# Data Wrangling (2)

TRSM R Bootcamp

## Playground dataframe

We will be using **Playground** dataset to demonstrate some functionality that could be achieved via pipes and other functions.

**Exercises**:

1. What is the max and min value in playground dataframe for age column?
2. Does this age range make sense?
3. What are all the unique values in the age column?

```
# Filter out unusual ages
playground <- playground %>%
  filter(age < 100)
```

### Modifications to dataframe

The code below uses ***lubridate*** library to reformat the treatment_date column in Playground. As beginners, you do not need to know the details of how this process works.

```
# Convert to date format
library(lubridate)

playground <- playground %>%
  mutate(treatment_date = mdy(treatment_date) )

# Add year, day, and month variables
playground <- playground %>%
  mutate(year = year(treatment_date),
         month = month(treatment_date),
         day=day(treatment_date))
```

## groupby and summarise

These two functions should be used with each other since summarise does not make sense without group_by.

### summarise(df, variable_name=condition)

- df: Dataset used to construct the summary statistics
- variable_name=condition: Formula to create the new variable

### groub_by():

- It runs operations on other columns based on this column's entries.

## Case Study

we would like to know the correlation between SaleType to SalePrice if there is any.

```
1. Data is passed via pipe
2. Data is grouped based on lgID column
3. The mean of SalePrice for each of the unique values in SaleType has been calculated
```

By observing the dataframe generated by running the code, we can see that the house where their SaleType equals **"New"** has the highest average in SalePrice. One obvious conclusion will be that given a house is built recently it will have a higher price compared to other sale types.

```
housing_df %>%
    group_by(SaleType) %>%
    summarise(mean_run = mean(SalePrice)) %>%
    arrange(-mean_run)
```

```
## # A tibble: 9 x 2
##   SaleType mean_run
##   <chr>       <dbl>
## 1 New       274945.
## 2 Con       269600
## 3 CWD       210600
## 4 ConLI     200390
## 5 WD        173402.
## 6 COD       143973.
## 7 ConLw     143700
## 8 ConLD     138781.
## 9 Oth       119850
```

## Case Study:

Getting the frequency of different values within the Neighborhood column.

```
1. Group by Neighborhood column since that is our column of reference.
2. Use summarise and n() function to get the number of repitions for different unique values within the
3. Sort in descending order based on freq column (where we stored the frequencies)
4. Filter for the repitions of higher than 80
```

```
housing_df %>%
  group_by(Neighborhood) %>%
  summarise(freq=n()) %>%
  arrange(-freq) %>%
  filter(freq > 80)
```

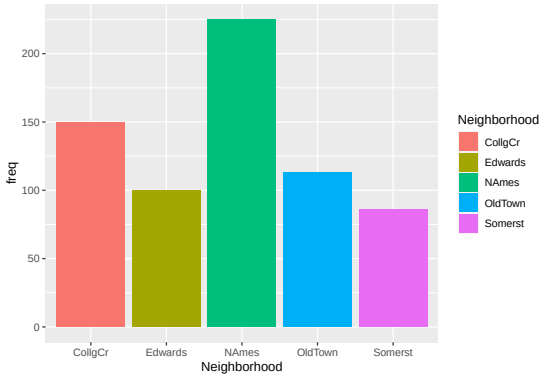
```
## # A tibble: 5 x 2
##   Neighborhood  freq
##   <chr>        <int>
## 1 NAmes          225
## 2 CollgCr        150
## 3 OldTown        113
## 4 Edwards        100
## 5 Somerst         86
```

## Case Study

We would like to find out which Neghbourhoods have the most repetitions (they are repeated in entries of Neighborhood column).

1. Follow the same process as the previous code.
2. Filter the entries where freq > 80, meaning there have more than 80 entries (rows) where a certain va

```
housing_df %>%
  group_by(Neighborhood) %>%
  summarise(freq=n()) %>%
  arrange(-freq) %>%
  filter(freq > 80) %>%
  # fill=Neighborhood -> applies colours to the bar chart
  ggplot(aes(x=Neighborhood, y=freq, fill=Neighborhood)) + geom_bar(stat = 'identity')
```




### Useful functions with summarise

- **Basic**
  - mean(): Average of vector x
  - median(): Median of vector x
  - sum(): Sum of vector x
- **Variation**
  - sd(): standard deviation of vector x
  - IQR(): Interquartile of vector x
- **Range**
  - min(): Minimum of vector x
  - max(): Maximum of vector x
  - quantile(): Quantile of vector x
- **Position**
  - first(): First observation of the group
  - last(): Last observation of the group
  - nth(): nth observation of the group
- **Count**
  - n(): Count the number of rows
  - n_distinct(): Count the number of distinct observations

## Case Study:

Which playground equipment causes the most injuries?

1. We have to specify our column of interest by group_by
2. Use summarize to count the number of occurances
3. Use arrange to list the products in a descending order

```
playground %>%
  group_by(injury_label) %>% # 1
  summarise(freq = n() ) %>% # 2
```

```
  arrange(-freq )%>% # 3
  filter(freq > 1000)
```

```
## # A tibble: 8 x 2
##   injury_label      freq
##   <chr>            <int>
## 1 FRACTURE         31256
## 2 LACERATION       10380
## 3 CONTUSIONS, ABR. 10253
## 4 STRAIN, SPRAIN    7635
## 5 INTERNAL INJURY   7580
## 6 OTHER             5941
## 7 CONCUSSION        2245
## 8 DISLOCATION       1002
```

**Exercise**: What if we wanted the relative frequency? (Hint: sum of all the values should equal 1)

```
playground %>%
  group_by(____) %>% # 1
  summarise(__ = __/ ____) %>% # 2
  arrange( ___ ) # 3
```

```
## Error: <text>:2:12: unexpected input
## 1: playground %>%
## 2:   group_by(_
##               ^
```

# More examples

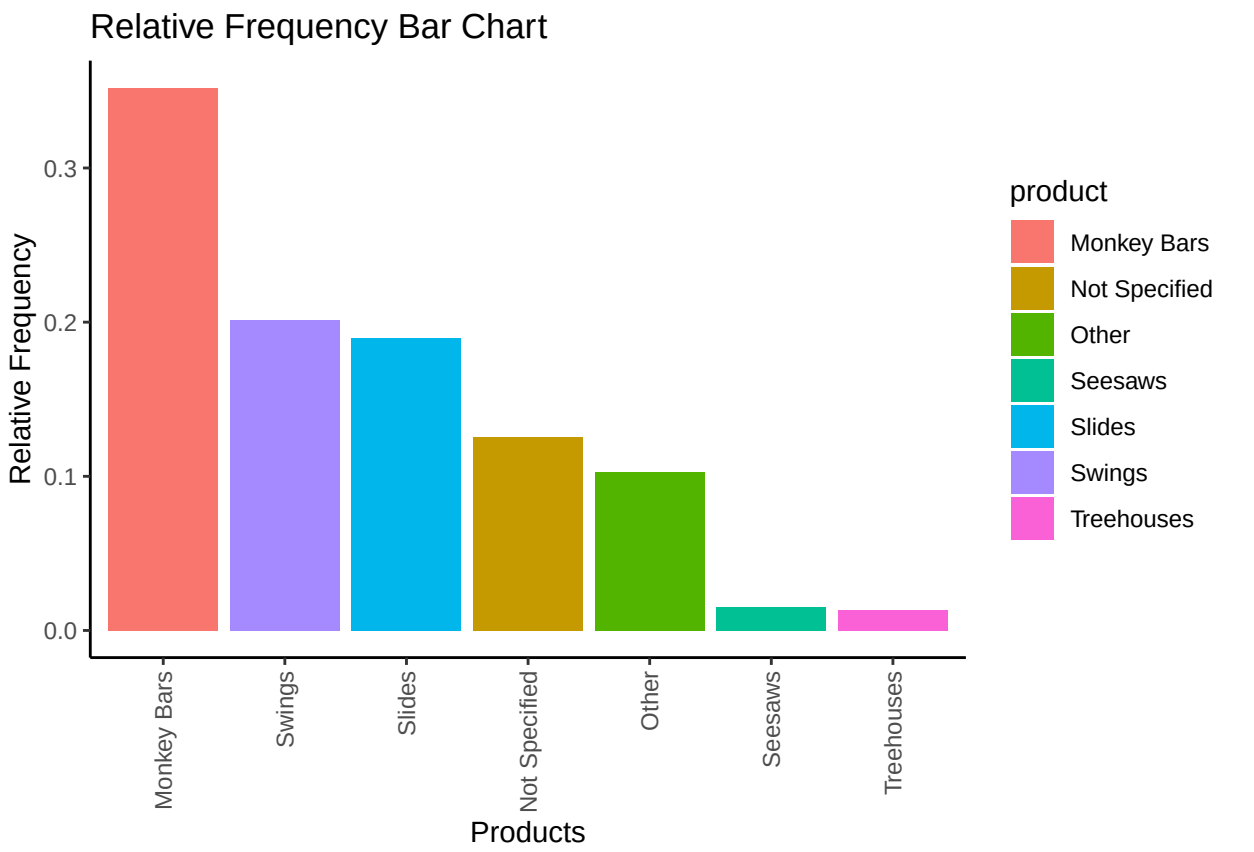
## Plotting the relative frequency of products

```
playground %>%
  group_by(product) %>% # 1
  summarise(freq = n()/ dim(playground)[1]) %>% # 2
  arrange( -freq ) # 3
```

```
## # A tibble: 7 x 2
##   product         freq
##   <chr>          <dbl>
## 1 Monkey Bars   0.352
## 2 Swings        0.201
## 3 Slides        0.190
## 4 Not Specified 0.125
## 5 Other         0.103
## 6 Seesaws       0.0153
## 7 Treehouses    0.0131
```

```
# Plotting the relative frequency of products
playground %>%
  group_by(product) %>%
  summarise(freq=n() / dim(playground)[1]) %>%
  ggplot(
    aes(x=reorder(product, -freq), y=freq, fill=product)) +
  geom_bar(stat="identity") +
  theme_classic() +
```

```
  theme(axis.text.x=element_text(angle=90,hjust=1,vjust=0.5))+
  labs(
    x="Products",
    y="Relative Frequency",
    title=paste("Relative Frequency Bar Chart")
  )
```




**Exercise**: What percentage of people who hurt themselves with Slides, hurt their heads?

```
1. Filter data by slides
2. Calculate the relative frequency of the data
3. Choose the entries with injury == HEAD
```

```
playground %>%
  filter(____ == ____) %>%  # 1
  group_by(____) %>% # 2.1
  summarise(freq=__) %>%  # 2.2
  mutate(freq=____) %>% # 2.3
  filter(___ == ___) # 3
```

**Exercise**: What playground equipment causes the most dislocations?

```
1. Filter by injury_label of dislocation
2. Make a column to count the number dislocations for each product
3. Sort in the descending order
```

```
playground %>%
  filter(injury_label == ____) %>%
  group_by(____) %>%
```

```
  summarise(num_injuries = __ ) %>%
  arrange(-num_injuries)
```

**Exercise**: What playground injury is most common for people over 30 years old?

1. Filter by age being bigger than 30
2. Count the number for each injury
3. Sort in the descending order

**Exercises**:

1. What is the average SalePrice for each unique value in Alley column?
2. What is the maximum value of LotFrontage for each unique value in Alley?
3. What is the maximum SalePrice for each unique value in Alley column?
4. What is the most repeated value of OverQual for each unique value in Alley?
5. Get the mean price and mode of OverQual for each of the unique values in Alley then sort in descending order based on Average SalePrice