Overview of Introduction to the Tidyverse (Datacamp)

Dan Sanchez

Table of Contents

[Preface 1](#_Toc96087956)

[Data Wrangling 2](#_Toc96087957)

[Filter Verb 2](#_Toc96087958)

[Arrange Verb 2](#_Toc96087959)

[Mutate Verb 4](#_Toc96087960)

[Data Visualization 5](#_Toc96087961)

[Scatterplots 5](#_Toc96087962)

[Log Scales 7](#_Toc96087963)

[Faceting 10](#_Toc96087964)

[Grouping and Summarizing 12](#_Toc96087965)

[Summarize Verb 12](#_Toc96087966)

[Group\_by Verb 13](#_Toc96087967)

[Visualizing Summarized Data 14](#_Toc96087968)

[Scatterplots 14](#_Toc96087969)

[Line Plots 17](#_Toc96087970)

[Bar Plots 19](#_Toc96087971)

[Histograms 21](#_Toc96087972)

[Boxplots 23](#_Toc96087973)

[Summary 25](#_Toc96087974)

## Preface

This document includes content from the “Introduction to the Tidyverse” course through Datacamp. I took this course in 2019 as part of my Master of Science in Business Analytics program with the University of North Carolina Wilmington.

This course is/was included as part of the Data Analyst with R Career Track on the Datacamp platform. This is a very helpful intro to the Tidyverse package (although, you may notice that “tidyverse” is never loaded in its entirety here).

## Data Wrangling

Here we load the necessary packages. If R presents an error, install.packages() may be needed to first, well…install the packages.

# Load the gapminder package  
#install.packages("gapminder")  
library(gapminder)  
# Load the dplyr package  
library(dplyr)  
# Loan the ggplot2 package  
library(ggplot2)  
  
# Look at the gapminder dataset  
gapminder

## # A tibble: 1,704 x 6  
## country continent year lifeExp pop gdpPercap  
## <fct> <fct> <int> <dbl> <int> <dbl>  
## 1 Afghanistan Asia 1952 28.8 8425333 779.  
## 2 Afghanistan Asia 1957 30.3 9240934 821.  
## 3 Afghanistan Asia 1962 32.0 10267083 853.  
## 4 Afghanistan Asia 1967 34.0 11537966 836.  
## 5 Afghanistan Asia 1972 36.1 13079460 740.  
## 6 Afghanistan Asia 1977 38.4 14880372 786.  
## 7 Afghanistan Asia 1982 39.9 12881816 978.  
## 8 Afghanistan Asia 1987 40.8 13867957 852.  
## 9 Afghanistan Asia 1992 41.7 16317921 649.  
## 10 Afghanistan Asia 1997 41.8 22227415 635.  
## # ... with 1,694 more rows

### Filter Verb

Filter is helpful for subsetting dataframes for conditions, to presumably perform additional operations or analysis.

# Filter for China in 2002  
gapminder %>%  
 filter(year==2002, country == "China")

## # A tibble: 1 x 6  
## country continent year lifeExp pop gdpPercap  
## <fct> <fct> <int> <dbl> <int> <dbl>  
## 1 China Asia 2002 72.0 1280400000 3119.

### Arrange Verb

Arrange is used primarily for sorting. I think of the ORDER BY clause in SQL. Arrange sorts in ascending order by default, so the reverse of that would need to be wrapped in desc().

# Sort in ascending order of lifeExp  
gapminder %>%  
 arrange(lifeExp)

## # A tibble: 1,704 x 6  
## country continent year lifeExp pop gdpPercap  
## <fct> <fct> <int> <dbl> <int> <dbl>  
## 1 Rwanda Africa 1992 23.6 7290203 737.  
## 2 Afghanistan Asia 1952 28.8 8425333 779.  
## 3 Gambia Africa 1952 30 284320 485.  
## 4 Angola Africa 1952 30.0 4232095 3521.  
## 5 Sierra Leone Africa 1952 30.3 2143249 880.  
## 6 Afghanistan Asia 1957 30.3 9240934 821.  
## 7 Cambodia Asia 1977 31.2 6978607 525.  
## 8 Mozambique Africa 1952 31.3 6446316 469.  
## 9 Sierra Leone Africa 1957 31.6 2295678 1004.  
## 10 Burkina Faso Africa 1952 32.0 4469979 543.  
## # ... with 1,694 more rows

# Sort in descending order of lifeExp  
gapminder %>%  
 arrange(desc(lifeExp))

## # A tibble: 1,704 x 6  
## country continent year lifeExp pop gdpPercap  
## <fct> <fct> <int> <dbl> <int> <dbl>  
## 1 Japan Asia 2007 82.6 127467972 31656.  
## 2 Hong Kong, China Asia 2007 82.2 6980412 39725.  
## 3 Japan Asia 2002 82 127065841 28605.  
## 4 Iceland Europe 2007 81.8 301931 36181.  
## 5 Switzerland Europe 2007 81.7 7554661 37506.  
## 6 Hong Kong, China Asia 2002 81.5 6762476 30209.  
## 7 Australia Oceania 2007 81.2 20434176 34435.  
## 8 Spain Europe 2007 80.9 40448191 28821.  
## 9 Sweden Europe 2007 80.9 9031088 33860.  
## 10 Israel Asia 2007 80.7 6426679 25523.  
## # ... with 1,694 more rows

# Filter for the year 1957, then arrange in descending order of population  
gapminder %>%  
 filter(year == 1957) %>%  
 arrange(desc(pop))

## # A tibble: 142 x 6  
## country continent year lifeExp pop gdpPercap  
## <fct> <fct> <int> <dbl> <int> <dbl>  
## 1 China Asia 1957 50.5 637408000 576.  
## 2 India Asia 1957 40.2 409000000 590.  
## 3 United States Americas 1957 69.5 171984000 14847.  
## 4 Japan Asia 1957 65.5 91563009 4318.  
## 5 Indonesia Asia 1957 39.9 90124000 859.  
## 6 Germany Europe 1957 69.1 71019069 10188.  
## 7 Brazil Americas 1957 53.3 65551171 2487.  
## 8 United Kingdom Europe 1957 70.4 51430000 11283.  
## 9 Bangladesh Asia 1957 39.3 51365468 662.  
## 10 Italy Europe 1957 67.8 49182000 6249.  
## # ... with 132 more rows

### Mutate Verb

Mutate is used to transform variables and creating new columns in the dataset.

# Use mutate to change lifeExp to be in months  
gapminder %>%  
 mutate(lifeExp = 12 \* lifeExp)

## # A tibble: 1,704 x 6  
## country continent year lifeExp pop gdpPercap  
## <fct> <fct> <int> <dbl> <int> <dbl>  
## 1 Afghanistan Asia 1952 346. 8425333 779.  
## 2 Afghanistan Asia 1957 364. 9240934 821.  
## 3 Afghanistan Asia 1962 384. 10267083 853.  
## 4 Afghanistan Asia 1967 408. 11537966 836.  
## 5 Afghanistan Asia 1972 433. 13079460 740.  
## 6 Afghanistan Asia 1977 461. 14880372 786.  
## 7 Afghanistan Asia 1982 478. 12881816 978.  
## 8 Afghanistan Asia 1987 490. 13867957 852.  
## 9 Afghanistan Asia 1992 500. 16317921 649.  
## 10 Afghanistan Asia 1997 501. 22227415 635.  
## # ... with 1,694 more rows

# Use mutate to create a new column called lifeExpMonths  
gapminder %>%  
 mutate(lifeExpMonths = 12 \* lifeExp)

## # A tibble: 1,704 x 7  
## country continent year lifeExp pop gdpPercap lifeExpMonths  
## <fct> <fct> <int> <dbl> <int> <dbl> <dbl>  
## 1 Afghanistan Asia 1952 28.8 8425333 779. 346.  
## 2 Afghanistan Asia 1957 30.3 9240934 821. 364.  
## 3 Afghanistan Asia 1962 32.0 10267083 853. 384.  
## 4 Afghanistan Asia 1967 34.0 11537966 836. 408.  
## 5 Afghanistan Asia 1972 36.1 13079460 740. 433.  
## 6 Afghanistan Asia 1977 38.4 14880372 786. 461.  
## 7 Afghanistan Asia 1982 39.9 12881816 978. 478.  
## 8 Afghanistan Asia 1987 40.8 13867957 852. 490.  
## 9 Afghanistan Asia 1992 41.7 16317921 649. 500.  
## 10 Afghanistan Asia 1997 41.8 22227415 635. 501.  
## # ... with 1,694 more rows

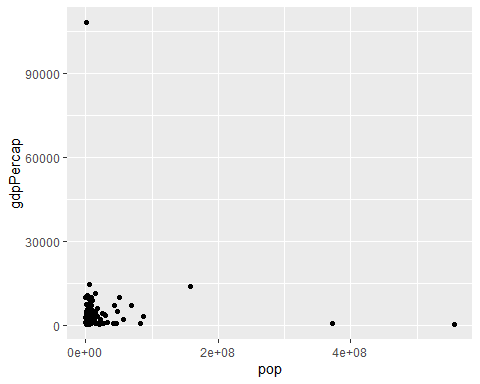
# Filter, mutate, and arrange the gapminder dataset  
gapminder %>%  
 filter(year==2007) %>%  
 mutate(lifeExpMonths = 12 \* lifeExp) %>%  
 arrange(desc(lifeExpMonths))

## # A tibble: 142 x 7  
## country continent year lifeExp pop gdpPercap lifeExpMonths  
## <fct> <fct> <int> <dbl> <int> <dbl> <dbl>  
## 1 Japan Asia 2007 82.6 127467972 31656. 991.  
## 2 Hong Kong, China Asia 2007 82.2 6980412 39725. 986.  
## 3 Iceland Europe 2007 81.8 301931 36181. 981.  
## 4 Switzerland Europe 2007 81.7 7554661 37506. 980.  
## 5 Australia Oceania 2007 81.2 20434176 34435. 975.  
## 6 Spain Europe 2007 80.9 40448191 28821. 971.  
## 7 Sweden Europe 2007 80.9 9031088 33860. 971.  
## 8 Israel Asia 2007 80.7 6426679 25523. 969.  
## 9 France Europe 2007 80.7 61083916 30470. 968.  
## 10 Canada Americas 2007 80.7 33390141 36319. 968.  
## # ... with 132 more rows

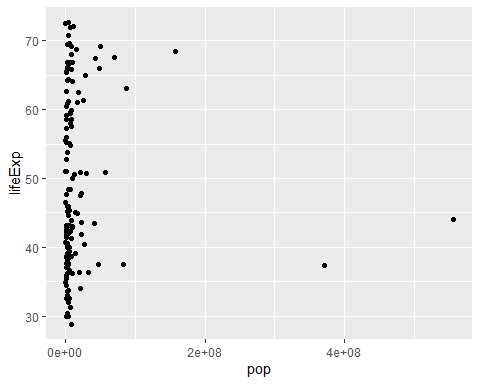
## Data Visualization

### Scatterplots

# Create gapminder\_1952  
gapminder\_1952 <- gapminder %>%  
filter (year==1952)  
  
# Put pop on the x-axis and gdpPercap on the y-axis  
ggplot(gapminder\_1952, aes(x = pop, y = gdpPercap)) +  
 geom\_point()

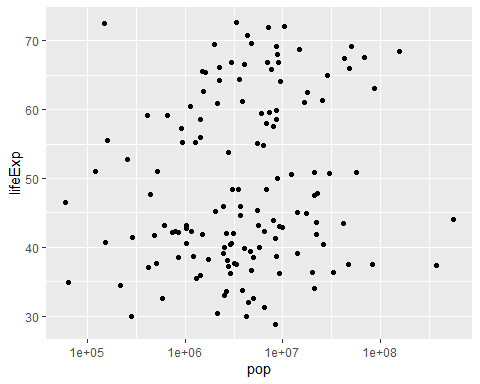


# Create a scatter plot with pop on the x-axis and lifeExp on the y-axis  
ggplot (gapminder\_1952, aes(x=pop, y=lifeExp)) +  
geom\_point()

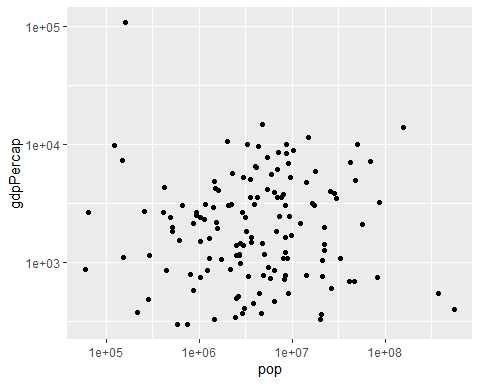


### Log Scales

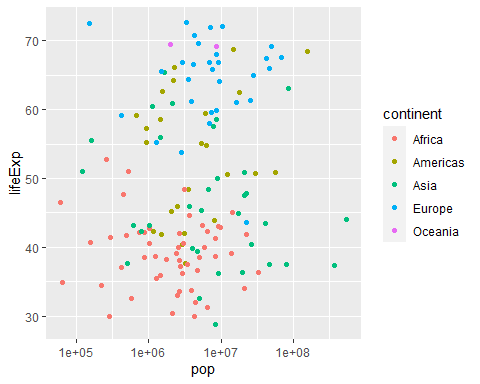
# Change this plot to put the x-axis on a log scale  
ggplot(gapminder\_1952, aes(x = pop, y = lifeExp)) +  
 geom\_point() +  
 scale\_x\_log10()



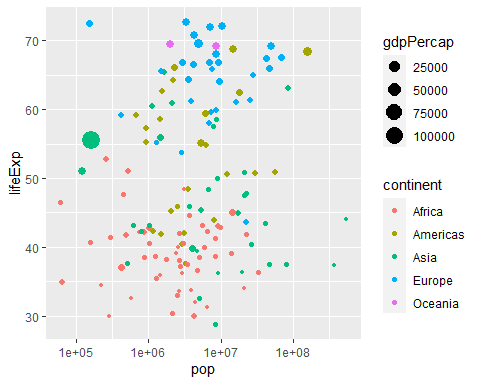
# Scatter plot comparing pop and gdpPercap, with both axes on a log scale  
ggplot(gapminder\_1952, aes(x=pop, y=gdpPercap))+  
geom\_point()+  
scale\_x\_log10()+  
scale\_y\_log10()



# Scatter plot comparing pop and lifeExp, with color representing continent  
ggplot(gapminder\_1952, aes(x=pop, y=lifeExp, color=continent)) +  
 geom\_point()+  
 scale\_x\_log10()



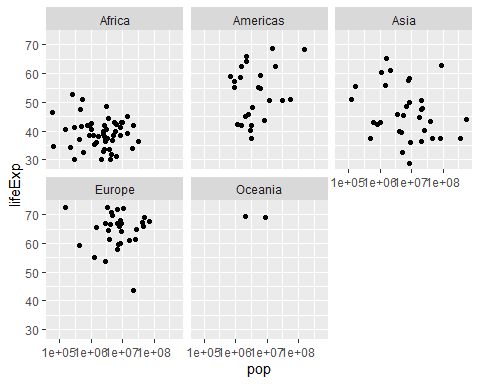
# Add the size aesthetic to represent a country's gdpPercap  
ggplot(gapminder\_1952, aes(x = pop, y = lifeExp, color = continent, size=gdpPercap)) +  
 geom\_point() +  
 scale\_x\_log10()



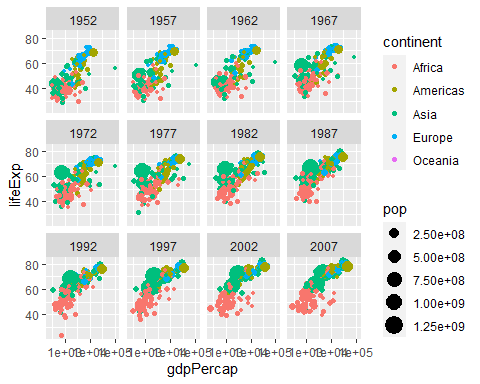
### Faceting

Faceting is normally used to create small multiple visualizations.

# Scatter plot comparing pop and lifeExp, faceted by continent  
ggplot(gapminder\_1952, aes(x=pop, y=lifeExp)) +  
 geom\_point() +  
 scale\_x\_log10() +  
 facet\_wrap(~ continent)



# Scatter plot comparing gdpPercap and lifeExp, with color representing continent  
# and size representing population, faceted by year  
ggplot(gapminder, aes(x=gdpPercap, y=lifeExp, color=continent, size=pop))+  
geom\_point()+  
scale\_x\_log10()+  
facet\_wrap(~year)



## Grouping and Summarizing

### Summarize Verb

The Summarize verb allows you to perform certain window function calculations on subsections of the dataset.

# Summarize to find the median life expectancy  
gapminder %>%  
 summarize(medianLifeExp = median(lifeExp))

## # A tibble: 1 x 1  
## medianLifeExp  
## <dbl>  
## 1 60.7

# Filter for 1957 then summarize the median life expectancy  
gapminder %>%  
 filter(year == 1957) %>%  
 summarize(medianLifeExp = median(lifeExp))

## # A tibble: 1 x 1  
## medianLifeExp  
## <dbl>  
## 1 48.4

# Filter for 1957 then summarize the median life expectancy and the maximum GDP per capita  
gapminder %>%  
 filter(year==1957) %>%  
 summarize(medianLifeExp = median(lifeExp), maxGdpPercap = max(gdpPercap))

## # A tibble: 1 x 2  
## medianLifeExp maxGdpPercap  
## <dbl> <dbl>  
## 1 48.4 113523.

### Group\_by Verb

Here we are aggregating groups of the dataset, hoping to perform calculations on those subsets.

# Find median life expectancy and maximum GDP per capita in each year  
gapminder %>%  
 group\_by(year) %>%  
 summarize(medianLifeExp = median(lifeExp), maxGdpPercap = max(gdpPercap))

## # A tibble: 12 x 3  
## year medianLifeExp maxGdpPercap  
## <int> <dbl> <dbl>  
## 1 1952 45.1 108382.  
## 2 1957 48.4 113523.  
## 3 1962 50.9 95458.  
## 4 1967 53.8 80895.  
## 5 1972 56.5 109348.  
## 6 1977 59.7 59265.  
## 7 1982 62.4 33693.  
## 8 1987 65.8 31541.  
## 9 1992 67.7 34933.  
## 10 1997 69.4 41283.  
## 11 2002 70.8 44684.  
## 12 2007 71.9 49357.

# Find median life expectancy and maximum GDP per capita in each continent in 1957  
gapminder %>%  
filter(year==1957) %>%  
group\_by(continent) %>%  
summarize(medianLifeExp = median(lifeExp), maxGdpPercap = max(gdpPercap))

## # A tibble: 5 x 3  
## continent medianLifeExp maxGdpPercap  
## <fct> <dbl> <dbl>  
## 1 Africa 40.6 5487.  
## 2 Americas 56.1 14847.  
## 3 Asia 48.3 113523.  
## 4 Europe 67.6 17909.  
## 5 Oceania 70.3 12247.

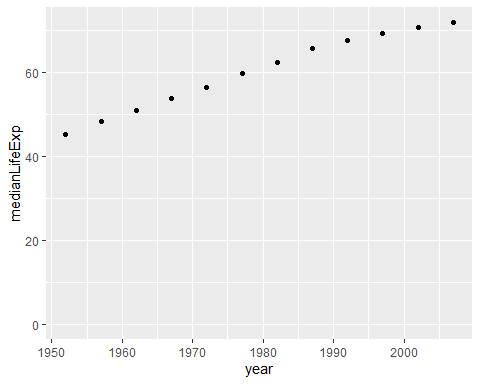
# Find median life expectancy and maximum GDP per capita in each continent/year combination  
gapminder %>%  
 group\_by(continent, year) %>%  
 summarize(medianLifeExp = median(lifeExp), maxGdpPercap = max(gdpPercap))

## # A tibble: 60 x 4  
## # Groups: continent [5]  
## continent year medianLifeExp maxGdpPercap  
## <fct> <int> <dbl> <dbl>  
## 1 Africa 1952 38.8 4725.  
## 2 Africa 1957 40.6 5487.  
## 3 Africa 1962 42.6 6757.  
## 4 Africa 1967 44.7 18773.  
## 5 Africa 1972 47.0 21011.  
## 6 Africa 1977 49.3 21951.  
## 7 Africa 1982 50.8 17364.  
## 8 Africa 1987 51.6 11864.  
## 9 Africa 1992 52.4 13522.  
## 10 Africa 1997 52.8 14723.  
## # ... with 50 more rows

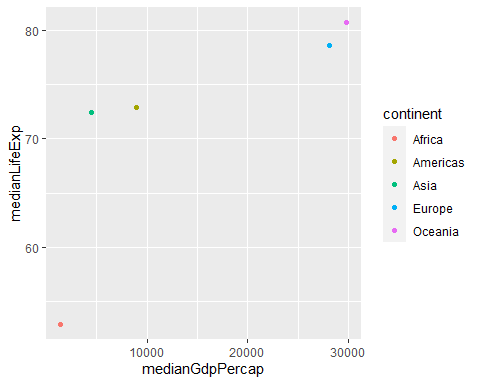
## Visualizing Summarized Data

### Scatterplots

by\_year <- gapminder %>%  
 group\_by(year) %>%  
 summarize(medianLifeExp = median(lifeExp),  
 maxGdpPercap = max(gdpPercap))  
  
# Create a scatter plot showing the change in medianLifeExp over time  
ggplot(by\_year, aes(x=year, y=medianLifeExp))+  
 geom\_point() +  
 expand\_limits(y=0)



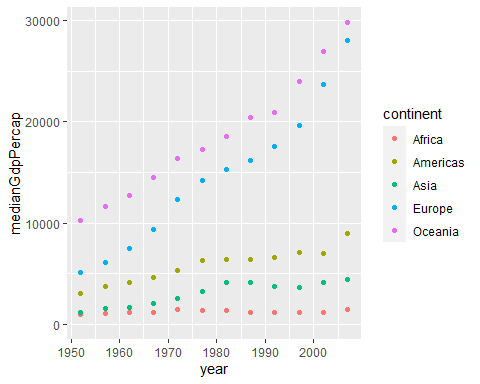
# Summarize the median GDP and median life expectancy per continent in 2007  
by\_continent\_2007 <- gapminder %>%  
 filter(year==2007) %>%  
 group\_by(continent) %>%  
 summarize(medianGdpPercap = median(gdpPercap), medianLifeExp = median(lifeExp))  
  
# Use a scatter plot to compare the median GDP and median life expectancy  
ggplot(by\_continent\_2007, aes(x=medianGdpPercap, y=medianLifeExp, color=continent)) +  
 geom\_point()



expand\_limits(y = 0)

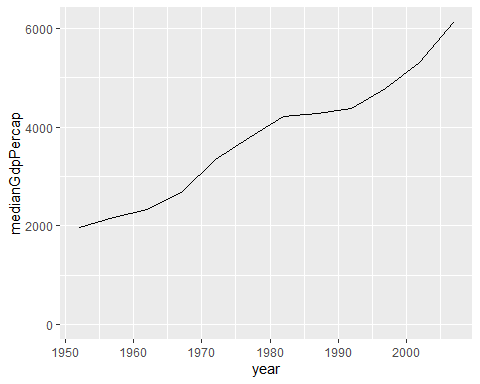
## mapping: y = ~y   
## geom\_blank: na.rm = FALSE  
## stat\_identity: na.rm = FALSE  
## position\_identity

# Summarize medianGdpPercap within each continent within each year: by\_year\_continent  
by\_year\_continent <- gapminder %>%  
 group\_by(continent, year) %>%  
 summarize(medianGdpPercap = median(gdpPercap))  
  
# Plot the change in medianGdpPercap in each continent over time  
ggplot(by\_year\_continent, aes(x=year, y=medianGdpPercap, color=continent))+  
 geom\_point() +  
 expand\_limits(y = 0)

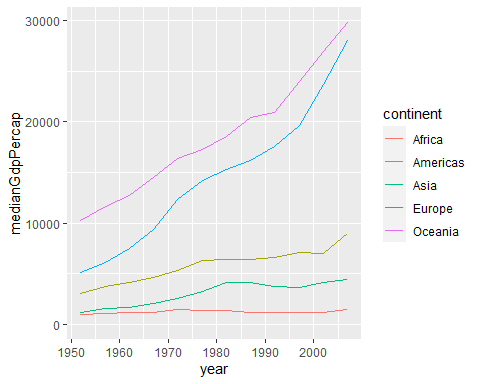


### Line Plots

# Summarize the median gdpPercap by year, then save it as by\_year  
by\_year <- gapminder %>%  
 group\_by(year) %>%  
 summarize(medianGdpPercap = median(gdpPercap))  
  
# Create a line plot showing the change in medianGdpPercap over time  
ggplot(by\_year, aes(x=year, y=medianGdpPercap))+  
 geom\_line()+  
 expand\_limits(y=0)

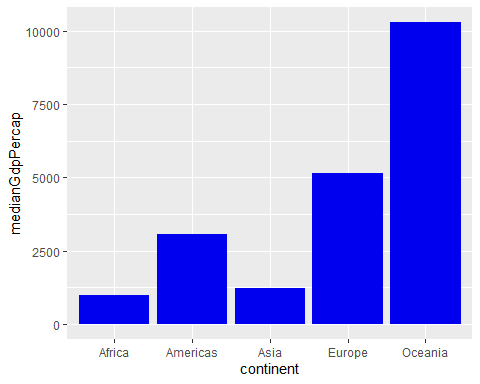


# Summarize the median gdpPercap by year & continent, save as by\_year\_continent  
by\_year\_continent <- gapminder %>%  
 group\_by(year, continent) %>%  
 summarize(medianGdpPercap=median(gdpPercap))  
  
# Create a line plot showing the change in medianGdpPercap by continent over time  
ggplot(by\_year\_continent, aes(x=year, y=medianGdpPercap, color=continent))+  
 geom\_line() +  
 expand\_limits(y=0)

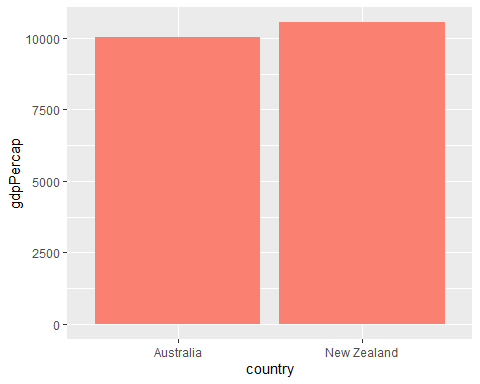


### Bar Plots

# Summarize the median gdpPercap by continent in 1952  
by\_continent <- gapminder %>%  
 filter(year == 1952) %>%  
 group\_by(continent) %>%  
 summarize(medianGdpPercap = median(gdpPercap))  
  
# Create a bar plot showing medianGdp by continent  
ggplot(by\_continent, aes(x=continent, y=medianGdpPercap))+  
 geom\_col(fill="blue2") #Playing with the colors here

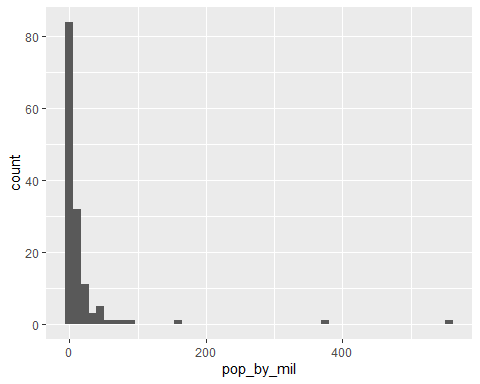


# Filter for observations in the Oceania continent in 1952  
oceania\_1952 <- gapminder %>%  
 filter(year==1952) %>%  
 filter(continent=="Oceania")  
  
# Create a bar plot of gdpPercap by country  
ggplot(oceania\_1952, aes(x=country, y=gdpPercap))+  
 geom\_col(fill="salmon") #Playing with the colors here

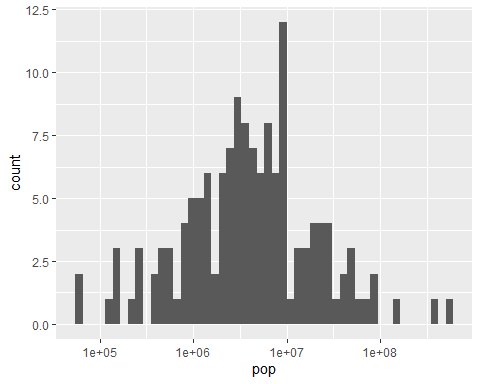


### Histograms

gapminder\_1952 <- gapminder %>%  
 filter(year == 1952) %>%  
 mutate(pop\_by\_mil = pop / 1000000)  
  
# Create a histogram of population (pop\_by\_mil)  
ggplot(gapminder\_1952, aes(x=pop\_by\_mil))+  
 geom\_histogram(bins=50)

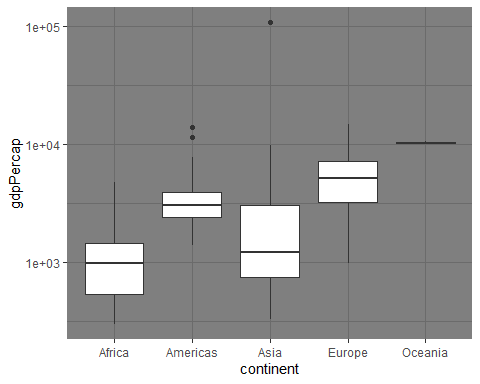


# Create a histogram of population (pop), with x on a log scale  
ggplot(gapminder\_1952, aes(x=pop))+  
 geom\_histogram(bins=50)+  
 scale\_x\_log10()

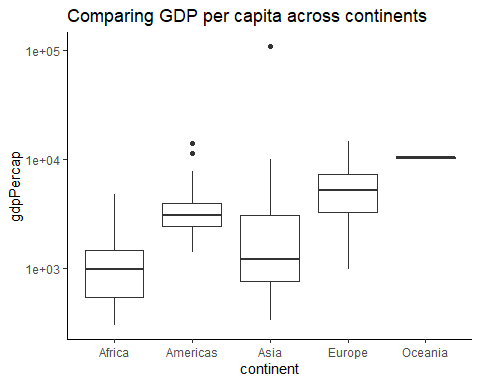


### Boxplots

gapminder\_1952 <- gapminder %>%  
 filter(year == 1952)  
  
# Create a boxplot comparing gdpPercap among continents  
ggplot(gapminder\_1952, aes(x=continent, y=gdpPercap))+  
 geom\_boxplot()+  
 scale\_y\_log10()+  
 theme\_dark() #Adds a dark theme



# Add a title to this graph: "Comparing GDP per capita across continents"  
ggplot(gapminder\_1952, aes(x = continent, y = gdpPercap)) +  
 geom\_boxplot() +  
 scale\_y\_log10() +  
 theme\_classic() + #Adds a classic theme  
 ggtitle("Comparing GDP per capita across continents")



## Summary

After completing this overview, students should feel more confident in applying the basic functions of dplyr, ggplot2 (both of which are included in the tidyverse collection of packages). There is plenty more to learn on your journey to becoming a data analyst, but this intro is helpful. Filter, Arrange, Mutate, Group By, and Summarize are a great start

The exploration of multiple visualizations in this course also give students the experience to see how each one operates. There are use cases for each, and determining which visualization to use will depend on your own analysis.