* [Question 1](https://elearn.waikato.ac.nz/pluginfile.php/3023398/mod_resource/content/1/Assn1-Ans.html" \l "question-1)
* [Question 2](https://elearn.waikato.ac.nz/pluginfile.php/3023398/mod_resource/content/1/Assn1-Ans.html#question-2)
* [Question 3](https://elearn.waikato.ac.nz/pluginfile.php/3023398/mod_resource/content/1/Assn1-Ans.html#question-3)
* [Question 4](https://elearn.waikato.ac.nz/pluginfile.php/3023398/mod_resource/content/1/Assn1-Ans.html#question-4)
* [Question 5](https://elearn.waikato.ac.nz/pluginfile.php/3023398/mod_resource/content/1/Assn1-Ans.html#question-5)
* [Question 6](https://elearn.waikato.ac.nz/pluginfile.php/3023398/mod_resource/content/1/Assn1-Ans.html#question-6)
* [Question 7](https://elearn.waikato.ac.nz/pluginfile.php/3023398/mod_resource/content/1/Assn1-Ans.html#question-7)

**Assignment 1**

DATAX201

**Question 1**

It initially involved using data analysis to make informed insights about what was studied. Nowadays, data science begins with manipulating (wrangling) raw data to prepare it for analysis or visualisation. Then, either analysis, visualisation (or dashboarding), or a combination of both follows. Typically, the key results are presented to clients and potentially deployed as products for clients. Lastly, this has been an emerging field of data science ethics, especially with the increased volume of data collected actively or passively.

**Question 2**

It is a style of data entry for a rectangular dataset where we enforce the following rules:

* Each observation (case) is a row.
* Each variable is a column.
* The values measured about observation for each variable is a cell.

**Question 3**

* We use read.csv() to load in a comma-separated value file into **R**.
* We use read.spss() from the foreign package to load in a SPSS data file into **R**.
* We use read\_excel() to load in a Microsoft Spreadsheet into **R**.
* We use read\_json() from the jsonlite package to load in a JavaScript Object Notation file into **R**.

**Question 4**

Calculate 1+2+3+5+8+13+21.

# R as a calculator approach

1 + 2 + 3 + 5 + 8 + 13 + 21

[1] 53

# R as a programming language approach (Week 6)

x <- c(1, 2, 3, 5, 8, 13, 21)

sum(x)

[1] 53

Calculate log2⁡(0.5).

# Using the log() function

log(0.5, base = 2)

[1] -1

# Using the log2() function

log2(0.5)

[1] -1

Calculate 335−3.

3^3/(5 - 3)

[1] 13.5

**Question 5**

The function presented in Question 5 is known as the logistic function adapted for logistic regression, a data analysis technique taught in *DATAX221 - Statistical Data Analysis*.

**Part (a)**

# Assign the values as requested

beta0 <- 5.14

beta1 <- -0.04

x <- 40

**Part (b)**

# Evaluate the logistic regression function

exp(beta0 + beta1 \* x)/(1 + exp(beta0 + beta1 \* x))

[1] 0.9718047

**Question 6**

**Part (a)**

# Load in the data

shell.df <- read.csv("data/MusselEnd.csv")

**Part (b)**

# Load in the dplyr package

library(dplyr)

# Create a new categorical variable using mutate()

shell.df <- shell.df %>%

mutate(PlotType = if\_else(Benthic == "Mud", "Soft sediment", "Shell substrate"))

**Part (c)**

# We want to group the data by the distinct values of Location and PlotType,

# \_before\_ producing the requested descriptive statistics for the mussel

# lengths

shell.df %>%

group\_by(Location, PlotType) %>%

summarise(xbar = mean(Length), s.sd = sd(Length))

# A tibble: 4 × 4

# Groups: Location [2]

Location PlotType xbar s.sd

<chr> <chr> <dbl> <dbl>

1 "Fairy Bay " Shell substrate 112. 8.85

2 "Fairy Bay " Soft sediment 113. 9.18

3 "Kenepuru Entrance" Shell substrate 111. 7.11

4 "Kenepuru Entrance" Soft sediment 109. 10.3

Alternative answer:

# Tidyverse is experimenting with embedding the group\_by() functionality within

# summarise() and other dplyr functions

shell.df %>%

summarise(xbar = mean(Length), s.sd = sd(Length), .by = c("Location", "PlotType"))

**Part (d)**

No, it does not seem plausible that the plot location, plot type, or both features influenced mussels’ average shell lengths by the study’s end. All sample means are quite similar. Furthermore, the distribution of the sampled mussel lengths for each combination of plot location and plot type will likely overlap once we factor in both sample mean and standard deviations. (Each group’s distribution is approximately centred at the same point, and the spread of the lengths within a group is relatively large).

**Question 7**

**Part (a)**

# Load in the readxl package

library(readxl)

# Load in the data

king.df <- read\_excel("data/KingCounty.xlsx")

**Part (b)**

# Load in the tidyr package

library(tidyr)

# Let's survey what the first few rows of the data looks like

head(king.df)

# A tibble: 6 × 13

bedroom\_0 bedroom\_1 bedroom\_2 bedroom\_3 bedroom\_4 bedroom\_5 bedroom\_6

<dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>

1 1095000 350000 180000 221900 604000 530000 464000

2 380000 369900 468000 538000 1225000 285000 315000

3 288000 299000 189000 510000 650000 861990 472000

4 228000 80000 252700 257500 485000 345000 279000

5 1295650 157000 280000 291850 385000 920000 1067000

6 339950 245000 585000 229500 719000 951000 1600000

# ℹ 6 more variables: bedroom\_7 <dbl>, bedroom\_8 <dbl>, bedroom\_9 <dbl>,

# bedroom\_10 <dbl>, bedroom\_11 <dbl>, bedroom\_33 <dbl>

As the question sheet described, the data was not organised using tidy data principles. Each house (observation) is a unique cell, and each cell additionally measures that house’s sale price. Furthermore, the houses are organised column-wise by how many bedrooms they have: zero bedrooms, one bedroom, two bedrooms, and so on. So we want to pivot the data longer.

This is precisely the scenario we had in the introduction to tidyr lecture recording. If we checked ?tidyr\_tidy\_select, we use cols = everything() input of pivot\_longer().

# As-is

king.tidied.df <- pivot\_longer(king.df, cols = everything(), names\_to = "num\_bedrooms",

values\_to = "price")

# Print the number of rows...

nrow(king.tidied.df)

[1] 127712

That is a bit worrying, but this is due to the missing values created by pivoting it longer. Recall that the descriptive statistic functions covered in Workshop 2 can ignore missing values, so this is not that big of a problem at this stage. However, if you want to drop all missing values, this is what you would do instead:

# You could filter out the NA values at this step prior to part (c)

king.tidied.df <- pivot\_longer(king.df, cols = everything(), names\_to = "num\_bedrooms",

values\_to = "price", values\_drop\_na = TRUE)

**Part (c)**

# Loading in the dplyr package

library(dplyr)

king.tidied.df <- king.tidied.df %>%

mutate(LQ = quantile(price, probs = 0.25, na.rm = TRUE), UQ = quantile(price,

probs = 0.75, na.rm = TRUE), IQR = IQR(price, na.rm = TRUE))

**Part (d)**

# Houses whose sale price was lower than LQ - 3 \* IQR

filter(king.tidied.df, price < LQ - 3 \* IQR)

# A tibble: 0 × 5

# ℹ 5 variables: num\_bedrooms <chr>, price <dbl>, LQ <dbl>, UQ <dbl>, IQR <dbl>

# Houses whose sale price was greater than UQ + 3 \* IQR

filter(king.tidied.df, price > UQ + 3 \* IQR)

# A tibble: 420 × 5

num\_bedrooms price LQ UQ IQR

<chr> <dbl> <dbl> <dbl> <dbl>

1 bedroom\_7 1950000 321950 645000 323050

2 bedroom\_8 2150000 321950 645000 323050

3 bedroom\_8 1650000 321950 645000 323050

4 bedroom\_7 3200000 321950 645000 323050

5 bedroom\_8 1970000 321950 645000 323050

6 bedroom\_8 3300000 321950 645000 323050

7 bedroom\_3 2000000 321950 645000 323050

8 bedroom\_6 5300000 321950 645000 323050

9 bedroom\_5 2050000 321950 645000 323050

10 bedroom\_7 2280000 321950 645000 323050

# ℹ 410 more rows

# Alternatively, do both at the same time

filter(king.tidied.df, price < LQ - 3 \* IQR | price > UQ + 3 \* IQR)

# A tibble: 420 × 5

num\_bedrooms price LQ UQ IQR

<chr> <dbl> <dbl> <dbl> <dbl>

1 bedroom\_7 1950000 321950 645000 323050

2 bedroom\_8 2150000 321950 645000 323050

3 bedroom\_8 1650000 321950 645000 323050

4 bedroom\_7 3200000 321950 645000 323050

5 bedroom\_8 1970000 321950 645000 323050

6 bedroom\_8 3300000 321950 645000 323050

7 bedroom\_3 2000000 321950 645000 323050

8 bedroom\_6 5300000 321950 645000 323050

9 bedroom\_5 2050000 321950 645000 323050

10 bedroom\_7 2280000 321950 645000 323050

# ℹ 410 more rows