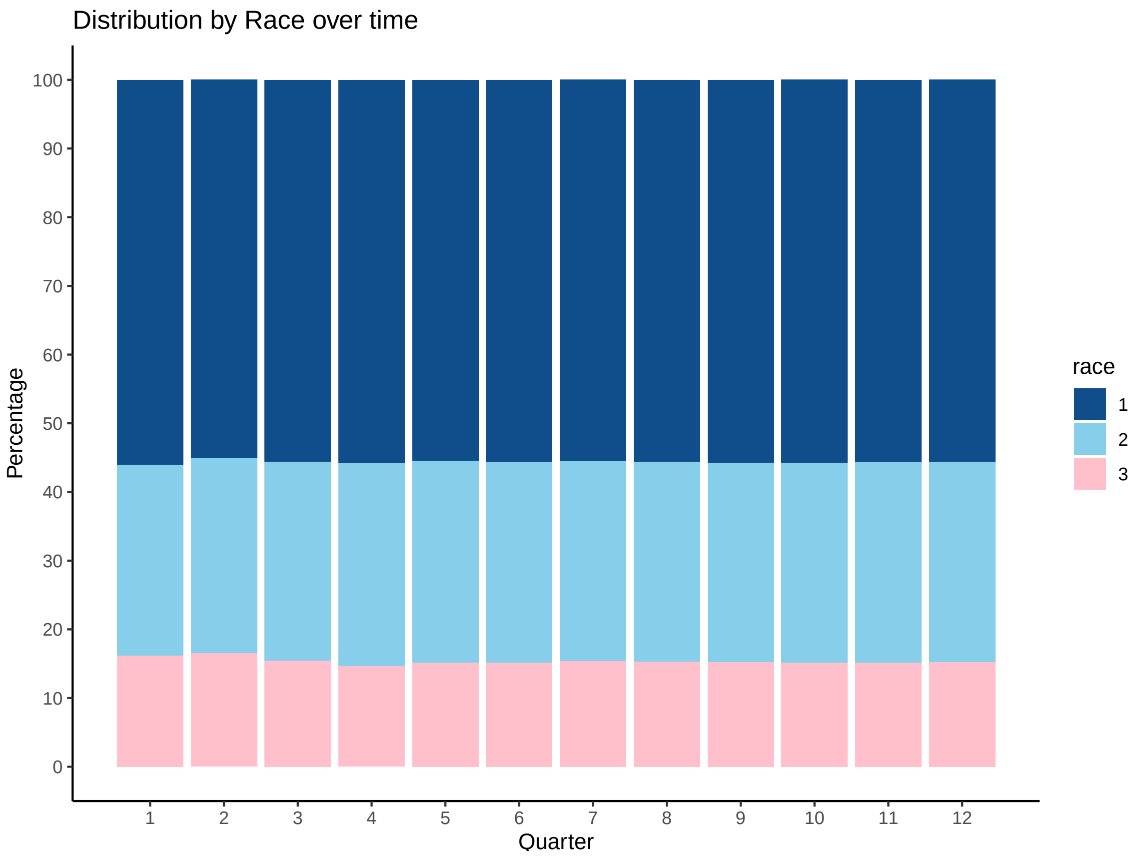
#Calculating the distribution by race over time

```
j = practice_data %>%
  drop_na() %>%
  mutate(
    race = factor(race)
  ) %>%
  select(
     quarter, race
  ) %>%
  group_by(
    quarter, race
  ) %>%
  summarise(n = n()) %>%
```

```
mutate(freq = n / sum(n)*100)
j
```

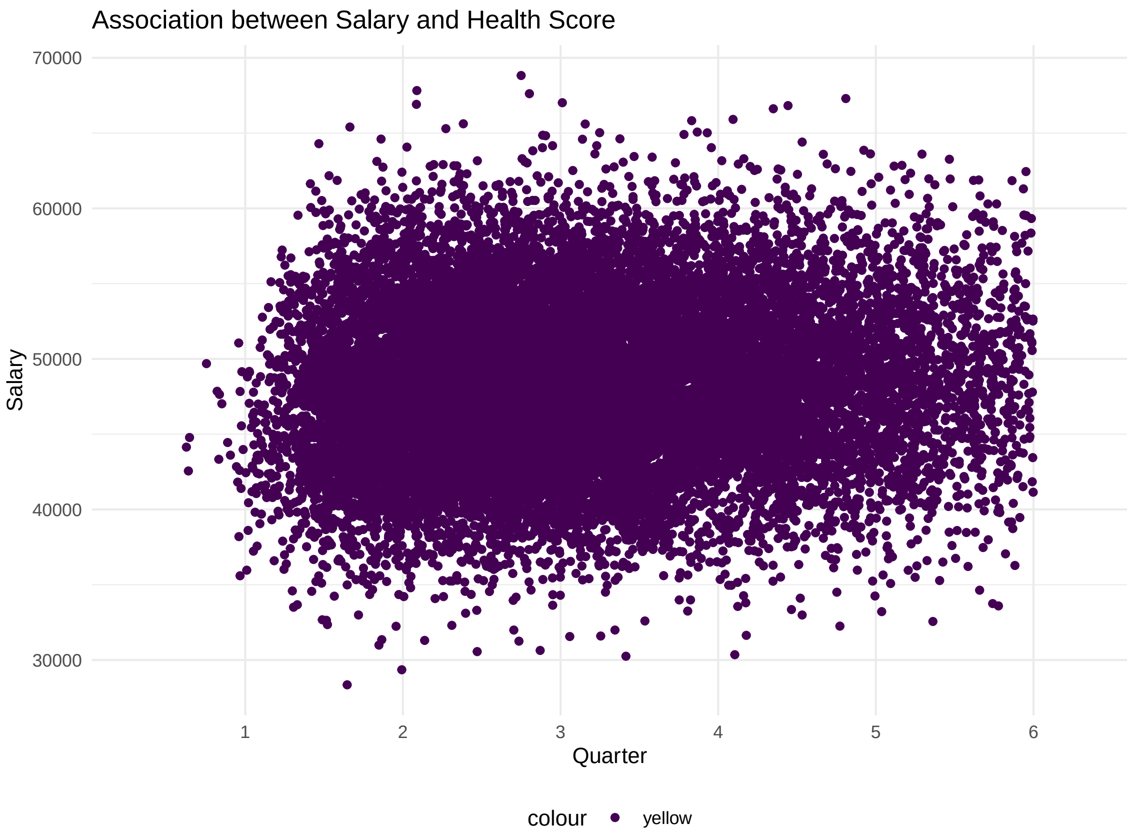
```
## # A tibble: 36 x 4
## # Groups:   quarter [12]
##    quarter race      n  freq
##      <dbl> <fct> <int> <dbl>
##  1       1 1       336  56.
##  2       1 2       167  27.8
##  3       1 3        97  16.2
##  4       2 1       432  55.1
##  5       2 2       222  28.3
##  6       2 3       130  16.6
##  7       3 1       561  55.5
##  8       3 2       293  29.0
##  9       3 3       156  15.4
## 10       4 1       718  55.8
## # ... with 26 more rows
```

```
k = j %>%
  ggplot(aes(x = quarter, y = freq, fill = race)) + geom_bar(stat = "identity") +
  theme_classic() + scale_fill_manual(values = c("dodgerblue4","skyblue","pink")) + scale_x_discrete(nam
  scale_y_continuous(name = "Percentage",
                         breaks = seq(0, 100, 10),
                         limits = c(0, 100)) + labs(x = "Quarter", y = "Percentage" , title = "Distrib
k
```



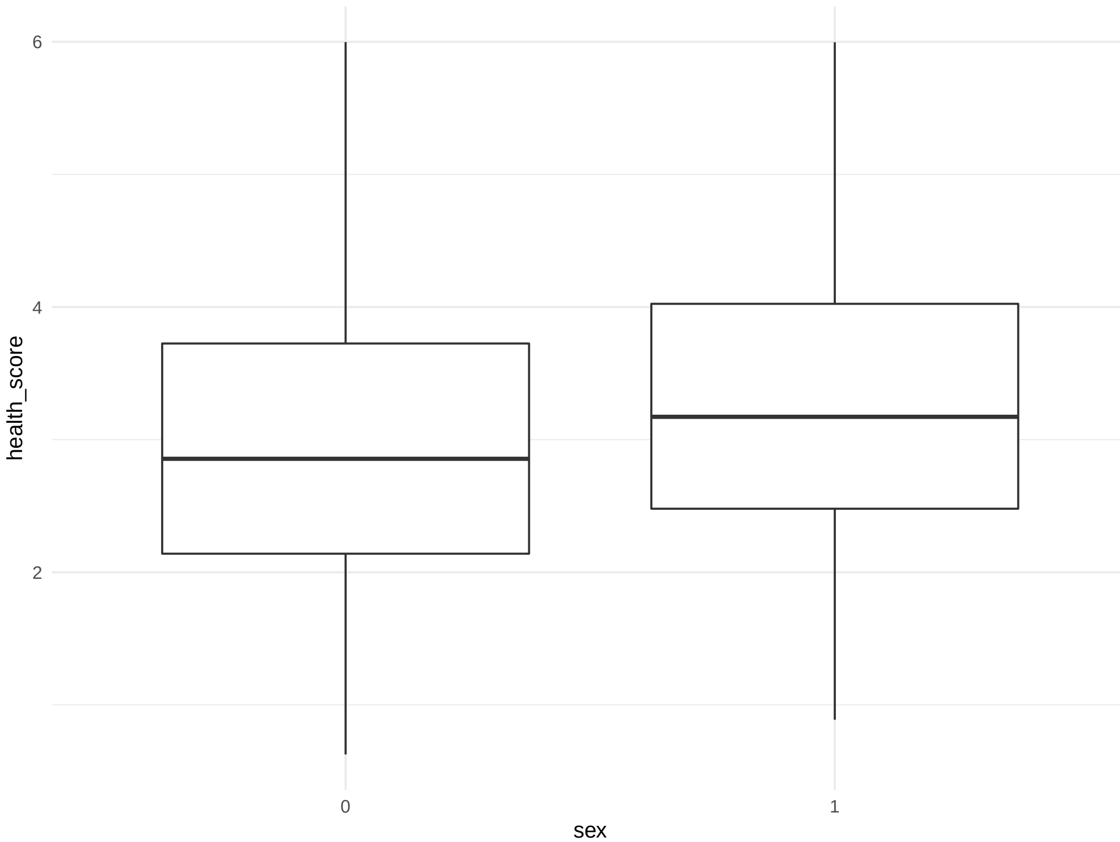
#Scatterplot between salary and health score

```
practice_data %>%
  filter(
    health_score <= 6
  ) %>%
  ggplot(
    aes(x = health_score, y = salary)
  ) + geom_point(aes(color = "yellow")) + labs(
    x = "Heath Score", y = "Salary", title = "Association between Salary and Health Score"
  ) + scale_x_discrete(name = "Quarter", limits = c("1","2","3","4","5","6","7","8","9","10","11","12")
```



#Box-plot showing the Mean health score for both genders

```
practice_data %>%
  drop_na() %>%
  filter(
    health_score <= 6
  ) %>%
  mutate(
    sex = factor(sex)
  ) %>%
  ggplot(
    aes(
      x = sex, y = health_score
    )
  ) + geom_boxplot()
```



#Checking the association between sex and health score using two sample t-test

```
t.test(health_score ~ sex, data = new_data)
```

```
##
##  Welch Two Sample t-test
##
## data:  health_score by sex
## t = -17.726, df = 15816, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.3343974 -0.2678069
## sample estimates:
## mean in group 0 mean in group 1
##        2.995977        3.297079
```

#Checking the association between hospital visit and health score using two sample t-test

```
t.test(health_score ~ hospital_visit, data = new_data)
```

```
##
##  Welch Two Sample t-test
##
## data:  health_score by hospital_visit
## t = -26.924, df = 2278.4, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
```

```
## 95 percent confidence interval:
##  -0.7129211 -0.6161208
## sample estimates:
## mean in group 0 mean in group 1
##        3.076413        3.740934
```

#Scatterplot showing the association between age and health score

```
new_data %>%
  filter(
    age <= 60
  ) %>%
ggplot(
  aes(
    x = age, y = health_score
  )
) + geom_point(
  aes(color = "pink")
)
```

#Checking the association between race and health score

```
av <- aov(health_score ~ race, data = new_data)
summary(av)
```

```
##                Df Sum Sq Mean Sq F value  Pr(>F)
## race            1     12  11.726   10.06 0.00152 **
```

```
## Residuals   15876  18513   1.166
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

#Creating 2*2 tables

```
practice_data %>%
  filter(
    age_cat != 3
  ) %>%
  mutate(
    sal_cat = case_when(
      salary < 40000 ~ "First",
      salary > 40000 ~ "Second"
    )
  ) %>%
  janitor::tabyl(age_cat, sal_cat)
```

```
##  age_cat First Second
##        1   678  11192
##        2   337   5884
```

```
practice_data %>%
  filter(
    age_cat != 3
  ) %>%
  mutate(
    sal_cat = case_when(
      salary < 40000 ~ "First",
      salary > 40000 ~ "Second"
    )
  ) %>%
  group_by(age_cat,sal_cat) %>%
  summarise(
    n = n()
  ) %>%
  pivot_wider(
    names_from = sal_cat, values_from = n
  )
```

```
## # A tibble: 2 x 3
## # Groups:   age_cat [2]
##   age_cat First Second
##     <dbl> <int>  <int>
## 1       1   678  11192
## 2       2   337   5884
```

#General summaries

```
practice_data %>%
  group_by(
    quarter
```

```r
  ) %>%
  summarise(
    n = n(),
    mean_age = mean(age),
    median_age = median(age),
    mean_salary = mean(salary),
    median_salary = median(salary)
    )
```

```
## # A tibble: 12 x 6
##    quarter     n mean_age median_age mean_salary median_salary
##      <dbl> <int>    <dbl>      <dbl>       <dbl>         <dbl>
##  1       1   684     28.8       26.6      43628.        43612.
##  2       2   891     28.6       26.5      44274.        44254.
##  3       3  1139     28.6       26.8      45021.        45002.
##  4       4  1448     28.8       27.1      45531.        45464.
##  5       5  1671     29.3       27.6      46133.        46119.
##  6       6  1775     29.9       28.1      46948.        46915.
##  7       7  1850     30.5       28.5      47780.        47749.
##  8       8  1885     30.9       28.9      48667.        48664.
##  9       9  1914     31.4       29.3      49562.        49603.
## 10      10  1934     31.8       29.6      50498.        50548.
## 11      11  1950     32.2       29.9      51433.        51512.
## 12      12  1962     32.6       30.2      52376.        52440.
```

Notes: Grouped mutate Summarizing collapses groups into single data points. In contrast, using mutate() in conjuntion with group_by() will retain all original data points and add new variables computed within groups.

```r
practice_data %>%
  group_by(
    quarter
  ) %>%
  mutate(
    hs = round(health_score, digits = 0)
  )
```

```
## # A tibble: 19,103 x 11
## # Groups:   quarter [12]
##    observation_num~ quarter employee_id   sex  race   age hospital_visit
##               <dbl>   <dbl>       <dbl> <dbl> <dbl> <dbl>          <dbl>
##  1                1       1           1     0     3  27.3              0
##  2                2       2           1     0     3  27.8              0
##  3                3       3           1     0     3  28.1              0
##  4                4       4           1     0     3  28.3              0
##  5                5       5           1     0     3  28.6              0
##  6                6       6           1     0     3  28.8              0
##  7                7       7           1     0     3  29.1              0
##  8                8       8           1     0     3  29.3              0
##  9                9       9           1     0     3  29.6              0
## 10               10      10           1     0     3  29.8              0
## # ... with 19,093 more rows, and 4 more variables: salary <dbl>,
## #   health_score <dbl>, age_cat <dbl>, hs <dbl>
```

Notes: Window functions The previous example used mean() to compute the mean within each group, which was then subtracted from the observed max tempurature. mean() takes n inputs and produces a single output.

Window functions, in contrast, take n inputs and return n outputs, and the outputs depend on all the inputs. There are several categories of window functions; you're most likely to need ranking functions and offsets, which we illustrate below.

```
practice_data %>%
  group_by(quarter, sex) %>%
  mutate(
    salary_rank = min_rank(salary)
  )
```

```
## # A tibble: 19,103 x 11
## # Groups:   quarter, sex [36]
##    observation_num~ quarter employee_id   sex  race   age hospital_visit
##               <dbl>   <dbl>       <dbl> <dbl> <dbl> <dbl>          <dbl>
##  1                1       1           1     0     3  27.3              0
##  2                2       2           1     0     3  27.8              0
##  3                3       3           1     0     3  28.1              0
##  4                4       4           1     0     3  28.3              0
##  5                5       5           1     0     3  28.6              0
##  6                6       6           1     0     3  28.8              0
##  7                7       7           1     0     3  29.1              0
##  8                8       8           1     0     3  29.3              0
##  9                9       9           1     0     3  29.6              0
## 10               10      10           1     0     3  29.8              0
## # ... with 19,093 more rows, and 4 more variables: salary <dbl>,
## #   health_score <dbl>, age_cat <dbl>, salary_rank <int>
```

This sort of ranking is useful when filtering data based on rank

```
practice_data %>%
  group_by(quarter, sex) %>%
    filter(
   min_rank(salary) < 2
    )
```

```
## # A tibble: 36 x 10
## # Groups:   quarter, sex [36]
##    observation_num~ quarter employee_id   sex  race   age hospital_visit
##               <dbl>   <dbl>       <dbl> <dbl> <dbl> <dbl>          <dbl>
##  1             3974       5         416     1    NA  30.5              0
##  2             3975       6         416     1    NA  30.7              0
##  3             4324       9         452     0     2  42.0              0
##  4             4325      10         452     0     2  42.3              0
##  5             4326      11         452     0     2  42.5              1
##  6             4327      12         452     0     2  42.8              0
##  7             4411       6         462     0     2  31.3              0
##  8             4412       7         462     0     2  31.5              0
##  9             4413       8         462     0     2  31.8              0
## 10             7712       1         806     0     3  47.6              0
## # ... with 26 more rows, and 3 more variables: salary <dbl>,
## #   health_score <dbl>, age_cat <dbl>
```

Notes:

Strings and Factors: The most frequent operation involving strings is to search for an exact match. You can use str_detect to find cases where the match exists (often useful in conjunction with filter), and you can use str_replace to replace an instance of a match with something else (often useful in conjunction with mutate)

str_detect(string_vec, “ˆi think”) - Starts with i think str_detect(string_vec, “i think$”) - ends with i think str_detect(string_vec,“[Bb]ush”) str_detect(string_vec, “[1][a-zA-Z]”) str_detect(string_vec, “7.11”) The character . matches any character. Some characters are “special”. These include [and], (and), and .. If you want to search for these, you have to indicate they’re special using . Unfortunately, is also special, so things get weird. str_detect(string_vec, “\[”) - search for an actual character i.e, the open bracket [

*imp: select(-contains"")

example for separate and str_replace:

data_marj = table_marj %>% select(-contains(“P Value”)) %>% # pivot_longer(-State, names_to = “age_year”, values_to = “percent”) %>%

separate(age_year, into = c(“age”, “year”), sep = “\(”) %>% #seperating the age_year into age and year, seperated by an open parenthesis. sep = “\(” means separate from where there is an (. We have to give \ since (is a special character.

mutate(year = str_replace(year, “\)”, "“), #After the above step, year had a) and we would want to remove that. We can do that by replacing that with a blank. str_replace(year,”\)“,”“) is replacing) with a black. Again we used \ since) is a spl character. percent = str_replace(percent,”[a-c]”, ””), $thereisacharactera, b, orcattheendofthepercentvalueswhichwewouldwanttoremove.Toremove, usestr_replace, and[a$ c] shows the presence of a-c at the end of the percent string and we want to replace that with a blank. percent = as.numeric(percent)) %>% filter(!(State %in% c(“Total U.S.”, “Northeast”, “Midwest”, “South”, “West”)))

#Interactivity

practice_data %>% drop_na() %>% select(quarter, race, health_score) %>% mutate(race = factor(race)) %>% group_by(race, quarter) %>% summarise(mean_score = mean(health_score)) %>% plot_ly(x = ~quarter, y = ~ mean_score, color = ~race, type = “bar”)

```
age_function = function(x){
  tibble(
    mean = mean(x),
    minimum = min(x),
    maximum = max(x)
  )
}
```

```
a = practice_data %>%
  select(
    age, salary, health_score
  )
```

```
age_function(x = a$age)
```

```
## # A tibble: 1 x 3
##    mean minimum maximum
##   <dbl>   <dbl>   <dbl>
## 1  30.6       7     172
```

[1] 0-9

```r
age_function(x = a$health_score)
```

```
## # A tibble: 1 x 3
##    mean minimum maximum
##   <dbl>   <dbl>   <dbl>
## 1  3.59   0.627      10
```

```r
age_function(x = a$salary)
```

```
## # A tibble: 1 x 3
##     mean minimum maximum
##    <dbl>   <dbl>   <dbl>
## 1 48298.  28351.  68826.
```

```r
practice_data1 =
  practice_data %>%
  select(employee_id, age, hospital_visit, salary, health_score)
```

```r
nested_data = nest(practice_data1, data = age:health_score)
```

```r
nested_data$data[[1]]
```

```
## # A tibble: 12 x 4
##      age hospital_visit salary health_score
##    <dbl>          <dbl>  <dbl>        <dbl>
##  1  27.3              0 36907.         3.70
##  2  27.8              0 37907.         4.98
##  3  28.1              0 38907.         4.01
##  4  28.3              0 39907.         2.34
##  5  28.6              0 40907.         2.11
##  6  28.8              0 41907.         1.46
##  7  29.1              0 42907.         4.73
##  8  29.3              0 43907.         2.34
##  9  29.6              0 44907.         2.76
## 10  29.8              0 45907.         2.83
## 11  30.1              0 46907.         1.80
## 12  30.3              0 47907.         1.70
```

```r
nested_data %>%
  unnest()
```

```
## # A tibble: 19,103 x 5
##    employee_id   age hospital_visit salary health_score
##          <dbl> <dbl>          <dbl>  <dbl>        <dbl>
##  1           1  27.3              0 36907.         3.70
##  2           1  27.8              0 37907.         4.98
##  3           1  28.1              0 38907.         4.01
##  4           1  28.3              0 39907.         2.34
##  5           1  28.6              0 40907.         2.11
##  6           1  28.8              0 41907.         1.46
```

```
## 7            1  29.1              0 42907.         4.73
## 8            1  29.3              0 43907.         2.34
## 9            1  29.6              0 44907.         2.76
## 10           1  29.8              0 45907.         2.83
## # ... with 19,093 more rows
```

```
lm_func = function(df) {
  lm(health_score ~ age, data = df)
}
```

```
lm_func(df = nested_data$data[[1]])
```

```
##
## Call:
## lm(formula = health_score ~ age, data = df)
##
## Coefficients:
## (Intercept)          age
##     23.1901      -0.7016
```

```
#map(nested_data$data, lm_func)
```