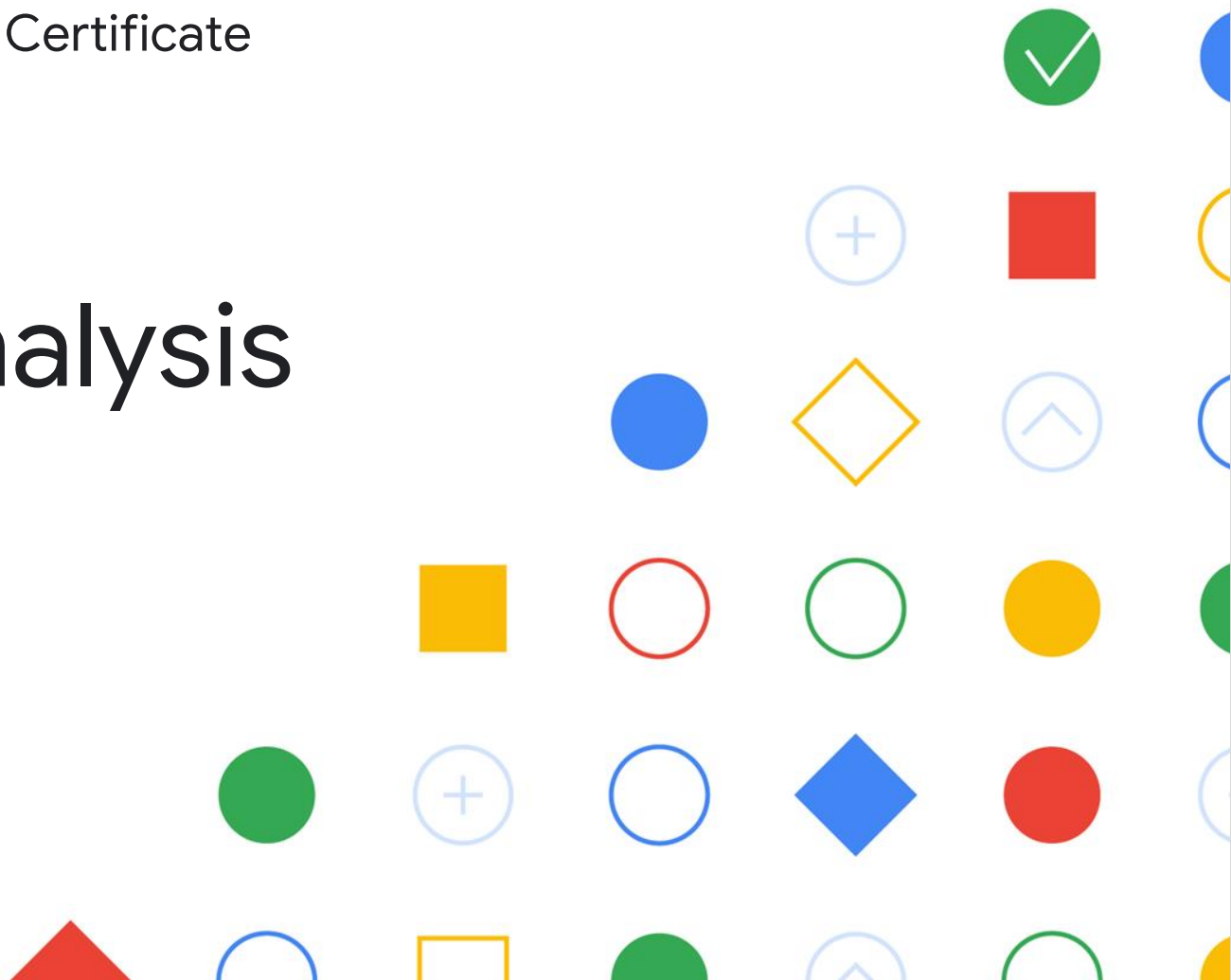
Google Data Analytics Certificate

7. Data Analysis

with R Programming



Samart Skills
A Grow with Google PROGRAM

Skooldio

Overview:

01

Programming and Data Analytics

02

Programming using RStudio

03

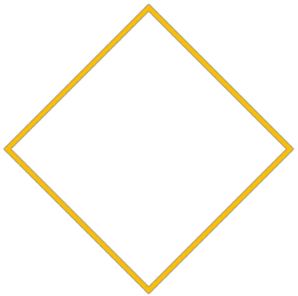
Working with Data in R

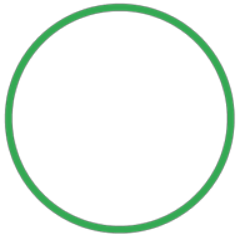
04

More about Visualizations, Aesthetics and Annotations

05

Documentation and Reports





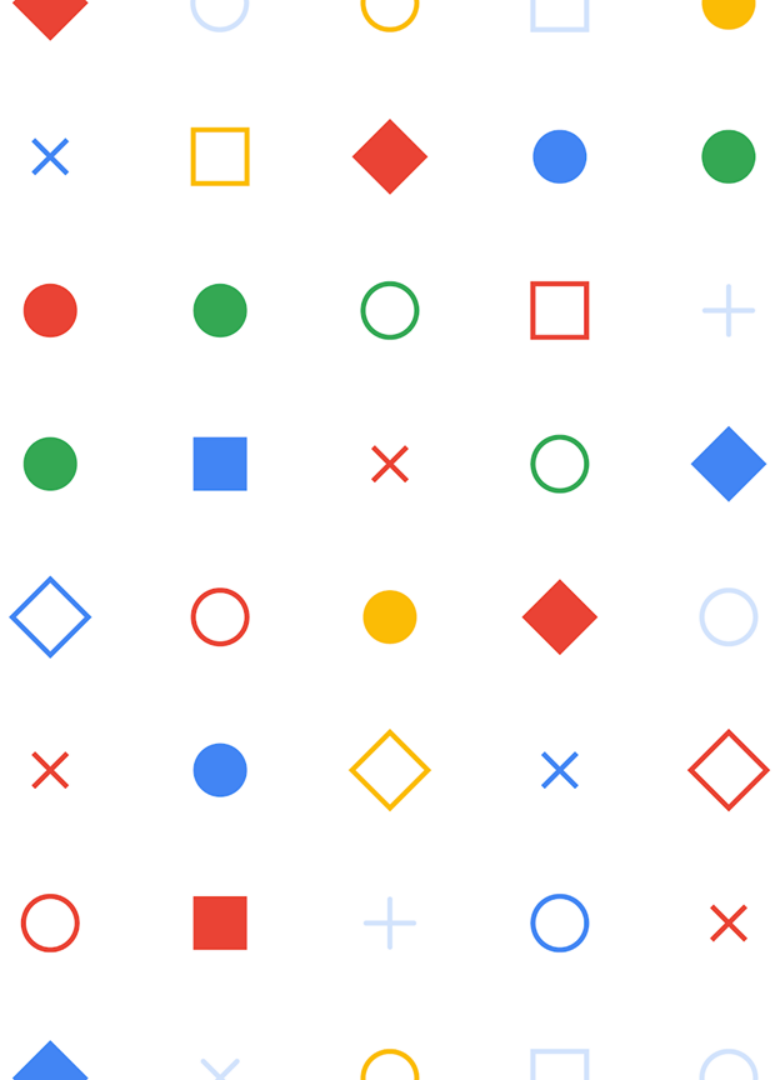
Samart Skills
A Grow with Google PROGRAM

Skooldio

Programming and Data Analytics

Part1:

1. Why Programming?
2. Why R?
3. Introducing RStudio



Samart Skills
A Grow with Google PROGRAM

Skooldio

Why Programming?

การเขียนโปรแกรมคือการเขียนคำสั่ง (Instructions) ให้คอมพิวเตอร์ทำตาม โดยคำสั่งที่เขียนต้องเป็นไปตาม “ไวยกรณ์” (Syntax) ของภาษาที่ใช้ เช่นภาษา Python กับ R มี Syntax ที่ต่างกัน

Why Programming?	Examples
ช่วยทำให้ขั้นตอนการวิเคราะห์ข้อมูลมีความชัดเจน (Clarify the steps of your analysis)	เราสามารถดูขั้นตอนการทำความสะอาดและวิเคราะห์ข้อมูลได้จาก Code ที่เราหรือคนอื่นเขียน ทำให้ตรวจสอบ Logic ได้ง่าย ต่างจาก Spreadsheets ที่เห็นแต่ผลลัพธ์สุดท้าย ไม่ได้มีสรุปขั้นตอนให้ชัดเจน
ช่วยประหยัดเวลาในการวิเคราะห์ข้อมูล (Saves time)	เราทำความสะอาดข้อมูลได้ด้วย Code ไม่กี่บรรทัด (บางทีบรรทัดเดียว) แต่การทำความสะอาดข้อมูลใน Spreadsheets อาจต้องทำหลายขั้นตอน
ทำการวิเคราะห์ซ้ำกับข้อมูลใหม่ได้ง่าย และแชร์ code หรือผลลัพธ์ได้สะดวก (Reproduce and share your work)	เราเขียนโปรแกรมให้นำเข้าข้อมูล ทำความสะอาดข้อมูล วิเคราะห์และสร้าง Report รายเดือนแบบอัตโนมัติได้ แต่ถ้าใช้ Spreadsheets เราต้องมานั่งทำแต่ละขั้นตอนด้วยตนเอง

Why R?

What is R?

- R คือภาษาคอมพิวเตอร์ (Programming language) ที่นิยมใช้สำหรับการวิเคราะห์ทางสถิติ (Statistical analysis) การสร้างกราฟและแผนภาพ (Visualizations) และการวิเคราะห์ข้อมูลทั่วไป (Data Analysis)

Why R?

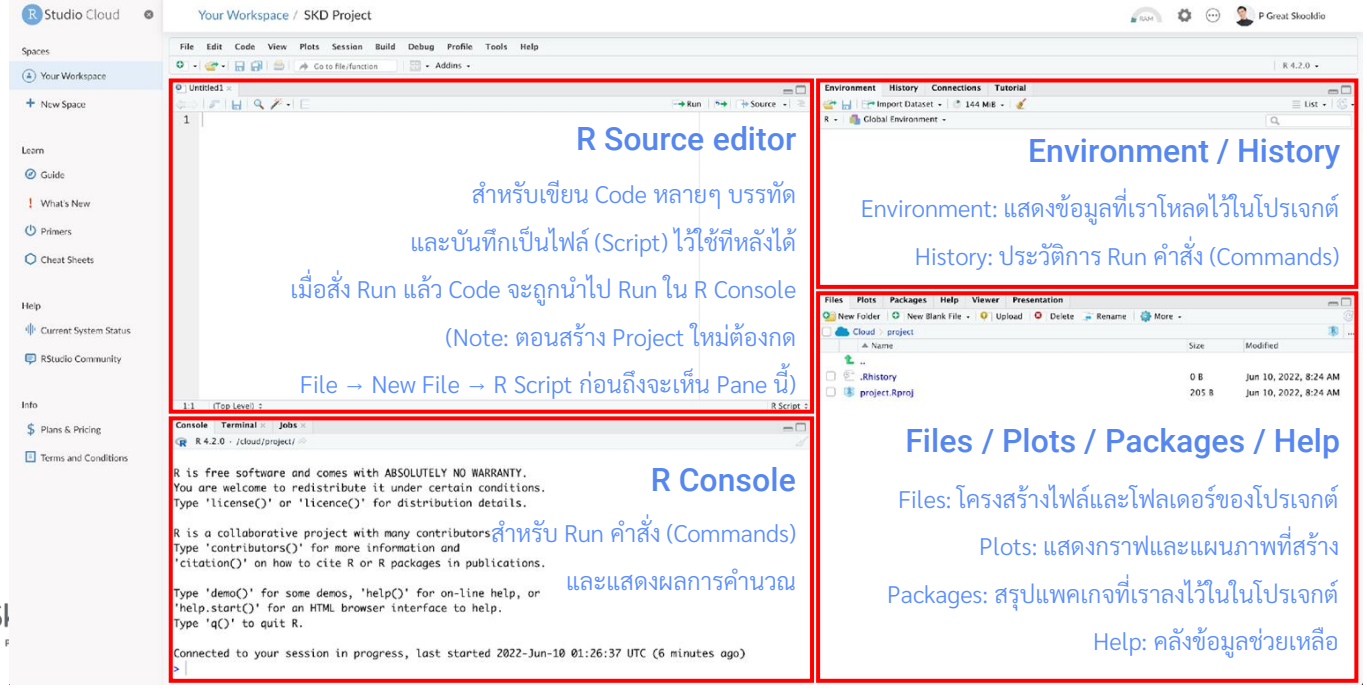
- Accessible — เข้าถึงได้ง่ายสำหรับผู้เริ่มต้นเขียนโปรแกรม
- Data-centric — ถูกออกแบบเพื่อเล่นกับข้อมูลโดยเฉพาะ
- Open source — ใคร ๆ ก็ดาวน์โหลดไปใช้ ดัดแปลง แก้ Bugs หรือสร้าง Packages เสริมใหม่ ๆ แชร์ให้คนอื่นใช้ได้ฟรี
- Community — มีสังคมออนไลน์คอยช่วยเหลือ แชร์ความรู้กัน

What can R do?

- Reproducing your analysis — ทำการวิเคราะห์ซ้ำเดิมได้อัตโนมัติแค่คลิกเดียว
- Processing lots of data — ประมวลผลข้อมูลปริมาณมาก ๆ ได้
- Creating visualizations — สร้างและปรับแต่งกราฟสวย ๆ ได้ในไม่กี่บรรทัด

Introducing RStudio

RStudio คือซอฟต์แวร์ที่รวมเอาเครื่องมือต่าง ๆ ในการช่วยเขียนโปรแกรมภาษา R มาไว้ในที่เดียว ซึ่งซอฟต์แวร์ลักษณะนี้เรียกว่า **Integrated Development Environment (IDE)** มีหลัก ๆ 4 หน้าต่าง (Panes) ดังนี้



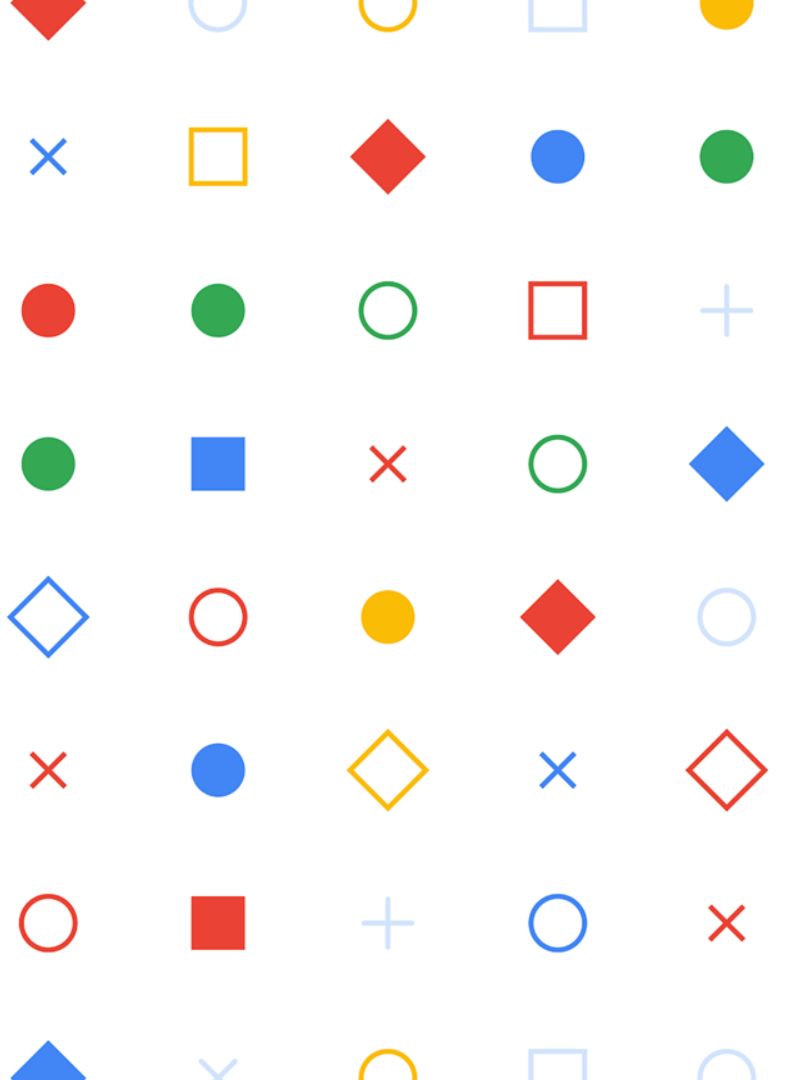
Programming with RStudio

Part2:

1. RStudio Workflow
2. Basic Data Types
3. Basic Data Structures
4. Operators and Calculations
5. Tidyverse Packages
6. Nested Commands and Pipe

Samart Skills
A Grow with Google PROGRAM

Skooldio



RStudio Workflow

1 Install and Load Packages

- สรุปแพคเกจที่ถูกลงไว้แล้วด้วยคำสั่ง `installed.packages()`
- ลงแพคเกจเพิ่มที่จะใช้ เช่น `install.packages("tidyverse")`
- โหลดแพคเกจเพื่อใช้ด้วยคำสั่ง `library(ชื่อแพคเกจ)` เช่น `library(tidyverse)`

2 Import Datasets

- นำเข้า Dataset จากไฟล์ของเราเอง (เช่น ไฟล์ .csv ด้วยฟังก์ชัน `read_csv("ที่ตั้งไฟล์")` จากแพจเกจ readr)
- หรือใช้ Dataset ตัวอย่างที่มีอยู่แล้ว (เช่น diamonds จากแพคเกจ ggplot2)

3 "Tidy" the Data

- ทำความสะอาด (Clean) และแปลง (Transform) ข้อมูลเพื่อเตรียมพร้อมสำหรับนำไปวิเคราะห์ต่อ (เช่น ด้วยฟังก์ชันต่าง ๆ ของแพจเกจ tidyr หรือ janitor)

4 Analyze and Visualize

- วิเคราะห์ข้อมูล เช่น ด้วยฟังก์ชันต่าง ๆ ของแพคเกจ dplyr
- สร้างกราฟ เช่น ด้วยฟังก์ชันของแพคเกจ ggplot2

5 Create and Share Reports

- ใช้ภาษา R Markdown ในการเขียนสรุปขั้นตอนและผลการวิเคราะห์ข้อมูล และบันทึกเป็นไฟล์ .Rmd, .pdf หรือ .html เพื่อส่งให้คนอื่นดูได้

Basic Data Types

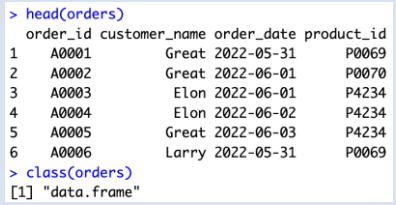
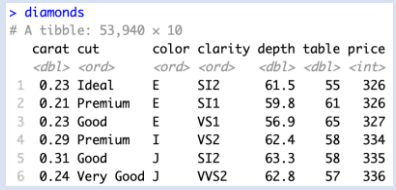
ประเภทของข้อมูลใน R ที่ควรรู้จักเบื้องต้นมีดังนี้

	Examples	Remarks
Logical ข้อมูล ถูก/ผิด	`TRUE, FALSE`	ต้องตัวพิมพ์ใหญ่ล้วน ห้ามใช้ True, true, False, false
Integer จำนวนเต็ม	`0, 1, 2, 3, 555, -123`	เราเรียกข้อมูลที่เป็นตัวเลข (Integer กับ Double) โดยรวมว่า Numeric
Double จำนวนทศนิยม (Decimal)	`31.44, 10.0, -999.11`	
Character ตัวอักษรหรือข้อความ (String)	`"a", 'Great', "Jennie is cute!", "TRUE"`	จะใช้เครื่องหมายคำพูดแบบ " " (Double quote) หรือ ' ' (single quote) ก็ได้

Datetime*
วันเวลา
	`"1991-01-07"`
	`"07:01:25 UTC"`
	`"2022-05-02 00:15:30 +07"`
		*ต้องลงและโหลดแพคเกจที่ชื่อว่า lubridate ก่อน

Basic Data Structures

โครงสร้างของข้อมูลใน R ที่ควรรู้จักเบื้องต้นมีดังนี้

	Examples	Remarks
(Atomic) Vector เวกเตอร์	`c(1, 2, 3, -20)` `c("Jennie", "is", "very cute!")`	ข้อมูลใน Vector ต้องเป็นประเภทเดียวกันนั่นคือเป็น Numeric ล้วน, Character ล้วน หรือ Logical ล้วน
List ลิสต์	`list("Great", 555, TRUE)`	ข้อมูลใน List ไม่จำเป็นต้องเป็นประเภทเดียวกันก็ได้ และสามารถสร้าง List ของ List (Nested Lists) ได้ด้วย
Data frame ตารางข้อมูลแบบมาตรฐาน		ข้อมูลตารางที่ Import เข้ามาด้วยฟังก์ชัน `read.csv("ที่ตั้งไฟล์")` จะเป็น Data frame แต่ถ้าใช้ฟังก์ชัน `read_csv("ที่ตั้งไฟล์")` จากแพคเกจ readr จะได้มาเป็น Tibble
Tibble ตารางข้อมูลแบบพิเศษของแพคเกจ tidyverse		Tibble คือ Data frame ที่มีการเพิ่มพฤติกรรมบางอย่างให้ใช้ง่ายได้ง่ายขึ้นกับตารางที่มีขนาดใหญ่ เช่น เลือกบางคอลัมน์ (Subsetting) ได้ง่ายขึ้น แสดงผลสวยขึ้น

Operators and Calculations

Assignment operator <-

"ยัดค่า" ใส่ตัวแปรด้วยเครื่องหมาย <-

Syntax:

ชื่อตัวแปร <- อะไรก็ได้

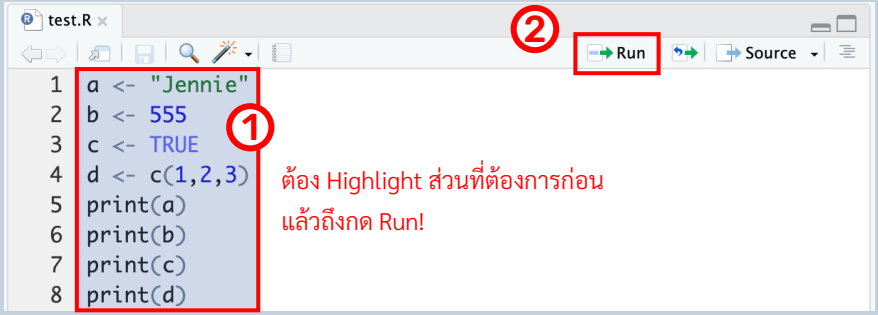
Notes :

- ชื่อตัวแปรต้องไม่ขึ้นต้นด้วยตัวเลขหรือสัญลักษณ์แปลก ๆ เช่น `lucky99` โอเค แต่ `99lucky` ไม่โอเค
- ตัวพิมพ์เล็กใหญ่มีผล เช่น ถ้า a <- 5 และ b <- 10 หากเราคำนวณ A+B จะ error เพราะเรายังไม่ได้ยัดค่าใส่ตัวแปรชื่อ A หรือ B เลย
- เราใช้เครื่องหมาย = แทน <- ก็ได้ แต่ในสังคมคนใช้ R เขานิยมใช้ <- กันส่วนใหญ่

Samart Skills
A Grow with Google PROGRAM

Skooldio

Ex. ลองยัดค่าใส่ตัวแปร แล้ว print ออกมา



Result

Operators and Calculations

Arithmetic operators + - * /

เครื่องหมายสำหรับการคำนวณ บวก ลบ คูณ หาร

Syntax:

```
a + 10*b + (2/3)
```

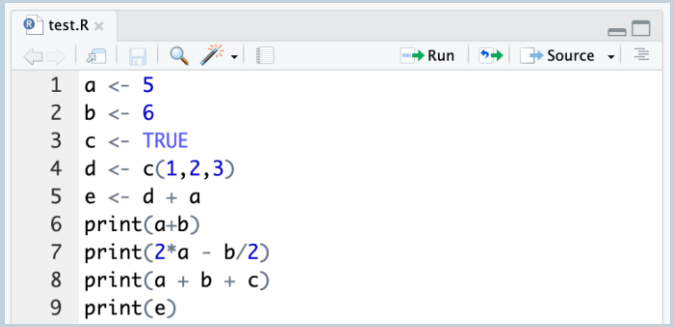
Notes :

- สำหรับภาษา R เราไม่สามารถเอาข้อมูลแบบ String มาเชื่อมกันด้วยเครื่องหมาย + ได้เหมือน Python (ต้องใช้ฟังก์ชันชื่อว่า paste() แทน เช่น paste("Jennie","Narak") ได้ "Jennie Narak")
- TRUE คือ 1 และ FALSE คือ 0 ดังนั้น ถ้าลอง TRUE + FALSE + 5 จะเทียบเท่า 1 + 0 + 5 ซึ่งเท่ากับ 6

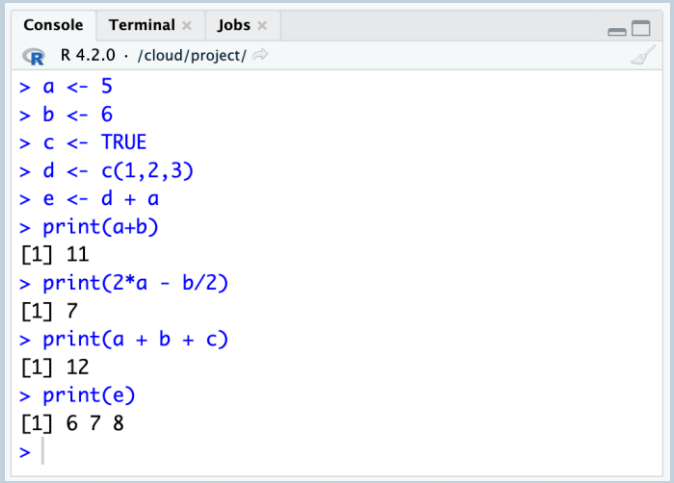
Samart Skills
A Grow with Google PROGRAM

Skooldio

Ex. ลองคำนวณแบบต่าง ๆ แล้ว print ออกมา



Result



Operators and Calculations

Logical operators
เครื่องหมายสำหรับการผสมข้อมูลถูก/ผิด

Syntax:

```
p & q       #AND (และ)
p | q       #OR (หรือ)
!p          #NOT (ตรงข้าม a.k.a. นิเสธ)
a == 2      #a มีค่าเท่ากับ 2 หรือไม่?
a <= b      #a มีค่าน้อยกว่าหรือเท่ากับ b ไหม?
```

Ex. ลองคำนวณแบบต่าง ๆ แล้ว print ออกมา

test.R

```
1  #Assign variables
2  a <- 5
3  b <- 6
4  c <- TRUE #Yo!
5  #Compute
6  print(a == b) #Is a equal to b?
7  print(a != b) #Is a NOT equal to b?
8  print((a < 2) & (b >= 6))
9  print((a < 2) | (b >= 6))
10 print(!c)
```

Result

Console | Terminal | Jobs

R 4.2.0 · /cloud/project/

```
> #Assign variables
> a <- 5
> b <- 6
> c <- TRUE #Yo!
> #Compute
> print(a == b) #Is a equal to b?
[1] FALSE
> print(a != b) #Is a NOT equal to b?
[1] TRUE
> print((a < 2) & (b >= 6))
[1] FALSE
> print((a < 2) | (b >= 6))
[1] TRUE
> print(!c)
[1] FALSE
>
```

Tidyverse Packages

R มีแพจเกจอยู่บนแพลตฟอร์มชื่อว่า CRAN (Comprehensive R Archive Network) ให้เราโหลดมาใช้ฟรี และที่นิยมสำหรับงานวิเคราะห์ข้อมูลคือกลุ่มแพคเกจชื่อ Tidyverse ซึ่งมีแพคเกจย่อยมากมาย แต่เบื้องต้นที่ควรรู้จักคือ

	Main Purposes	Frequently-used Functions
readr รี้ด-อาร์	นำเข้าข้อมูล	• `table <- read_csv(" ที่ตั้งไฟล์ ")`
tidyr ไทดี-อาร์	ทำความสะอาด (Tidy) และจัดรูปข้อมูล	• `as_tibble(df,...)` #เปลี่ยน Data frame เป็น Tibble • `unite(table, col1, col2,...)` #เอาค่า 2 คอลัมน์มา concatenate กัน • `separate(table, col, sep='/',...)` #แยกค่าตามตัวคั่นเป็นคอลัมน์ย่อย • `drop_na(table, คอลัมน์ที่สนใจ)` #เอาแถวที่มีค่าว่างในคอลัมน์ที่สนใจออก
dplyr ดี-พลาย-อาร์	กรอง ดัดแปลง คำนวณทางสถิติแบบแบ่งกลุ่ม และอื่น ๆ	• `filter(orders, price > 100)` #กรองข้อมูล • `select(table, col1, col2,...)` #เลือกมาเฉพาะบางคอลัมน์ • `table %>% group_by(col1) %>% summarise(...)` #สรุปข้อมูลแบบแบ่งกลุ่ม
ggplot2 จีจีพล็อต-ทู	สร้างและปรับแต่ง Visualizations	• `geom_bar()`, `geom_histogram()`, `geom_line()`, `geom_boxplot()`, `geom_violin()`, `geom_point()` และอื่นๆ อีกมากมาย!

Nested Commands and Pipes

Nested functions
ฟังก์ชันที่ฝังอยู่ด้านในของอีกฟังก์ชันหนึ่ง

Syntax:

```
func2(func1(data, ...), ...)
```

Notes:

- การประมวลผลจะเริ่มต้นจากฟังก์ชันด้านในสุดก่อน แล้วผลของฟังก์ชันด้านใน จะเป็น Input ให้กับฟังก์ชันด้านนอก
- จากที่เห็น func1 จัดเป็น Nested function เพราะฝังอยู่ด้านใน func2 อีกทีหนึ่ง

Samart Skills
A Grow with Google PROGRAM

Skooldio

Ex. จากตารางรายการสั่งซื้อสินค้า จงกรองเอาเฉพาะสินค้าราคา 5 บาท และแสดงผลเรียงลำดับรายได้ต่อการสั่งซื้อ จากน้อยไปมาก

```
1  orders <- read_csv('./data/orders_full.csv')
2
3  orders$revenue = orders$product_price * orders$quantity
```

```
> orders
# A tibble: 10 × 7
   order_id customer_name order_date product_id product_price quantity revenue
   <chr>    <chr>         <date>     <chr>              <dbl>    <dbl>   <dbl>
 1 A0001    Great         2022-05-31 P0069              23000        1   23000
 2 A0002    Great         2022-06-01 P0070               2000        2    4000
 3 A0003    Elon          2022-06-01 P4234                  5    10000   50000
 4 A0004    Elon          2022-06-02 P4234                  5    20000  100000
 5 A0005    Great         2022-06-03 P4234                  5    30000  150000
 6 A0006    Larry         2022-05-31 P0069              23000        1   23000
 7 A0007    Great         2022-06-04 P4234                  5      300    1500
 8 A0008    Elon          2022-06-04 P4234                  5    40000  200000
 9 A0009    Elon          2022-06-05 P4234                  5    50000  250000
10 A0010    Great         2022-06-06 P4234                  5   100000  500000
```

```
4
5  arrange(filter(orders, product_price == 5), revenue)
```

Result

```
> arrange(filter(orders, product_price == 5), revenue)
# A tibble: 7 × 7
  order_id customer_name order_date product_id product_price quantity revenue
  <chr>    <chr>         <date>     <chr>              <dbl>    <dbl>   <dbl>
1 A0007    Great         2022-06-04 P4234                  5      300    1500
2 A0003    Elon          2022-06-01 P4234                  5    10000   50000
3 A0004    Elon          2022-06-02 P4234                  5    20000  100000
4 A0005    Great         2022-06-03 P4234                  5    30000  150000
5 A0008    Elon          2022-06-04 P4234                  5    40000  200000
6 A0009    Elon          2022-06-05 P4234                  5    50000  250000
7 A0010    Great         2022-06-06 P4234                  5   100000  500000
```

Nested Commands and Pipes

Pipes %>%

ตัวดำเนินการของ R ที่ส่งต่อผลลัพธ์ของฟังก์ชันแรก ไปยังฟังก์ชันถัดไปเป็นทอด ๆ เหมือนไหลไปตามท่อ

Syntax:

```
data %>%
  func1(...) %>%
  func2(...)
```

Note: สังเกตว่า เราไม่ต้องเอา data ตั้งต้นไปใส่ในวงเล็บของฟังก์ชันอีกต่อไปแล้ว เพราะ %>% ทำการ "ยัดค่า" data ใส่ฟังก์ชันตัวถัดไปให้เราเอง!

Samart Skills
A Grow with Google PROGRAM

Skooldio

Ex. จากตารางรายการสั่งซื้อสินค้า จงกรองเอาเฉพาะสินค้าราคา 5 บาท และแสดงผลเรียงลำดับรายได้ต่อการสั่งซื้อ จากน้อยไปมาก

```
5 arrange(filter(orders, product_price == 5), revenue)
6
```

```
7 orders %>% filter(product_price == 5) %>% arrange(revenue)
```

Results

```
> arrange(filter(orders, product_price == 5), revenue)
# A tibble: 7 × 7
  order_id customer_name order_date product_id product_price quantity revenue
  <chr>    <chr>         <date>     <chr>              <dbl>    <dbl>   <dbl>
1 A0007    Great         2022-06-04 P4234                  5      300    1500
2 A0003    Elon          2022-06-01 P4234                  5    10000   50000
3 A0004    Elon          2022-06-02 P4234                  5    20000  100000
4 A0005    Great         2022-06-03 P4234                  5    30000  150000
5 A0008    Elon          2022-06-04 P4234                  5    40000  200000
6 A0009    Elon          2022-06-05 P4234                  5    50000  250000
7 A0010    Great         2022-06-06 P4234                  5   100000  500000
```

```
> orders %>% filter(product_price == 5) %>% arrange(revenue)
# A tibble: 7 × 7
  order_id customer_name order_date product_id product_price quantity revenue
  <chr>    <chr>         <date>     <chr>              <dbl>    <dbl>   <dbl>
1 A0007    Great         2022-06-04 P4234                  5      300    1500
2 A0003    Elon          2022-06-01 P4234                  5    10000   50000
3 A0004    Elon          2022-06-02 P4234                  5    20000  100000
4 A0005    Great         2022-06-03 P4234                  5    30000  150000
5 A0008    Elon          2022-06-04 P4234                  5    40000  200000
6 A0009    Elon          2022-06-05 P4234                  5    50000  250000
7 A0010    Great         2022-06-06 P4234                  5   100000  500000
```

Nested Commands and Pipes

Pipes %>%

ตัวดำเนินการของ R ที่ส่งต่อผลลัพธ์ของฟังก์ชันแรก
ไปยังฟังก์ชันถัดไปเป็นทอด ๆ เหมือนไหลไปตามท่อ

Syntax:

```
data %>%
  func1(...) %>%
  func2(...)
```

Note: สังเกตว่า เราไม่ต้องเอา data ตั้งต้นไปใส่ในวงเล็บของฟังก์ชันอีกต่อไปแล้ว เพราะ %>% ทำการ "ยัดค่า" data ใส่ฟังก์ชันตัวถัดไปให้เราเอง!

Samart Skills
A Grow with Google PROGRAM

Skooldio

Ex. จากตารางรายการสั่งซื้อสินค้า จงกรองเอาเฉพาะสินค้าราคา 5 บาท
จากนั้นหาผลรวมรายได้แบ่งตามลูกค้าแต่ละคน

```
9  summarize(group_by(filter(orders, product_price == 5), customer_name), sum_revenue=sum(revenue))
10
```

```
11 orders %>%
12   filter(product_price == 5) %>%
13   group_by(customer_name) %>%
14   summarize(sum_revenue = sum(revenue))
```

Results

```
> summarize(group_by(filter(orders, product_price == 5), customer_name), sum_revenue=sum(revenue))
# A tibble: 2 × 2
  customer_name sum_revenue
  <chr>               <dbl>
1 Elon               600000
2 Great              651500
```

```
> orders %>%
+   filter(product_price == 5) %>%
+   group_by(customer_name) %>%
+   summarize(sum_revenue = sum(revenue))
# A tibble: 2 × 2
  customer_name sum_revenue
  <chr>               <dbl>
1 Elon               600000
2 Great              651500
```

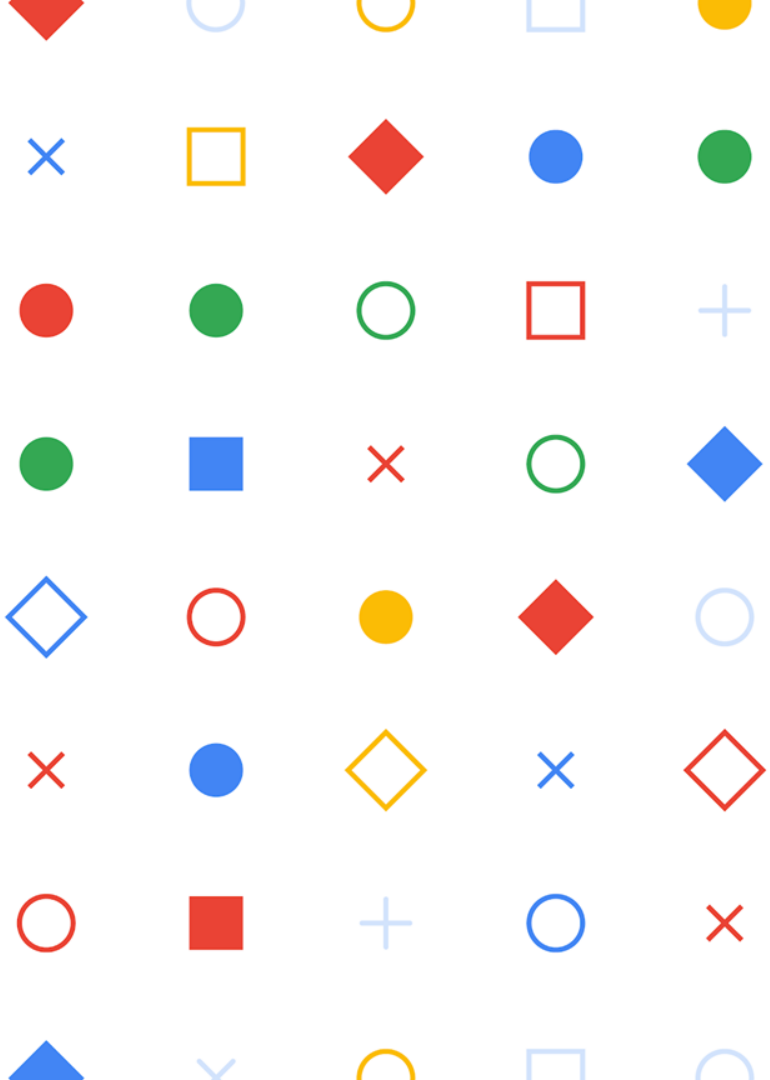
Working with Data in R

Part3:

1. Data Frames vs. Tibbles
2. Tidy data standards
3. Clean: Inspecting Data Frames
4. Clean: Selecting and Renaming Columns
5. Organize: Sorting and Filtering
6. Organize: Summarize and Group By
7. Transform: Separate, Unite and Mutate
8. Warning: Anscombe's Quartet
9. The Bias Function

Samart Skills
A Grow with Google PROGRAM

Skooldio



Data Frames vs. Tibbles

Data frames

โครงสร้างเก็บข้อมูลตารางแบบมาตรฐานของ R

Notes:

- คอลัมน์ควรมีชื่อที่สอดคล้องกับข้อมูลในแต่ละคอลัมน์ (Columns should be named)
- ข้อมูลในตารางมีได้หลาย Types ไม่ว่าจะเป็น Numeric, Factor หรือ Character
- แต่ละคอลัมน์ควรมีจำนวนข้อมูลที่เท่ากัน (Each column should contain the same number of data items) ซึ่งรวมถึงค่าว่าง (Missing values) ด้วย

Ex. โหลดตารางรายการสั่งซื้อสินค้าของเดือนมิถุนายน 2022 ใส่ตัวแปร orders

```
> orders <- read.csv('./data/orders_2022_06.csv')
> class(orders)
[1] "data.frame"
> orders
   Order.ID Customer.Name  Band Order.Date Product.ID Product.Price    QTT
1     A0001         Great SKD69 2022-05-31      P0069         23000      1
2     A0002         Great SKD69 2022-06-01      P0070          2000      2
3     A0003          Elon SHIBA 2022-06-01      P4234             5  10000
4     A0004          Elon SHIBA 2022-06-02      P4234             5  20000
5     A0005         Great SKD69 2022-06-03      P4234             5  30000
6     A0006         Larry GOOGL 2022-05-31      P0069         23000      1
7     A0007         Great SKD69 2022-06-04      P4234             5    300
8     A0008          Elon SHIBA 2022-06-04      P4234             5  40000
9     A0009          Elon SHIBA 2022-06-05      P4234             5  50000
10    A0010         Great SKD69 2022-06-06      P4234             5 100000
11    A0011        Sergey GOOGL 2022-06-08      P4234             5  10000
12    A0012          Jeff AMAZE 2022-06-10      P4234             5  20000
13    A0013          Jeff AMAZE 2022-06-11      P4234             5  30000
14    A0014        Sergey GOOGL 2022-06-12      P4234             5  40000
15    A0015        Sergey GOOGL 2022-06-13      P4234             5  50000
16    A0016        Sergey GOOGL 2022-06-14      P4234             5  60000
17    A0017          Jeff AMAZE 2022-06-15      P4234             5  70000
18    A0018          Mark FB555 2022-06-16      P4234             5     10
19    A0019          Mark FB555 2022-06-17      P4234             5     20
20    A0020          Mark FB555 2022-06-18      P4234             5     30
21    A0021        Sergey GOOGL 2022-06-19      P4234             5  10000
22    A0022        Sergey GOOGL 2022-06-20      P4234             5  20000
23    A0023        Sergey GOOGL 2022-06-24      P4234             5  30000
24    A0024        Sergey GOOGL 2022-06-27      P4234             5  40000
25    A0025        Sergey GOOGL 2022-06-30      P4234             5  50000
```

Data Frames vs. Tibbles

Tibbles

โครงสร้างเก็บข้อมูลตารางแบบพิเศษของ tidyverse

Notes:

- Tibble คือ Data frame ที่ถูกออกแบบมาให้ทำงานได้สะดวกคล่องตัวขึ้น (Streamlined) โดยมีสมบัติต่อไปนี้
  - ประเภทของข้อมูลแต่ละคอลัมน์จะไม่เปลี่ยนจากต้นฉบับ เช่น คอลัมน์ตัวเลขก็ต้องเป็นตัวเลขไปตลอด
  - ชื่อตัวแปรของแต่ละคอลัมน์จะไม่เปลี่ยน (เว้นแต่จะใช้ฟังก์ชันเพื่อเปลี่ยนชื่อโดยเฉพาะ)
  - ไม่มีชื่อแถว มีแต่หมายเลขแถว
  - เวลา Print จะแสดงเฉพาะ 10 แถวแรกเพื่อให้ไม่รกหน้าจอ console

Ex. เปลี่ยนตาราง orders จาก data.frame เป็น tibble โดยบันทึกใส่ตัวแปร orders_tb

```
> orders_tb <- as_tibble(orders)
> class(orders_tb)
[1] "tbl_df"     "tbl"        "data.frame"
> orders_tb
# A tibble: 25 × 7
   Order.ID Customer.Name Band  Order.Date Product.ID Product.Price   QTT
   <chr>    <chr>         <chr> <chr>      <chr>              <int> <int>
 1 A0001    Great         SKD69 2022-05-31 P0069              23000     1
 2 A0002    Great         SKD69 2022-06-01 P0070               2000     2
 3 A0003    Elon          SHIBA 2022-06-01 P4234                  5 10000
 4 A0004    Elon          SHIBA 2022-06-02 P4234                  5 20000
 5 A0005    Great         SKD69 2022-06-03 P4234                  5 30000
 6 A0006    Larry         GOOGL 2022-05-31 P0069              23000     1
 7 A0007    Great         SKD69 2022-06-04 P4234                  5   300
 8 A0008    Elon          SHIBA 2022-06-04 P4234                  5 40000
 9 A0009    Elon          SHIBA 2022-06-05 P4234                  5 50000
10 A0010    Great         SKD69 2022-06-06 P4234                  5 100000
# … with 15 more rows
```

Tidy Data Standards

Tidy data คือมาตรฐานการจัดการข้อมูลในระบบนิเวศของ R ซึ่งมี 3 ข้อดังนี้

order_id	name	order_date	product
A0001	Great	2022-05-31	PS5
A0002	Great	2022-06-01	LAC
A0003	Elon	2022-06-01	PS5
A0004	Elon	2022-06-02	LAC
A0005	Great	2022-06-03	LAC
A0006	Larry	2022-05-31	PS5

1 **Variables are organized into columns**

ตัวแปรถูกจัดให้อยู่ในรูปคอลัมน์

order_id	name	order_date	product
A0001	Great	2022-05-31	PS5
A0002	Great	2022-06-01	LAC
A0003	Elon	2022-06-01	PS5
A0004	Elon	2022-06-02	LAC
A0005	Great	2022-06-03	LAC
A0006	Larry	2022-05-31	PS5

2 **Observations are organized into rows**

1 แถวคือ 1 รายการ

order_id	name	order_date	product
A0001	Great	2022-05-31	PS5
A0002	Great	2022-06-01	LAC
A0003	Elon	2022-06-01	PS5
A0004	Elon	2022-06-02	LAC
A0005	Great	2022-06-03	LAC
A0006	Larry	2022-05-31	PS5

3 **Each value must have its own cell**

1 ช่อง (cell) ในตารางมีข้อมูลได้ 1 ค่า

Source: https://r4ds.had.co.nz/tidy-data.html

Clean: Inspecting Data Frames

เราสำรวจตารางเบื้องต้นได้ด้วยฟังก์ชันต่อไปนี้

Syntax:

```
View(table)          #ดูตารางแบบเต็มๆ
head(table)          #ดู 6 แถวบน
colnames(table)      #ดูชื่อคอลัมน์ทั้งหมด
#ดูโครงสร้างตาราง
str(table)
glimpse(table)
#ดูสรุปข้อมูลในตารางคร่าว ๆ
skim_without_charts(table)
```

Note: glimpse() เป็นของแพคเกจ dplyr และ
skim_without_charts() เป็นของแพคเกจ skimr

Samart Skills
A Grow with Google PROGRAM

Skooldio

Ex. ดูตาราง orders_tb แบบเต็มๆ

```
> View(orders_tb)
```

V ใน View ต้องตัวพิมพ์ใหญ่!!!

orders_tb | test.R | Untitled1*

Filter

	Order.ID	Customer.Name	Band	Order.Date	Product.ID	Product.Price	QTT
1	A0001	Great	SKD69	2022-05-31	P0069	23000	1e+00
2	A0002	Great	SKD69	2022-06-01	P0070	2000	2e+00
3	A0003	Elon	SHIBA	2022-06-01	P4234	5	1e+04
4	A0004	Elon	SHIBA	2022-06-02	P4234	5	2e+04
5	A0005	Great	SKD69	2022-06-03	P4234	5	3e+04
6	A0006	Larry	GOOGL	2022-05-31	P0069	23000	1e+00
7	A0007	Great	SKD69	2022-06-04	P4234	5	3e+02
8	A0008	Elon	SHIBA	2022-06-04	P4234	5	4e+04
9	A0009	Elon	SHIBA	2022-06-05	P4234	5	5e+04
10	A0010	Great	SKD69	2022-06-06	P4234	5	1e+05
11	A0011	Sergey	GOOGL	2022-06-08	P4234	5	1e+04
12	A0012	Jeff	AMAZE	2022-06-10	P4234	5	2e+04
13	A0013	Jeff	AMAZE	2022-06-11	P4234	5	3e+04
14	A0014	Sergey	GOOGL	2022-06-12	P4234	5	4e+04
15	A0015	Sergey	GOOGL	2022-06-13	P4234	5	5e+04
16	A0016	Sergey	GOOGL	2022-06-14	P4234	5	6e+04
17	A0017	Jeff	AMAZE	2022-06-15	P4234	5	7e+04
18	A0018	Mark	FB555	2022-06-16	P4234	5	1e+01
19	A0019	Mark	FB555	2022-06-17	P4234	5	2e+01
20	A0020	Mark	FB555	2022-06-18	P4234	5	3e+01
21	A0021	Sergey	GOOGL	2022-06-19	P4234	5	1e+04
22	A0022	Sergey	GOOGL	2022-06-20	P4234	5	2e+04
23	A0023	Sergey	GOOGL	2022-06-24	P4234	5	3e+04
24	A0024	Sergey	GOOGL	2022-06-27	P4234	5	4e+04
25	A0025	Sergey	GOOGL	2022-06-30	P4234	5	5e+04

Showing 1 to 25 of 25 entries, 7 total columns

Clean: Inspecting Data Frames

เราสำรวจตารางเบื้องต้นได้ด้วยฟังก์ชันต่อไปนี้

Syntax:

```
View(table)          #ดูตารางแบบเต็มๆ
head(table)          #ดู 6 แถวบน
colnames(table)      #ดูชื่อคอลัมน์ทั้งหมด
#ดูโครงสร้างตาราง
str(table)
glimpse(table)
#ดูสรุปข้อมูลในตารางคร่าว ๆ
skim_without_charts(table)
```

Note: glimpse() เป็นของแพคเกจ dplyr และ skim_without_charts() เป็นของแพคเกจ skimr

Ex. ดู 6 แถวแรกของตาราง orders_tb และสรุปรายชื่อคอลัมน์ทั้งหมดของตาราง

```
> head(orders_tb)
# A tibble: 6 × 7
  Order.ID Customer.Name Band  Order.Date Product.ID Product.Price   QTT
  <chr>    <chr>         <chr> <chr>      <chr>              <int> <int>
1 A0001    Great         SKD69 2022-05-31 P0069              23000     1
2 A0002    Great         SKD69 2022-06-01 P0070               2000     2
3 A0003    Elon          SHIBA 2022-06-01 P4234                  5 10000
4 A0004    Elon          SHIBA 2022-06-02 P4234                  5 20000
5 A0005    Great         SKD69 2022-06-03 P4234                  5 30000
6 A0006    Larry         GOOGL 2022-05-31 P0069              23000     1
> colnames(orders_tb)
[1] "Order.ID"      "Customer.Name" "Band"          "Order.Date"    "Product.ID"
[6] "Product.Price" "QTT"
```

Clean: Inspecting Data Frames

เราสำรวจตารางเบื้องต้นได้ด้วยฟังก์ชันต่อไปนี้

Syntax:

```
View(table)          #ดูตารางแบบเต็มๆ
head(table)          #ดู 6 แถวบน
colnames(table)      #ดูชื่อคอลัมน์ทั้งหมด
#ดูโครงสร้างตาราง
str(table)
glimpse(table)
#ดูสรุปข้อมูลในตารางคร่าว ๆ
skim_without_charts(table)
```

Note: glimpse() เป็นของแพคเกจ dplyr และ skim_without_charts() เป็นของแพคเกจ skimr

Ex. ดูโครงสร้างตาราง orders_tb

```
> str(orders_tb)
tibble [25 × 7] (S3: tbl_df/tbl/data.frame)
 $ Order.ID     : chr [1:25] "A0001" "A0002" "A0003" "A0004" ...
 $ Customer.Name: chr [1:25] "Great" "Great" "Elon" "Elon" ...
 $ Band         : chr [1:25] "SKD69" "SKD69" "SHIBA" "SHIBA" ...
 $ Order.Date   : chr [1:25] "2022-05-31" "2022-06-01" "2022-06-01" "2022-06-02" ...
 $ Product.ID   : chr [1:25] "P0069" "P0070" "P4234" "P4234" ...
 $ Product.Price: int [1:25] 23000 2000 5 5 5 23000 5 5 5 5 ...
 $ QTT          : int [1:25] 1 2 10000 20000 30000 1 300 40000 50000 100000 ...
> glimpse(orders_tb)
Rows: 25
Columns: 7
$ Order.ID      <chr> "A0001", "A0002", "A0003", "A0004", "A0005", "A0006", "A0007…
$ Customer.Name <chr> "Great", "Great", "Elon", "Elon", "Great", "Larry", "Great",…
$ Band          <chr> "SKD69", "SKD69", "SHIBA", "SHIBA", "SKD69", "GOOGL", "SKD69…
$ Order.Date    <chr> "2022-05-31", "2022-06-01", "2022-06-01", "2022-06-02", "202…
$ Product.ID    <chr> "P0069", "P0070", "P4234", "P4234", "P4234", "P0069", "P4234…
$ Product.Price <int> 23000, 2000, 5, 5, 5, 23000, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5…
$ QTT           <int> 1, 2, 10000, 20000, 30000, 1, 300, 40000, 50000, 100000, 100…
```

Clean: Inspecting Data Frames

เราสำรวจตารางเบื้องต้นได้ด้วยฟังก์ชันต่อไปนี้

Syntax:

```
View(table)          #ดูตารางแบบเต็มๆ
head(table)          #ดู 6 แถวบน
colnames(table)      #ดูชื่อคอลัมน์ทั้งหมด
#ดูโครงสร้างตาราง
str(table)
glimpse(table)
#ดูสรุปข้อมูลในตารางคร่าว ๆ
skim_without_charts(table)
```

Note: glimpse() เป็นของแพคเกจ dplyr และ skim_without_charts() เป็นของแพคเกจ skimr

Ex. ดูสรุปข้อมูลในตาราง orders_tb แบบคร่าว ๆ

```
> skim_without_charts(orders_tb)
── Data Summary ────────────────────────
                           Values
Name                       orders_tb
Number of rows             25
Number of columns          7
_________________________
Column type frequency:
  character                5
  numeric                  2
_________________________
Group variables            None

── Variable type: character ────────────────────────────────────────────────────
  skim_variable n_missing complete_rate min max empty n_unique whitespace
1 Order.ID              0             1   5   5     0       25          0
2 Customer.Name         0             1   4   6     0        6          0
3 Band                  0             1   5   5     0        5          0
4 Order.Date            0             1  10  10     0       22          0
5 Product.ID            0             1   5   5     0        3          0

── Variable type: numeric ──────────────────────────────────────────────────────
  skim_variable n_missing complete_rate   mean     sd p0 p25   p50   p75   p100
1 Product.Price         0             1  1924.  6356.  5   5     5     5  23000
2 QTT                   0             1 27215. 26208.  1 300 20000 40000 100000
```

Samart Skills A Grow with Google PROGRAM | Skooldio

Clean: Selecting and Renaming Columns

เราเลือกบางคอลัมน์มาใช้ได้ด้วยฟังก์ชัน select()

Syntax:

```
#เลือกคอลัมน์ col1 และ col2 จากตาราง
select(table, col1, col2, ...)
#ใช้วิธี Pipe ก็ได้
table %>%
  select(col1, col2, ...)
#เลือกทุกคอลัมน์ยกเว้น col1
table %>%
  select(-col1)
```

Note: หากต้องการเก็บผลไว้ใช้ต่อ ต้องบันทึกใส่ตัวแปรใหม่ เช่น bands <- select(orders_tb, Band)

Ex. เลือกคอลัมน์ Customer.Name และ Band

```
> select(orders_tb, Customer.Name, Band)
# A tibble: 25 × 2
   Customer.Name Band
   <chr>         <chr>
 1 Great         SKD69
 2 Great         SKD69
 3 Elon          SHIBA
 4 Elon          SHIBA
 5 Great         SKD69
 6 Larry         GOOGL
 7 Great         SKD69
 8 Elon          SHIBA
 9 Elon          SHIBA
10 Great         SKD69
# … with 15 more rows
```

```
> orders_tb %>% select(Customer.Name, Band)
# A tibble: 25 × 2
   Customer.Name Band
   <chr>         <chr>
 1 Great         SKD69
 2 Great         SKD69
 3 Elon          SHIBA
 4 Elon          SHIBA
 5 Great         SKD69
 6 Larry         GOOGL
 7 Great         SKD69
 8 Elon          SHIBA
 9 Elon          SHIBA
10 Great         SKD69
# … with 15 more rows
```

Ex. เลือกทุกคอลัมน์ยกเว้น Band มาดู

```
> orders_tb %>% select(-Band)
# A tibble: 25 × 6
   Order.ID Customer.Name Order.Date Product.ID Product.Price   QTT
   <chr>    <chr>         <chr>      <chr>              <int> <int>
 1 A0001    Great         2022-05-31 P0069              23000     1
 2 A0002    Great         2022-06-01 P0070               2000     2
 3 A0003    Elon          2022-06-01 P4234                  5 10000
 4 A0004    Elon          2022-06-02 P4234                  5 20000
 5 A0005    Great         2022-06-03 P4234                  5 30000
 6 A0006    Larry         2022-05-31 P0069              23000     1
 7 A0007    Great         2022-06-04 P4234                  5   300
 8 A0008    Elon          2022-06-04 P4234                  5 40000
 9 A0009    Elon          2022-06-05 P4234                  5 50000
10 A0010    Great         2022-06-06 P4234                  5 100000
# … with 15 more rows
```

Clean: Selecting and Renaming Columns

มีหลายฟังก์ชันที่ใช้เปลี่ยนชื่อคอลัมน์ให้เลือกใช้

Syntax:

```
#เปลี่ยนคอลัมน์เดียว
rename(table, ชื่อcolใหม่=ชื่อcolเดิม)
#เปลี่ยนทีเดียวทุกคอลัมน์ เช่น เป็นตัวพิมพ์เล็ก
rename_with(table, tolower)
#เปลี่ยนให้ "สะอาด" ตามมาตรฐาน Tidy data
clean_names(table)
```

Notes:

- rename() และ rename_with() เป็นของแพคเกจ dplyr ส่วน clean_names() เป็นของแพคเกจ janitor
- ถ้าลืมบันทึกผลใส่ตัวแปร!
  เช่น orders_tb <- clean_names(orders_tb)

Ex. เปลี่ยนชื่อคอลัมน์ QTT เป็น Quantity

```
> orders_tb <- rename(orders_tb, Quantity = QTT)
```

```
> head(orders_tb)
# A tibble: 6 × 7
  Order.ID Customer.Name Band  Order.Date Product.ID Product.Price Quantity
  <chr>    <chr>         <chr> <chr>      <chr>              <int>    <int>
1 A0001    Great         SKD69 2022-05-31 P0069              23000        1
2 A0002    Great         SKD69 2022-06-01 P0070               2000        2
3 A0003    Elon          SHIBA 2022-06-01 P4234                  5    10000
4 A0004    Elon          SHIBA 2022-06-02 P4234                  5    20000
5 A0005    Great         SKD69 2022-06-03 P4234                  5    30000
6 A0006    Larry         GOOGL 2022-05-31 P0069              23000        1
```

Ex. เปลี่ยนชื่อคอลัมน์ทั้งหมดให้ "สะอาด" ด้วย clean_names()

```
> orders_tb <- clean_names(orders_tb)
```

```
> head(orders_tb)
# A tibble: 6 × 7
  order_id customer_name band  order_date product_id product_price quantity
  <chr>    <chr>         <chr> <chr>      <chr>              <int>    <int>
1 A0001    Great         SKD69 2022-05-31 P0069              23000        1
2 A0002    Great         SKD69 2022-06-01 P0070               2000        2
3 A0003    Elon          SHIBA 2022-06-01 P4234                  5    10000
4 A0004    Elon          SHIBA 2022-06-02 P4234                  5    20000
5 A0005    Great         SKD69 2022-06-03 P4234                  5    30000
6 A0006    Larry         GOOGL 2022-05-31 P0069              23000        1
```

Organize: Sorting and Filtering

เรียงลำดับข้อมูลได้ด้วยคำสั่ง arrange()

Syntax: เรียงจากน้อยไปมาก (Ascending)

```
arrange(table, col)
```
หรือ
```
table %>%
  arrange(col)
```

Syntax: เรียงจากมากไปน้อย (Descending)

```
arrange(table, -col)
```
หรือ
```
arrange(table, desc(col))
```
หรือ
```
table %>%
  arrange(desc(col))
```

Samart Skills
A Grow with Google PROGRAM

Skooldio

Ex. เรียงลำดับการแสดงผลจาก quantity น้อยไปมาก

```
> arrange(orders_tb, quantity)
```
หรือ
```
> orders_tb %>% arrange(quantity)
# A tibble: 25 × 7
   order_id customer_name band  order_date product_id product_price quantity
   <chr>    <chr>         <chr> <chr>      <chr>              <int>    <int>
 1 A0001    Great         SKD69 2022-05-31 P0069              23000        1
 2 A0006    Larry         GOOGL 2022-05-31 P0069              23000        1
 3 A0002    Great         SKD69 2022-06-01 P0070               2000        2
 4 A0018    Mark          FB555 2022-06-16 P4234                  5       10
 5 A0019    Mark          FB555 2022-06-17 P4234                  5       20
 6 A0020    Mark          FB555 2022-06-18 P4234                  5       30
 7 A0007    Great         SKD69 2022-06-04 P4234                  5      300
 8 A0003    Elon          SHIBA 2022-06-01 P4234                  5    10000
 9 A0011    Sergey        GOOGL 2022-06-08 P4234                  5    10000
10 A0021    Sergey        GOOGL 2022-06-19 P4234                  5    10000
# … with 15 more rows
```

Ex. เรียงลำดับการแสดงผลจาก quantity มากไปน้อย

```
> arrange(orders_tb, desc(quantity))
```
หรือ
```
> orders_tb %>% arrange(-quantity)
```
หรือ
```
> orders_tb %>% arrange(desc(quantity))
# A tibble: 25 × 7
   order_id customer_name band  order_date product_id product_price quantity
   <chr>    <chr>         <chr> <chr>      <chr>              <int>    <int>
 1 A0010    Great         SKD69 2022-06-06 P4234                  5   100000
 2 A0017    Jeff          AMAZE 2022-06-15 P4234                  5    70000
 3 A0016    Sergey        GOOGL 2022-06-14 P4234                  5    60000
 4 A0009    Elon          SHIBA 2022-06-05 P4234                  5    50000
 5 A0015    Sergey        GOOGL 2022-06-13 P4234                  5    50000
 6 A0025    Sergey        GOOGL 2022-06-30 P4234                  5    50000
 7 A0008    Elon          SHIBA 2022-06-04 P4234                  5    40000
 8 A0014    Sergey        GOOGL 2022-06-12 P4234                  5    40000
 9 A0024    Sergey        GOOGL 2022-06-27 P4234                  5    40000
10 A0005    Great         SKD69 2022-06-03 P4234                  5    30000
# … with 15 more rows
```

Organize: Sorting and Filtering

กรองข้อมูลได้ด้วยคำสั่ง filter()

Syntax:

```
filter(table, เงื่อนไขของสิ่งที่คงอยู่)
```

```
table %>%
  filter(เงื่อนไขของสิ่งที่คงอยู่)
```

Ex. เอาเฉพาะการสั่งซื้อสินค้าที่มีราคาเกิน 10 บาท

```
> filter(orders_tb, product_price > 10)
```

หรือ

```
> orders_tb %>% filter(product_price > 10)
# A tibble: 3 × 7
  order_id customer_name band  order_date product_id product_price quantity
  <chr>    <chr>         <chr> <chr>      <chr>              <int>    <int>
1 A0001    Great         SKD69 2022-05-31 P0069              23000        1
2 A0002    Great         SKD69 2022-06-01 P0070               2000        2
3 A0006    Larry         GOOGL 2022-05-31 P0069              23000        1
```

Ex. เอาเฉพาะการสั่งซื้อสินค้าโดยคนชื่อ Great

```
> filter(orders_tb, customer_name == 'Great')
```

หรือ

```
> orders_tb %>% filter(customer_name == 'Great')
# A tibble: 5 × 7
  order_id customer_name band  order_date product_id product_price quantity
  <chr>    <chr>         <chr> <chr>      <chr>              <int>    <int>
1 A0001    Great         SKD69 2022-05-31 P0069              23000        1
2 A0002    Great         SKD69 2022-06-01 P0070               2000        2
3 A0005    Great         SKD69 2022-06-03 P4234                  5    30000
4 A0007    Great         SKD69 2022-06-04 P4234                  5      300
5 A0010    Great         SKD69 2022-06-06 P4234                  5   100000
```

Organize: Summarize and Group By

สรุปข้อมูลสถิติด้วย summarize() และแบบแบ่งกลุ่มด้วย group_by() ตามด้วย summarize()

Syntax:

```
summarize(table, mean(col))
```

```
table %>%
  summarize(min(col),max(col))
```

Ex. สรุปค่าเฉลี่ย ค่ามัธยฐาน ค่าต่ำสุดและค่าสูงสุดของ quantity

```
> summarize(orders_tb, mean(quantity), median(quantity), min(quantity), max(quantity))
```

หรือ

```
> orders_tb %>% summarize(mean(quantity), median(quantity), min(quantity), max(quantity))
# A tibble: 1 × 4
  `mean(quantity)` `median(quantity)` `min(quantity)` `max(quantity)`
             <dbl>              <int>           <int>           <int>
1           27215.              20000               1          100000
```

Ex. ตั้งชื่อคอลัมน์ของผลลัพธ์ว่า mean_qtt, median_qtt, min_qtt และ max_qtt

```
1 orders_tb %>%
2   summarize(mean_qtt = mean(quantity),
3             median_qtt = median(quantity),
4             min_qtt = min(quantity),
5             max_qtt = max(quantity))
```

```
# A tibble: 1 × 4
  mean_qtt median_qtt min_qtt max_qtt
     <dbl>      <int>   <int>   <int>
1   27215.      20000       1  100000
```

Samart Skills
A Grow with Google PROGRAM

Skooldio

Organize: Summarize and Group By

สรุปข้อมูลสถิติด้วย summarize() และแบบแบ่งกลุ่มด้วย group_by() ตามด้วย summarize()

Syntax:

```
table %>%
  group_by(col1) %>%
  summarize(sum(col2),mean(col3))
```

Ex. สรุปผลรวมของ quantity สินค้ารหัส P4234 แบ่งตาม customer_name และเรียงผลลัพธ์จากมากไปหาน้อย

```
1 orders_tb %>%
2   filter(product_id == 'P4234') %>%
3   group_by(customer_name) %>%
4   summarize(total_quantity = sum(quantity)) %>%
5   arrange(desc(total_quantity))
```

```
# A tibble: 5 × 2
  customer_name total_quantity
  <chr>                  <int>
1 Sergey                310000
2 Great                 130300
3 Elon                  120000
4 Jeff                  120000
5 Mark                      60
```

Samart Skills
A Grow with Google PROGRAM

Skooldio

Transform: Separate, Unite and Mutate

เราแยกข้อความออกเป็นส่วน ๆ ได้ด้วย separate()

Syntax:

```
separate(table, คอลัมน์ตั้งต้น,
         into=c(รายชื่อคอลัมน์ที่แยกตัว),
         sep='ตัวคั่น')
```

Notes:

- หากต้องการเก็บผลไว้ใช้ต่อ ต้องบันทึกใส่ตัวแปรใหม่ เช่น orders_tb <- separate(orders_tb, order_date, ...)
- หากไม่ต้องการลบคอลัมน์ตั้งต้นทิ้ง ให้ใส่ remove=FALSE เพิ่มเข้าไปใน arguments

Ex. แยกคอลัมน์ order_date ออกเป็น 3 คอลัมน์ย่อยคือ year, month และ date

```
> head(orders_tb)
# A tibble: 6 × 7
  order_id customer_name band  order_date product_id product_price quantity
  <chr>    <chr>         <chr> <chr>      <chr>              <int>    <int>
1 A0001    Great         SKD69 2022-05-31 P0069              23000        1
2 A0002    Great         SKD69 2022-06-01 P0070               2000        2
3 A0003    Elon          SHIBA 2022-06-01 P4234                  5    10000
4 A0004    Elon          SHIBA 2022-06-02 P4234                  5    20000
5 A0005    Great         SKD69 2022-06-03 P4234                  5    30000
6 A0006    Larry         GOOGL 2022-05-31 P0069              23000        1
```

```
> separate(orders_tb, order_date, into=c('year','month','date'), sep='-')
# A tibble: 25 × 9
   order_id customer_name band  year  month date  product_id product_price quantity
   <chr>    <chr>         <chr> <chr> <chr> <chr> <chr>              <int>    <int>
 1 A0001    Great         SKD69 2022  05    31    P0069              23000        1
 2 A0002    Great         SKD69 2022  06    01    P0070               2000        2
 3 A0003    Elon          SHIBA 2022  06    01    P4234                  5    10000
 4 A0004    Elon          SHIBA 2022  06    02    P4234                  5    20000
 5 A0005    Great         SKD69 2022  06    03    P4234                  5    30000
 6 A0006    Larry         GOOGL 2022  05    31    P0069              23000        1
 7 A0007    Great         SKD69 2022  06    04    P4234                  5      300
 8 A0008    Elon          SHIBA 2022  06    04    P4234                  5    40000
 9 A0009    Elon          SHIBA 2022  06    05    P4234                  5    50000
10 A0010    Great         SKD69 2022  06    06    P4234                  5   100000
# … with 15 more rows
```

Transform: Separate, Unite and Mutate

เรารวมข้อความย่อยเข้าด้วยกันด้วย unite()

Syntax:

```
unite(table, 'ชื่อคอลัมน์ใหม่',
      col1, col2, ...,
      sep='ตัวคั่น')
```

Notes:

- หากต้องการเก็บผลไว้ใช้ต่อ ต้องบันทึกใส่ตัวแปรใหม่ เช่น orders_tb <- unite(orders_tb, 'stage_name', ...)
- หากไม่ต้องการลบคอลัมน์ตั้งต้นทิ้ง ให้ใส่ remove=FALSE เพิ่มเข้าไปใน arguments

Ex. รวมคอลัมน์ customer_name และ band คั่นด้วยเว้นวรรค (whitespace) เรียกชื่อคอลัมน์ที่รวมแล้วว่า stage_name

```
> head(orders_tb)
# A tibble: 6 × 7
  order_id customer_name band  order_date product_id product_price quantity
  <chr>    <chr>         <chr> <chr>      <chr>              <int>    <int>
1 A0001    Great         SKD69 2022-05-31 P0069              23000        1
2 A0002    Great         SKD69 2022-06-01 P0070               2000        2
3 A0003    Elon          SHIBA 2022-06-01 P4234                  5    10000
4 A0004    Elon          SHIBA 2022-06-02 P4234                  5    20000
5 A0005    Great         SKD69 2022-06-03 P4234                  5    30000
6 A0006    Larry         GOOGL 2022-05-31 P0069              23000        1
```

```
> unite(orders_tb, 'stage_name', customer_name, band, sep=' ')
# A tibble: 25 × 6
   order_id stage_name  order_date product_id product_price quantity
   <chr>    <chr>       <chr>      <chr>              <int>    <int>
 1 A0001    Great SKD69 2022-05-31 P0069              23000        1
 2 A0002    Great SKD69 2022-06-01 P0070               2000        2
 3 A0003    Elon SHIBA  2022-06-01 P4234                  5    10000
 4 A0004    Elon SHIBA  2022-06-02 P4234                  5    20000
 5 A0005    Great SKD69 2022-06-03 P4234                  5    30000
 6 A0006    Larry GOOGL 2022-05-31 P0069              23000        1
 7 A0007    Great SKD69 2022-06-04 P4234                  5      300
 8 A0008    Elon SHIBA  2022-06-04 P4234                  5    40000
 9 A0009    Elon SHIBA  2022-06-05 P4234                  5    50000
10 A0010    Great SKD69 2022-06-06 P4234                  5   100000
# … with 15 more rows
```

Transform: Separate, Unite and Mutate

เรานำค่าในคอลัมน์ต่าง ๆ มาแปลงแล้วสร้างเป็นคอลัมน์ใหม่ได้ด้วย mutate()

Syntax:

```
mutate(table, new_col = ...)
```

```
table %>%
  mutate(new_col = ...)
```

Note: หากต้องการเก็บผลไว้ใช้ต่อ ต้องบันทึกใส่ตัวแปรใหม่
เช่น orders_tb <- mutate(orders_tb, revenue=...)

Ex. สร้างคอลัมน์ใหม่ชื่อ revenue ซึ่งเกิดจากการนำเอา product_price * quantity พร้อมทั้งบันทึกใส่ตารางเดิมด้วย

```
> orders_tb <- mutate(orders_tb, revenue = product_price * quantity)
> orders_tb
# A tibble: 25 × 8
   order_id customer_name band  order_date product_id product_price quantity revenue
   <chr>    <chr>         <chr> <chr>      <chr>              <int>    <int>   <int>
 1 A0001    Great         SKD69 2022-05-31 P0069              23000        1   23000
 2 A0002    Great         SKD69 2022-06-01 P0070               2000        2    4000
 3 A0003    Elon          SHIBA 2022-06-01 P4234                  5    10000   50000
 4 A0004    Elon          SHIBA 2022-06-02 P4234                  5    20000  100000
 5 A0005    Great         SKD69 2022-06-03 P4234                  5    30000  150000
 6 A0006    Larry         GOOGL 2022-05-31 P0069              23000        1   23000
 7 A0007    Great         SKD69 2022-06-04 P4234                  5      300    1500
 8 A0008    Elon          SHIBA 2022-06-04 P4234                  5    40000  200000
 9 A0009    Elon          SHIBA 2022-06-05 P4234                  5    50000  250000
10 A0010    Great         SKD69 2022-06-06 P4234                  5   100000  500000
# … with 15 more rows
```

Ex. เปลี่ยนคอลัมน์ revenue จาก int เป็น double (ทศนิยม) คราวนี้ลองใช้วิธี Pipe

```
> orders_tb <- orders_tb %>% mutate(revenue = as.double(revenue))
> orders_tb
# A tibble: 25 × 8
   order_id customer_name band  order_date product_id product_price quantity revenue
   <chr>    <chr>         <chr> <chr>      <chr>              <int>    <int>   <dbl>
 1 A0001    Great         SKD69 2022-05-31 P0069              23000        1   23000
 2 A0002    Great         SKD69 2022-06-01 P0070               2000        2    4000
 3 A0003    Elon          SHIBA 2022-06-01 P4234                  5    10000   50000
 4 A0004    Elon          SHIBA 2022-06-02 P4234                  5    20000  100000
 5 A0005    Great         SKD69 2022-06-03 P4234                  5    30000  150000
 6 A0006    Larry         GOOGL 2022-05-31 P0069              23000        1   23000
 7 A0007    Great         SKD69 2022-06-04 P4234                  5      300    1500
 8 A0008    Elon          SHIBA 2022-06-04 P4234                  5    40000  200000
 9 A0009    Elon          SHIBA 2022-06-05 P4234                  5    50000  250000
10 A0010    Great         SKD69 2022-06-06 P4234                  5   100000  500000
# … with 15 more rows
```

Warning: Anscombe's quartet

Anscombe's quartet คือชุดข้อมูล 4 ชุดที่แตกต่างกันสิ้นเชิง แต่กลับให้ค่า mean, sd และ correlation ที่เหมือนกันเปะ เพื่อเป็นการย้ำเตือนว่า เราไม่ควรดูแค่ค่าสรุปทางสถิติ แต่ควรดูธรรมชาติข้อมูลดิบด้วย

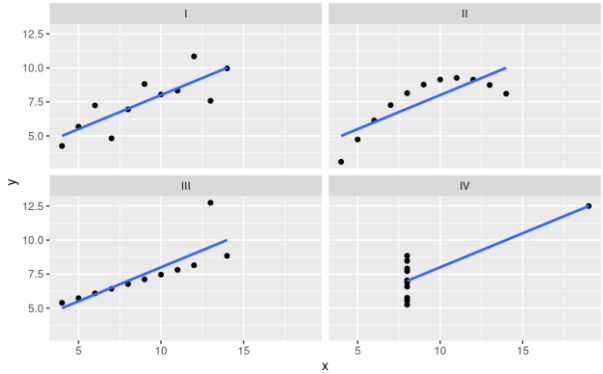
```
1  install.packages("Tmisc")
2  library("Tmisc")
3  data(quartet)
4  View(quartet)
```

	set	x	y
1	I	10	8.04
2	I	8	6.95
3	I	13	7.58
4	I	9	8.81
5	I	11	8.33
6	I	14	9.96
7	I	6	7.24
8	I	4	4.26
9	I	12	10.84
10	I	7	4.82
11	I	5	5.68
12	II	10	9.14
13	II	8	8.14
14	II	13	8.74
15	II	9	8.77
16	II	11	9.26
17	II	14	8.10
18	II	6	6.13
19	II	4	3.10
20	II	12	9.13
21	II	7	7.26
22	II	5	4.74
23	III	10	7.46
24	III	8	6.77
25	III	13	12.74
26	III	9	7.11
27	III	11	7.81
28	III	14	8.84
29	III	6	6.08
30	III	4	5.39
31	III	12	8.15
32	III	7	6.42
33	III	5	5.73
34	IV	8	6.58
35	IV	8	5.76
36	IV	8	7.71
37	IV	8	8.84
38	IV	8	8.47
39	IV	8	7.04
40	IV	8	5.25
41	IV	19	12.50
42	IV	8	5.56
43	IV	8	7.91
44	IV	8	6.89

```
6  quartet %>%
7    group_by(set) %>%
8    summarize(mean(x),sd(x),mean(y),sd(y),cor(x,y))
9
```

```
# A tibble: 4 × 6
  set   `mean(x)` `sd(x)` `mean(y)` `sd(y)` `cor(x, y)`
  <fct>     <dbl>   <dbl>     <dbl>   <dbl>       <dbl>
1 I             9    3.32      7.50    2.03       0.816
2 II            9    3.32      7.50    2.03       0.816
3 III           9    3.32      7.5     2.03       0.816
4 IV            9    3.32      7.50    2.03       0.817
```

```
10 ggplot(quartet,aes(x,y)) +
11   geom_point() +
12   geom_smooth(method=lm, se=FALSE) +
13   facet_wrap(~set)
```



The Bias Function

เราดูว่าค่าจริง (Actual values) กับค่าทำนาย (Predicted values) ต่างกันมากแค่ไหนด้วยฟังก์ชัน bias()

Syntax:

```
bias(actual_vec, predicted_vec)
```

Notes:

- ถ้าผลลัพธ์เป็น 0 แปลว่า actual กับ predicted ตรงกันเปะ นั่นคือ ไม่ลำเอียง (unbiased)
- ยิ่งผลลัพธ์มีค่าต่างจาก 0 มาก แปลว่า predicted ยิ่งต่างจาก actual มาก ก็คือยิ่ง biased มาก
- ฟังก์ชัน bias() เป็นของแพคเกจ SimDesign ซึ่งต้อง install.packages("SimDesign") และโหลดด้วย library("SimDesign") ก่อน

Ex. ดูว่าค่าจริงกับค่าทำนายของยอด likes และ shares ในช่วง 5 วัน ต่างกันมากน้อยแค่ไหน

```
1  install.packages("SimDesign")
2  library("SimDesign")
3
4  actual_likes = c(100, 200, 300, 400, 500)
5  predicted_likes = c(100, 200, 300, 400, 500)
```

```
> bias(actual_likes, predicted_likes)
[1] 0
```

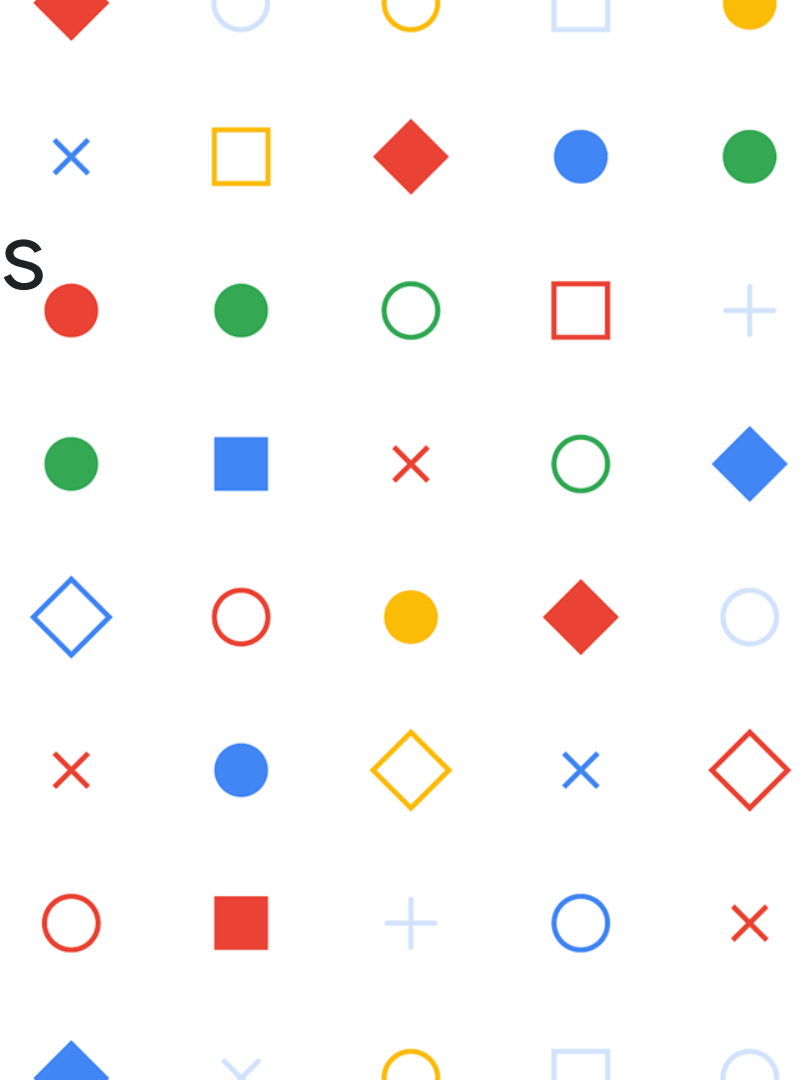
```
8  actual_shares = c(10, 20, 30, 40, 50)
9  predicted_shares = c(100, 200, 300, 400, 500)
```

```
> bias(actual_shares, predicted_shares)
[1] -270
> bias(predicted_shares, actual_shares)
[1] 270
```

More about Visualizations, Aesthetics and Annotations

Part4:

1. Basic Components of ggplot2
2. Example: Basic Scatter Plots
3. Example: Scatter Plots with Trend Line
4. Example: Bar Charts
5. Example: Facet Functions
6. Example: Labels and Annotations
7. Saving Your Visualizations



Samart Skills
A Grow with Google PROGRAM

Skooldio

Basic Components of ggplot2

ggplot2 คือ library ที่ใช้สร้าง Visualizations ซึ่งมีโครงสร้างเบื้องต้นดังนี้

Syntax:

```
ggplot(data=my_table) +
  geom_xxx(mapping=aes(x=col1,y=col2,...)) +
  facet_wrap(~col3) +
  labs(title='Plot Name') +
  annotate('text',x=2,y=3,label='Google!')
```

Notes:

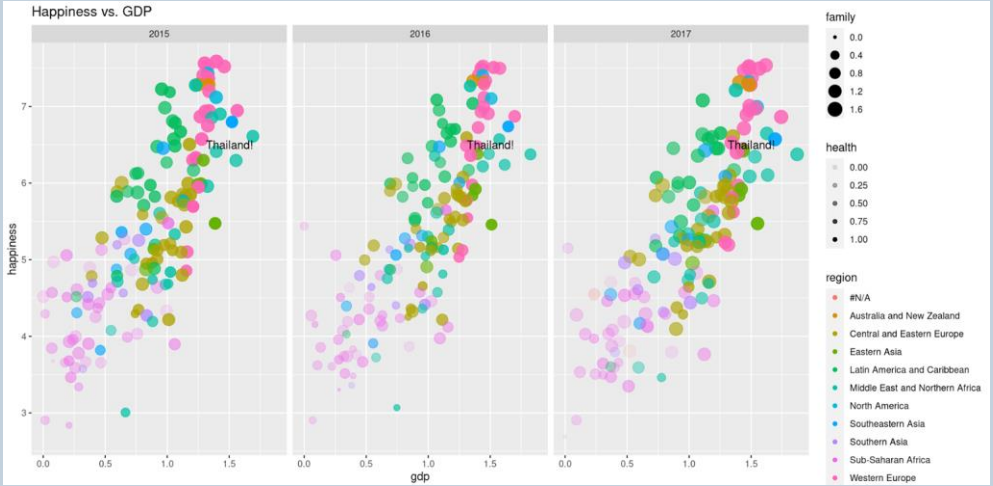
- geom = geometry คือ ชนิดเรขาคณิตของพล็อต เช่น geom_point, geom_line, geom_bar, geom_boxplot
- aes = aesthetics คือ "หน้าตา" ของพล็อต เช่น ตำแหน่ง (ค่า x และ y) สี (color) ความทึบ (alpha) ขนาด (size) รูปร่าง (shape)
- facet คือ แบ่งข้อมูลเป็นพล็อตย่อย (subplots) มีแบบ facet_wrap กับ facet_grid

Samart Skills
A Grow with Google PROGRAM

Skooldio

Ex. สร้าง Scatter Plot หาความสัมพันธ์ระหว่าง GDP และ Happiness Score ของประเทศทั่วโลก แยกปี โดยให้สีของจุดแทนภูมิภาคของประเทศ ขนาดจุดแทนขนาดครอบครัว และความทึบของสีแทนดัชนีสุขภาพ

```
ggplot(data=happy_table) +
  geom_point(mapping=aes(x=gdp,y=happiness,color=region,alpha=health,size=family)) +
  facet_wrap(~year) +
  labs(title='Happiness vs. GDP') +
  annotate("text",x=1.5,y=6.5,label="Thailand!")
```



Example: Basic Scatter Plots

สร้างด้วย geom_point()

Syntax:

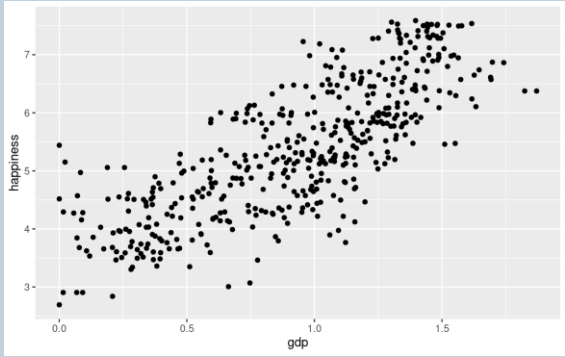
```
ggplot(data=my_table) +
  geom_point(mapping=aes(x=col1,y=col2,...))
```

Notes:

- การเพิ่ม "Layer" ของพล็อต เราใช้เครื่องหมาย + ไม่ใช่ %>%
- ถ้าจุดมันทับกัน อาจใช้ geom_jitter() แทนเพื่อให้โปรแกรม "แอบเนียน" สุ่มขยับจุดให้เล็กน้อย เพื่อให้จุดไม่ทับกัน

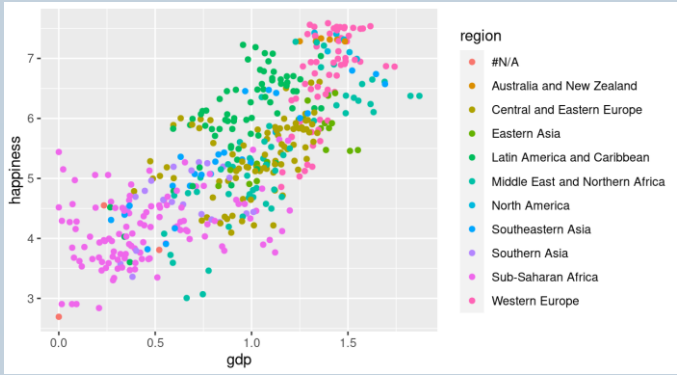
Ex. สร้าง Scatter Plot หาความสัมพันธ์ระหว่าง GDP และ Happiness Score

```
ggplot(data=happy_table) +
  geom_point(mapping=aes(x=gdp,y=happiness))
```



Ex. สร้าง Scatter Plot หาความสัมพันธ์ระหว่าง GDP และ Happiness Score ใช้สีแทนภูมิภาค และใช้ geom_jitter() แทนเพื่อให้จุดไม่ทับกันและ

```
ggplot(data=happy_table) +
  geom_jitter(mapping=aes(x=gdp,y=happiness,color=region))
```



Example: Scatter Plots with Trend Line

สร้างด้วย geom_point() และ geom_smooth()

Syntax:

```
ggplot(data=table) +
  geom_point(mapping=aes(x=c1,y=c2,...)) +
  geom_smooth(mapping=aes(x=c1,y=c2,...))
```

```
ggplot(data=table,mapping=aes(x=c1,y=c2))+
  geom_point(mapping=aes(color=...))+
  geom_smooth(method='lm')
```

Notes:

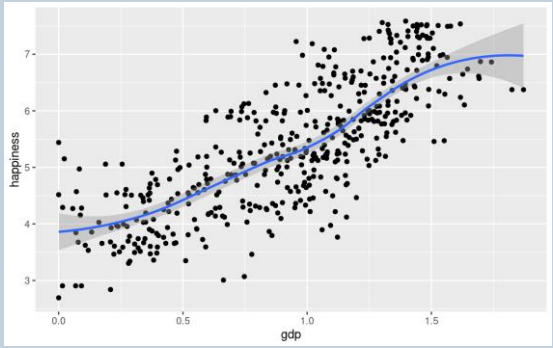
- หากต้องการเส้นตรง ต้องระบุ method='lm' (Linear Model)
- เอา mapping=aes(...) ไปใส่ใน ggplot() ก็ได้ถ้าใช้ค่าร่วมกันทุก Layer

Samart Skills
A Grow with Google PROGRAM

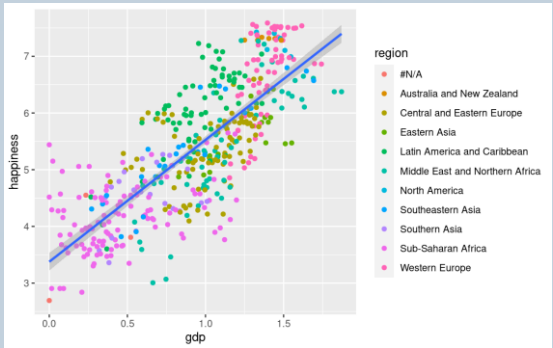
Skooldio

Ex. สร้าง Scatter Plot และเส้น Trend Line แสดงความสัมพันธ์ระหว่าง GDP และ Happiness Score

```
ggplot(data=happy_table) +
  geom_point(mapping=aes(x=gdp,y=happiness)) +
  geom_smooth(mapping=aes(x=gdp,y=happiness))
```




```
ggplot(data=happy_table,mapping=aes(x=gdp,y=happiness)) +
  geom_point(mapping=aes(color=region)) +
  geom_smooth(method='lm')
```



Example: Basic Bar Charts

สร้างแผนภูมิแท่งความถี่ด้วย geom_bar()

Syntax:

```
ggplot(data=my_table) +
  geom_bar(mapping=aes(x=col))
```

Notes:

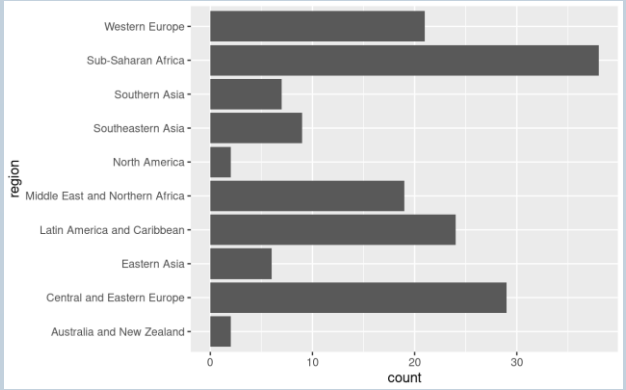
- ถ้าต้องการแท่งแนวนอน เปลี่ยนเป็น aes(y=col) แทน
- [Bonus] ถ้าต้องการเรียงค่าให้ใช้ฟังก์ชัน fct_infreq() ช่วย ซึ่งเป็นของ library ชื่อ forcats (มีอยู่ใน tidyverse แล้ว)

Samart Skills
A Grow with Google PROGRAM

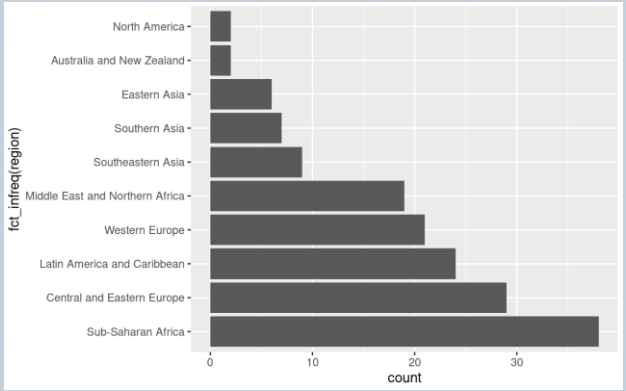
Skooldio

Ex. สร้าง Bar chart แสดงจำนวนประเทศที่เข้าร่วมในปี 2016 แบ่งตามภูมิภาค

```
happy_table %>%
  filter(year==2016) %>%
  ggplot() +
    geom_bar(mapping=aes(y=region))
```




```
happy_table %>%
  filter(year==2016) %>%
  ggplot() +
    geom_bar(mapping=aes(y=fct_infreq(region)))
```



Example: Facet Functions

สร้างกราฟย่อย (Subplots) ด้วย facet_wrap() หรือ facet_grid()

Syntax:

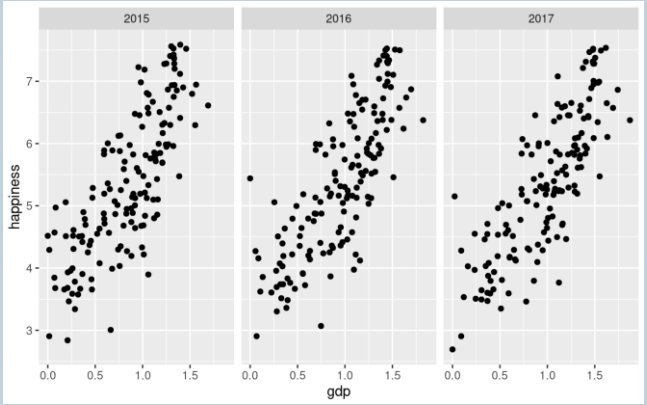
```
ggplot(data=table) +
  geom_xxx(mapping=aes(x=c1,y=c2,...)) +
  facet_wrap(~c3)
```

```
ggplot(data=table) +
  geom_xxx(mapping=aes(x=c1,y=c2,...)) +
  facet_grid(c3~c4...)
```

Note: หากต้องการใช้ facet_wrap แต่แบ่งย่อยมากกว่า 1 ตัวแปร ก็เติม ~col เพิ่มได้ เช่น facet_wrap(~year~region)

Ex. สร้าง Scatter Plot แสดงความสัมพันธ์ระหว่าง GDP และ Happiness Score แยกตามปี

```
1 ggplot(data=happy_table) +
2   geom_point(mapping=aes(x=gdp,y=happiness)) +
3   facet_wrap(~year)
```



Samart Skills
A Grow with Google PROGRAM

Skooldio

Example: Facet Functions

สร้างกราฟย่อย (Subplots) ด้วย facet_wrap() หรือ facet_grid()

Syntax:

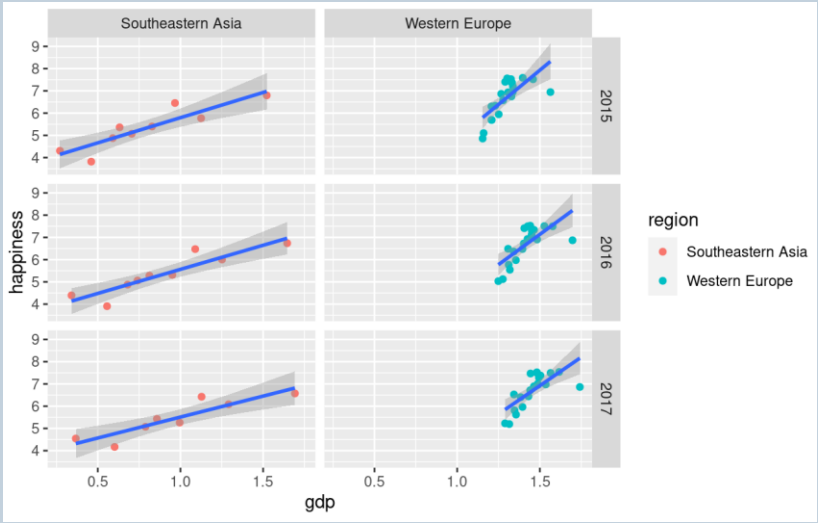
```
ggplot(data=table) +
  geom_xxx(mapping=aes(x=c1,y=c2,...)) +
  facet_wrap(~c3)
```

```
ggplot(data=table) +
  geom_xxx(mapping=aes(x=c1,y=c2,...)) +
  facet_grid(c3~c4...)
```

Note: หากต้องการใช้ facet_wrap แต่แบ่งย่อยมากกว่า 1 ตัวแปร ก็เติม ~col เพิ่มได้ เช่น facet_wrap(~year~region)

Ex. สร้าง Scatter Plot แสดงความสัมพันธ์ระหว่าง GDP และ Happiness Score แยกตามปี

```
happy_table %>%
  filter(region %in% c('Southeastern Asia','Western Europe')) %>%
  ggplot(mapping=aes(x=gdp,y=happiness)) +
    geom_point(mapping=aes(color=region)) +
    geom_smooth(method='lm') +
    facet_grid(year~region)
```



Samart Skills
A Grow with Google PROGRAM

Skooldio

Example: Labels and Annotations

เราใส่ Labels และ Annotations ได้ด้วยฟังก์ชัน labs() และ annotate() ตามลำดับ

Syntax:

```
ggplot(data=table) +
  geom_xxx(mapping=aes(x=c1,y=c2,...)) +
  labs(title='aa',subtitle='bb',caption='cc') +
  annotate('text',x=2,y=3,label='Google!')
```

Notes:

- เราสามารถปรับแต่งขนาด (size) สี (color) และรูปแบบฟอนต์ (font style) ของตัวหนังสือได้ด้วยการใส่ arguments เพิ่มเติม ลองพิมพ์ ?annotate ใน console ของ R Studio แล้วกด Enter ดูได้ครับ
- หากใครสนใจวิธีสำหรับ Annotate ชื่อประเทศของแต่ละจุดแบบครบทุกประเทศ ลองศึกษาฟังก์ชัน geom_text() ดูได้ครับ (พิมพ์ ?geom_text ใน console)

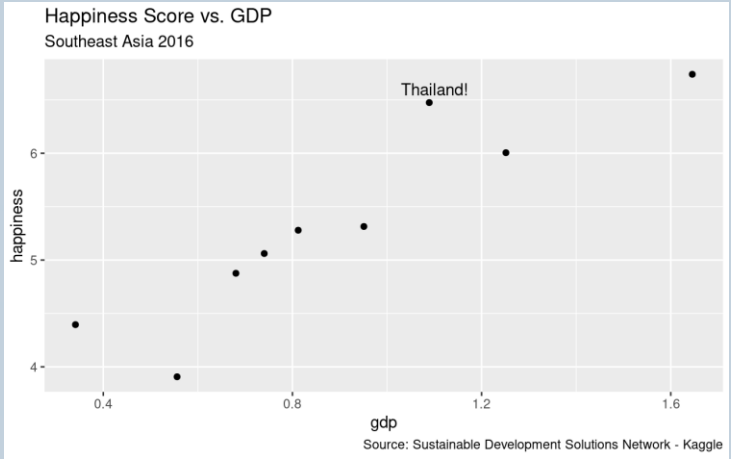
Samart Skills
A Grow with Google PROGRAM

Skooldio

Ex. สร้าง Scatter Plot แสดงความสัมพันธ์ระหว่าง GDP และ Happiness Score เฉพาะประเทศใน Southeastern Asia ในปี 2016 และแสดงว่าจุดไหนคือประเทศไทย

```
# A tibble: 9 × 11
  country     region               year happiness_rank happiness   gdp
  <chr>       <chr>               <dbl>          <dbl>     <dbl> <dbl>
1 Singapore   Southeastern Asia    2016             22      6.74 1.65
2 Thailand    Southeastern Asia    2016             33      6.47 1.09
3 Malaysia    Southeastern Asia    2016             47      6.00 1.25
4 Indonesia   Southeastern Asia    2016             79      5.31 0.951
5 Philippines Southeastern Asia    2016             82      5.28 0.812
6 Vietnam     Southeastern Asia    2016             96      5.06 0.740
7 Laos        Southeastern Asia    2016            102      4.88 0.680
8 Myanmar     Southeastern Asia    2016            119      4.39 0.341
9 Cambodia    Southeastern Asia    2016            140      3.91 0.556
```

```
happy_table %>%
  filter(year==2016 & region=='Southeastern Asia') %>%
  ggplot(mapping=aes(x=gdp,y=happiness)) +
  geom_point() +
  labs(title='Happiness Score vs. GDP', subtitle='Southeast Asia 2016',
       caption='Source: Sustainable Development Solutions Network - Kaggle') +
  annotate('text',x=1.10,y=6.6,label='Thailand!')
```



Saving Your Visualizations

บันทึก Plots ได้ 2 วิธีคือ
1. คลิก Export -> Save as Image
2. ใช้ฟังก์ชัน ggsave()

Syntax:

```
ggsave('filename.png',...)
```

Notes:

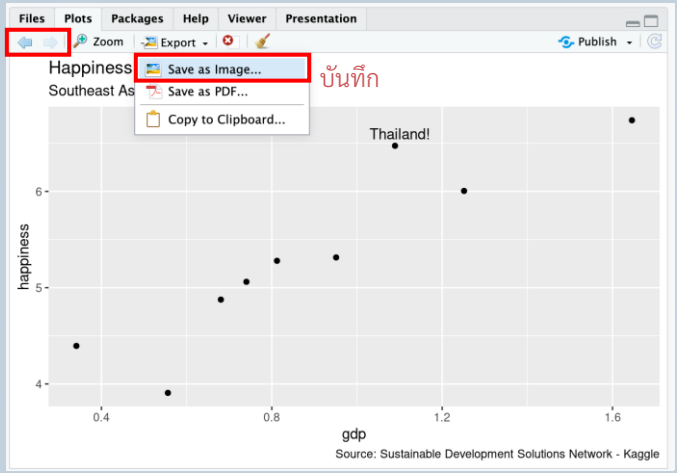
- หากต้องการบันทึก Plot ในอดีตด้วย ggsave() ต้องบันทึก Plot ในอดีตใส่ตัวแปรก่อน เช่น p <- ggplot() + ... แล้วบันทึกด้วย ggsave('filename.png', plot=p)
- หากต้องการระบุสมบัติอื่นๆ ของรูป เช่น ความกว้าง (width) ความสูง (height) ความละเอียด (dpi) สามารถใส่ arguments เพิ่มได้เลยครับ ศึกษาเพิ่มเติม พิมพ์ ?ggsave ใน console ของ R Studio แล้วกด Enter

Samart Skills
A Grow with Google PROGRAM

Skooldio

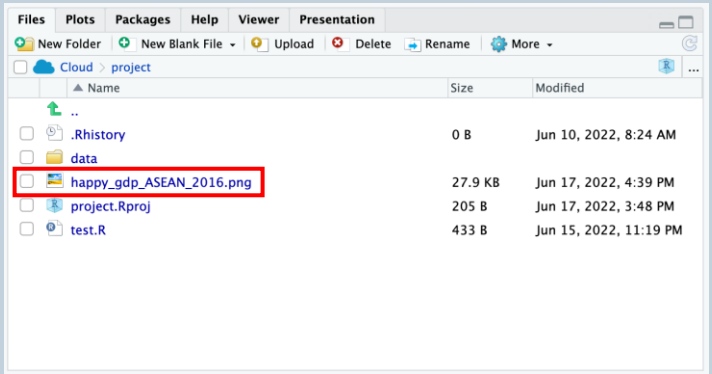
Ex. บันทึก Plot ล่าสุด ด้วยวิธี Export -> Save as Image

เลื่อนหา Plot ที่ต้องการ



Ex. บันทึก Plot ล่าสุด ด้วยวิธี ggsave()

```
1 ggsave('happy_gdp_ASEAN_2016.png')
```



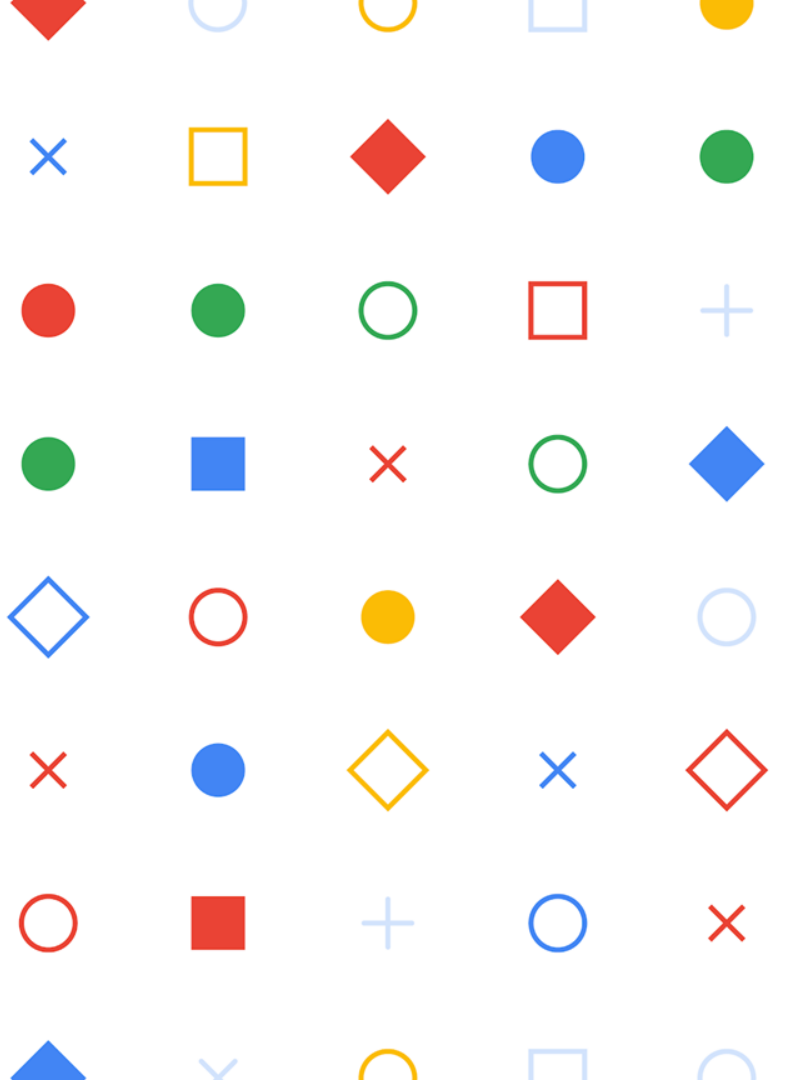
Documentation and Reports

Part5:

1. Intro to R Markdown (.Rmd file)
2. R Markdown: Structure
3. R Markdown: Syntax
4. R Markdown: Code Chunks

Samart Skills
A Grow with Google PROGRAM

Skooldio



Intro to R Markdown

Markdown คือไวยกรณ์ (Syntax) ที่ใช้ในการปรับแต่งไฟล์เอกสารข้อความ

```
---
title: "Thailand is Happy"
author: "Great"
date: '2022-06-17'
output: html_document
---

## Import Libraries
We need tidyverse and janitor libraries for this project.
```{r Import Libraries, message=FALSE}
library(tidyverse)
library(janitor)
```

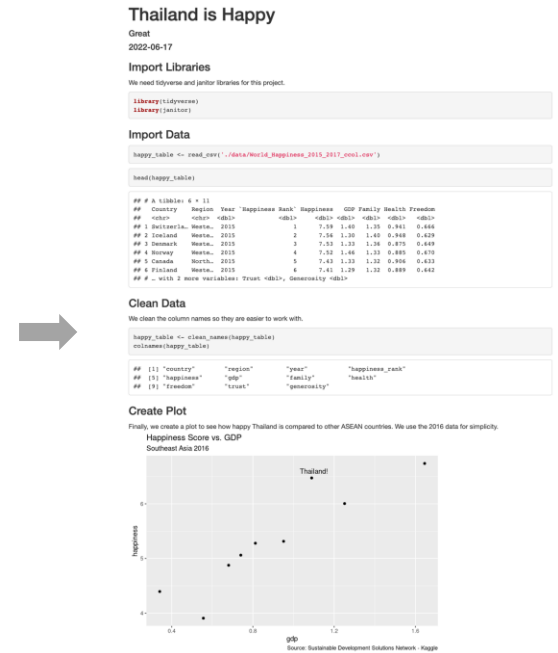
Import Data

```{r Import World Happiness Dataset, message=FALSE}
happy_table <- read_csv('./data/World_Happiness_2015_2017_ccol.csv')
```

```{r Head of Table}
head(happy_table)
```

Clean Data
We clean the column names so they are easier to work with.
```{r Clean Data}
happy_table <- clean_names(happy_table)
colnames(happy_table)
```

Create Plot
Finally, we create a plot to see how happy Thailand is compared to other ASEAN countries.
We use the 2016 data for simplicity.
```{r, echo=FALSE}
happy_table %>%
  filter(year==2016 & region=='Southeastern Asia') %>%
  ggplot(mapping=aes(x=gdp,y=happiness)) +
  geom_point() +
  labs(title='Happiness Score vs. GDP', subtitle='Southeast Asia 2016',
       caption='Source: Sustainable Development Solutions Network - Kaggle') +
  annotate('text',x=1.10,y=6.6,label='Thailand!')
```
```



Thailand is Happy

Great

2022-06-17

Import Libraries

We need tidyverse and janitor libraries for this project.

```
library(tidyverse)
library(janitor)
```

Import Data

```
happy_table <- read_csv('./data/World_Happiness_2015_2017_ccol.csv')
```

```
head(happy_table)
```

```
## # A tibble: 6 × 11
##   Country    Region  Year `Happiness Rank` Happiness   GDP Family Health Freedom
##   <chr>      <chr>  <dbl>            <dbl>     <dbl> <dbl>  <dbl>  <dbl>   <dbl>
## 1 Switzerla… Weste… 2015                1      7.59  1.40   1.35  0.941   0.666
## 2 Iceland    Weste… 2015                2      7.56  1.30   1.40  0.948   0.629
## 3 Denmark    Weste… 2015                3      7.53  1.33   1.36  0.875   0.649
## 4 Norway     Weste… 2015                4      7.52  1.46   1.33  0.885   0.670
## 5 Canada     North… 2015                5      7.43  1.33   1.32  0.906   0.633
## 6 Finland    Weste… 2015                6      7.41  1.29   1.32  0.889   0.642
## # … with 2 more variables: Trust <dbl>, Generosity <dbl>
```

Clean Data

We clean the column names so they are easier to work with.

```
happy_table <- clean_names(happy_table)
colnames(happy_table)
```

```
##  [1] "country"      "region"       "year"         "happiness_rank"
##  [5] "happiness"    "gdp"          "family"       "health"
##  [9] "freedom"      "trust"        "generosity"
```

Create Plot

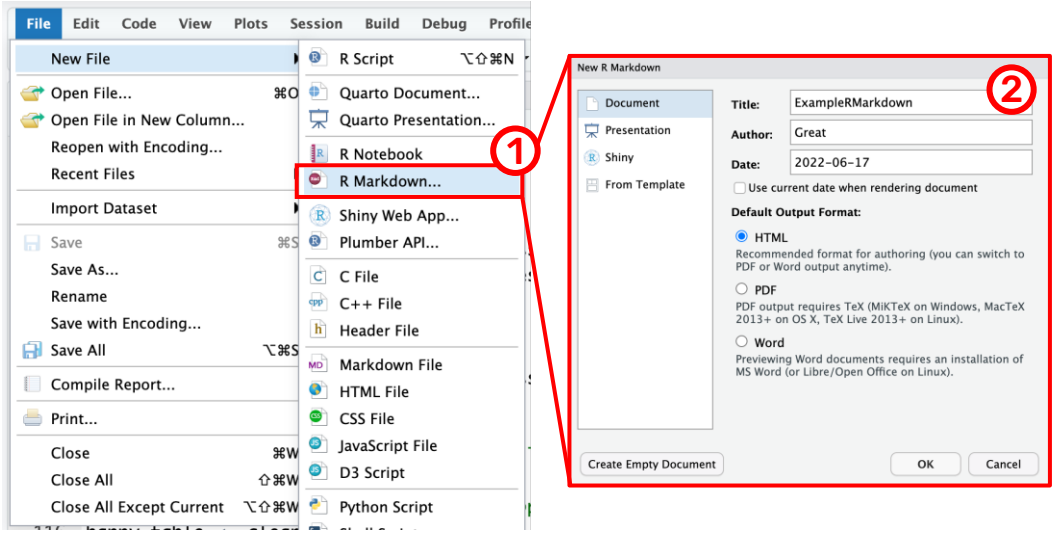
Finally, we create a plot to see how happy Thailand is compared to other ASEAN countries. We use the 2016 data for simplicity.

Samart Skills
A Grow with Google PROGRAM

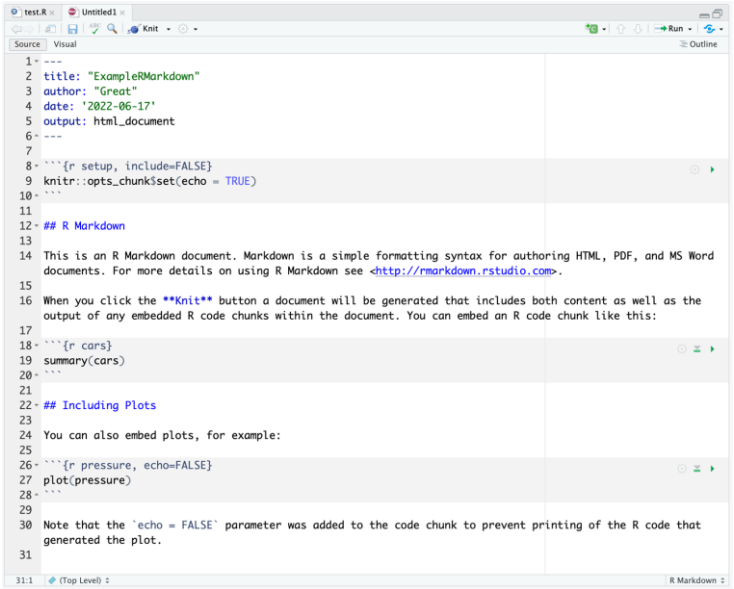
Skooldio

Intro to R Markdown

ใน R Studio เราสร้างไฟล์ R Markdown (.Rmd) ได้ง่าย ๆ ดังนี้



Result

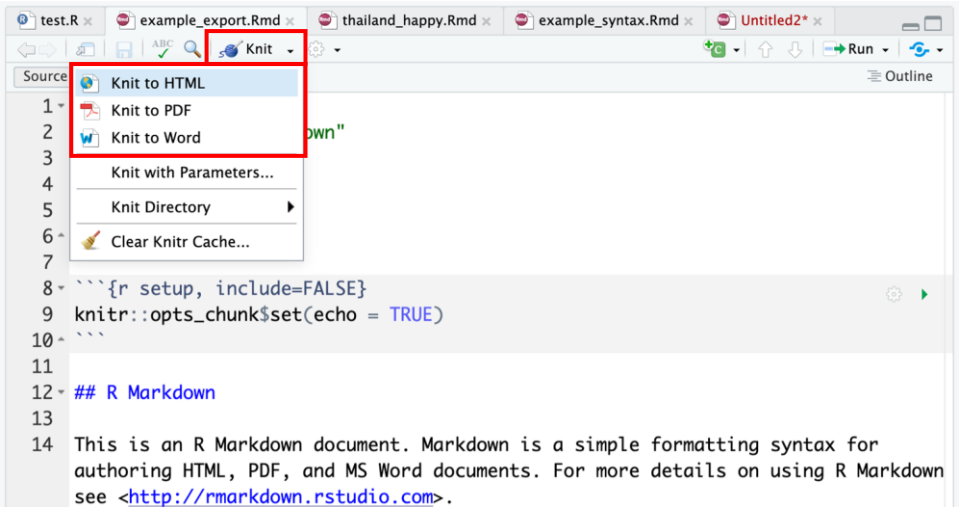


Samart Skills
A Grow with Google PROGRAM

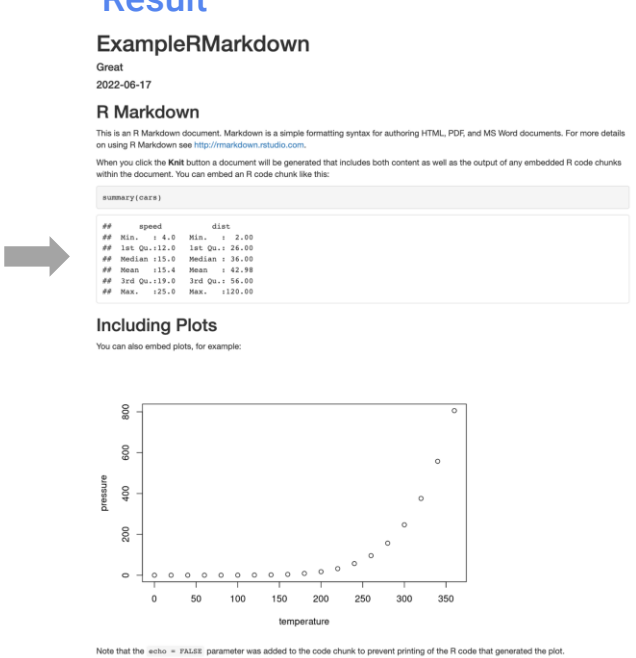
Skooldio

Intro to R Markdown

สามารถ Export เป็นไฟล์สกุลต่างๆ เช่น .html .pdf .docx เป็นต้นด้วยปุ่ม Knit



Result

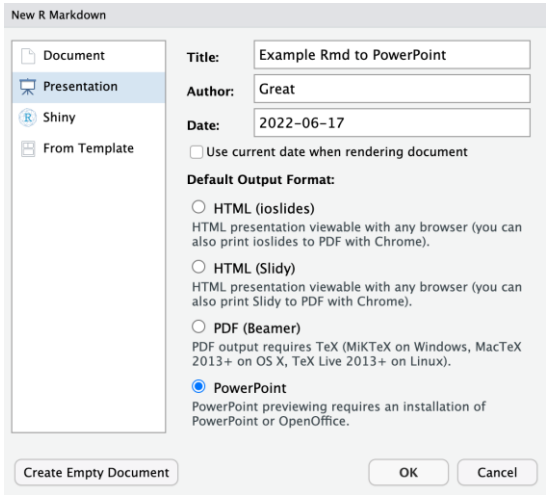


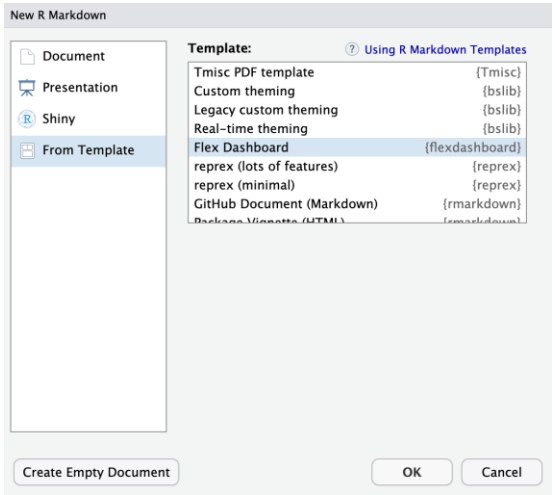
Samart Skills
A Grow with Google PROGRAM

Skooldio

Intro to R Markdown

เรายังสามารถ Export ไฟล์ .Rmd เป็นไฟล์ Presentations เช่น PowerPoint หรือแม้แต่เป็น Dashboard ได้ด้วย!



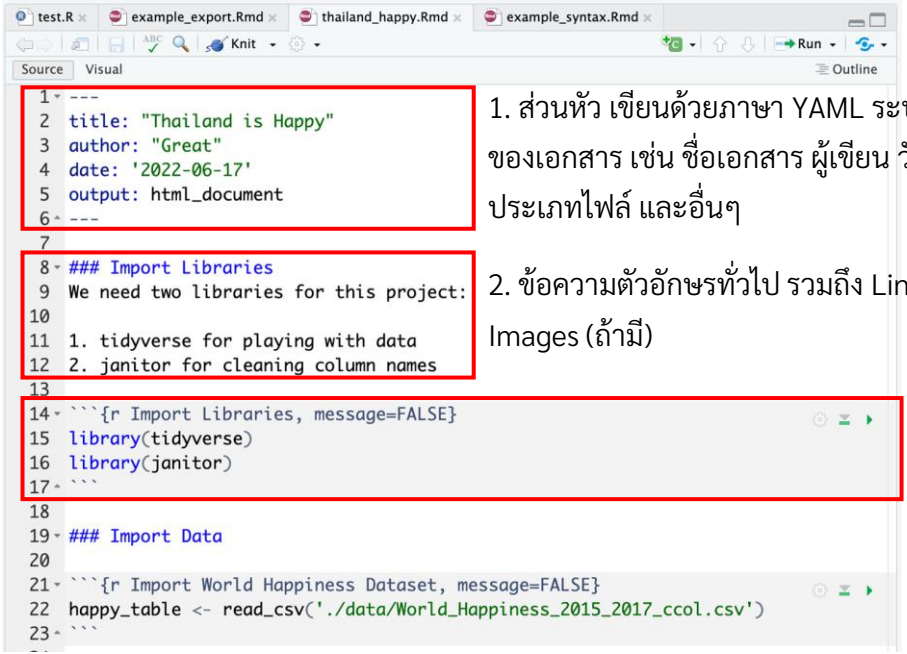


สำหรับ Flex Dashboard ต้องลง library ชื่อ "flexdashboard" ก่อนถึงจะมีให้เลือกใช้ครับ ดูตัวอย่างผลลัพธ์ได้ที่ https://rmarkdown.rstudio.com/gallery.html

Trick: เราสามารถใช้ Template หรือโครงร่างของเอกสารที่มีอยู่ก่อนแล้วได้ เพื่อประหยัดเวลาและเพิ่มความคล่องตัวในการทำงาน (Streamline the work process) เช่น Templates สำหรับรายงานยอดขายรายเดือน ใครสนใจสร้าง Template เองลองศึกษาเพิ่มเติมได้ที่ https://bookdown.org/yihui/rmarkdown/document-templates.html ครับ

R Markdown: Structure

โครงสร้างเบื้องต้นของไฟล์ R Markdown มีดังนี้



1. ส่วนหัว เขียนด้วยภาษา YAML ระบุ Metadata ของเอกสาร เช่น ชื่อเอกสาร ผู้เขียน วันที่ผลิต ประเภทไฟล์ และอื่นๆ

2. ข้อความตัวอักษรทั่วไป รวมถึง Link และ Images (ถ้ามี)

3. ส่วนของโค้ดที่เราต้องการ Run ใส่เป็นช่วง ๆ ในเอกสาร เรียกว่า Inline Code Chunks (Inline ในที่นี้แปลว่า ใส่อยู่ท่ามกลางส่วนอื่นๆ ของเอกสารได้เลย)

R Markdown: Syntax

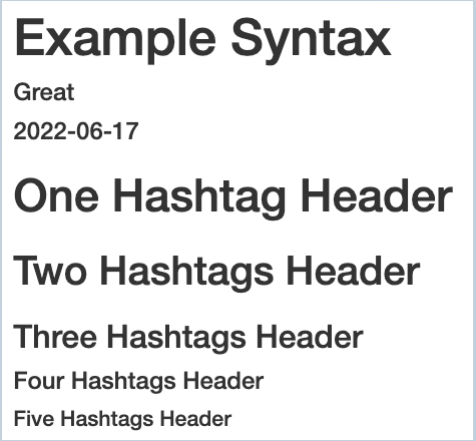
เราสร้างชื่อหัวข้อ (Headers) ขนาดต่างๆ ได้ ด้วยเครื่องหมาย Hashtags #

Syntax:

```
---
title: "Example Syntax"
author: "Great"
date: '2022-06-17'
output: html_document
---

# One Hashtag Header
## Two Hashtags Header
### Three Hashtags Header
#### Four Hashtags Header
##### Five Hashtags Header
```

Result



Note: สังเกตว่า ยิ่งจำนวน # มาก ตัวหนังสือยิ่งเล็กลง

R Markdown: Syntax

เราปรับแต่งรูปแบบตัวหนังสือให้เป็นตัวหนา ตัวเอียง ได้ด้วยวิธีดังนี้

Syntax:

```
16 #### Example Styles
17
18 Normal Text
19
20 **Bold Text**
21
22 *Italic Text* or _Italic Text_
```

Result

Example Styles

Normal Text

Bold Text

Italic Text or *Italic Text*

R Markdown: Syntax

เราสร้าง Lists แบบ Bullet Points หรือแบบเป็นตัวเลขได้ดังนี้

Notes:

- เราเรียกเครื่องหมายดอกจัน * ภาษาอังกฤษว่า Asterisk
- สังเกตว่า ต้องมีการเว้นบรรทัด ก่อนจะขึ้นส่วนที่เป็น Bullet Points หรือ Numbered List (ดูการเว้นบรรทัดที่ 27 และ 32) ซึ่งถ้าไม่เว้นผลลัพธ์จะไม่ออกมาเป็นแบบที่ต้องการ

Syntax:

```
24 #### Example Lists
25
26 Bullet Points
27
28 * Point 1
29 * Point 2
30
31 Numbered List
32
33 1. Item 1
34 2. Item 2
```

Result

Example Lists

Bullet Points

- Point 1
- Point 2

Numbered List

1. Item 1
2. Item 2

R Markdown: Syntax

เราสร้าง Link ได้ 2 แบบ ดังนี้

Syntax:

```
36 #### Links
37
38 <https://www.skooldio.com/>
39
40 [Skooldio is Awesome!](https://www.skooldio.com/)
```

Result

Links

https://www.skooldio.com/

Skooldio is Awesome!

R Markdown: Syntax

เราแสดงรูปภาพจาก Link ไปยังรูปภาพได้ดังนี้

Syntax:

```
42 #### Image
43
44 ![Skooldio Logo](https://www.skooldio.com/static/images/Skooldio_Logo.svg)
```

Result

Image

Skooldio Logo

R Markdown: Code Chunks

เราสร้าง Code Chunks ได้ดังนี้

Notes:

- Code Chunk 1 ก้อน จะเริ่มต้นและจบลงด้วยเครื่องหมาย backtick 3 อัน ```
    - เครื่องหมายลักษณะนี้เรียกว่า "Delimiter" หรือ "ตัวคั่น" สำหรับกั้นข้อมูล (Data items) เป็นส่วน ๆ)
    - เครื่องหมายนี้อยู่ตรงมุมซ้ายบนของคีย์บอร์ดด้านล่างปุ่ม Esc
- ตัว r ด้านใน { ... } บอกโปรแกรมว่าภาษาที่ใช้คือภาษา R
- My Chunk No. 1 ด้านใน { ... } เอาไว้ใช้บอกชื่อของ Chunk เฉย ๆ เพื่อความสะดวกในการค้นหา โดยจะไม่โผล่ในผลลัพธ์ที่ Export

Samart Skills
A Grow with Google PROGRAM

Skooldio

Syntax:

````
#### Inline Code Chunk

```{r My Chunk No.1}
summary(cars)
```
````

Result

Inline Code Chunk

```
summary(cars)
```

```
##      speed           dist       
##  Min.   : 4.0   Min.   :  2.00  
##  1st Qu.:12.0   1st Qu.: 26.00  
##  Median :15.0   Median : 36.00  
##  Mean   :15.4   Mean   : 42.98  
##  3rd Qu.:19.0   3rd Qu.: 56.00  
##  Max.   :25.0   Max.   :120.00
```

R Markdown: Code Chunks

เราสามารถ Run โค้ดได้โดยตรงระหว่างกำลังเขียนไฟล์ .Rmd ได้ โดยการกดปุ่ม Play

Syntax:

````
#### Inline Code Chunk

```{r My Chunk No.1}
summary(cars)
```
````

click ตรงนี้

Result

````
#### Inline Code Chunk

```{r My Chunk No.1}
summary(cars)
```
````

```
     speed           dist
 Min.   : 4.0   Min.   :  2.00
 1st Qu.:12.0   1st Qu.: 26.00
 Median :15.0   Median : 36.00
 Mean   :15.4   Mean   : 42.98
 3rd Qu.:19.0   3rd Qu.: 56.00
 Max.   :25.0   Max.   :120.00
```

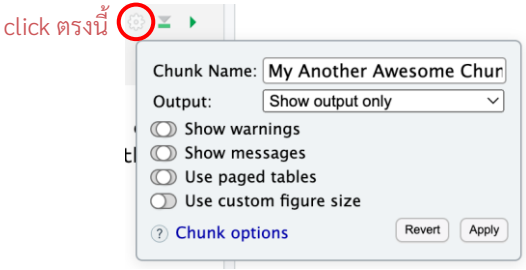
Samart Skills
A Grow with Google PROGRAM

Skooldio

R Markdown: Code Chunks

เราสามารถซ่อน Code แล้วโชว์แค่ผลลัพธ์ได้ ด้วยการใส่พารามิเตอร์ echo=FALSE

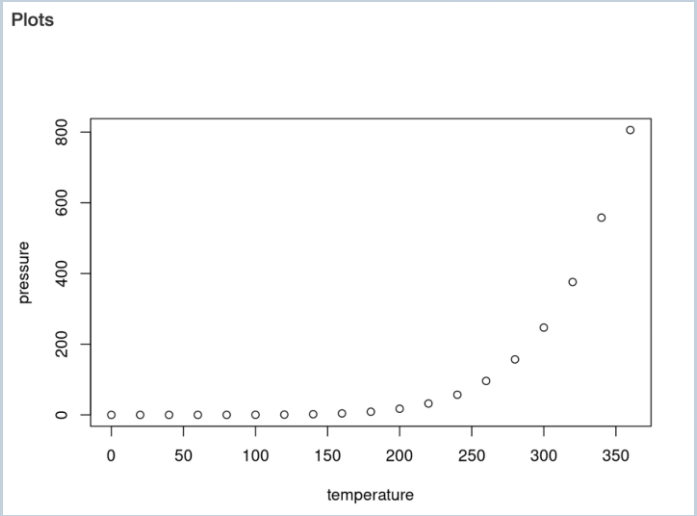
Note: หากกดปุ่มเกียร์ตรงมุมขวาบน จะมี Options การแสดงผลให้เลือกอีกมากมาย



Syntax:

````
52 #### Plots
53 ```{r My Another Awesome Chunk, echo=FALSE}
54 plot(pressure)
55 ```
56
````

Result



Samart Skills
A Grow with Google PROGRAM

Skooldio