ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC & KỸ THUẬT MÁY TÍNH

CẤU TRÚC RỜI RẠC CHO KHMT (CO1007)

R cho bài tập lớn

GVHD: Huỳnh Tường Nguyên
Trần Tuấn Anh
Nguyễn Ngọc Lễ

Tp. Hồ Chí Minh, Tháng 05/2020

Mục lục

1 Giới thiệu ngôn ngữ lập trình R

Trong mùa dịch Covid-19, trường Đại học Bách Khoa, DDHQF-HCM đã triển khai giảng dạy trực tuyến và yêu cầu sinh viên thực hiện các bài tập nhỏ để thu nhận phản hồi về việc học tập và hiểu biết của các bạn thông qua các tài nguyên online được cung cấp.

Phân tích & thống kê dữ liệu qua các lần nộp bài của sinh viên không những giúp giáo viên có những hướng đúng trong việc phát hiện ra những kiến thức mà sinh viên chưa chắc chắn, cũng như có hướng để cải thiện bổ sung phần học liệu trong tương lai để phù hợp với hơn người học.

1.1 Xử lý cơ bản

Khai phá dữ liệu từ hệ thống nộp bài online có ý nghĩa quan trọng trong việc đánh giá chất lượng của sinh viên. Ngoài ra, những đánh giá kết quả nộp bài của từng sinh viên, hay từng bài tập sẽ góp phần xác định những điểm mạnh, điểm yếu của sinh viên để giáo viên có phương pháp phù hợp trong việc cải thiện kỹ năng của sinh viên.

Trong bài tập lớn này, các sinh viên sẽ bắt đầu với các bài toán thống kê đơn giản từ những dữ liệu được cung cấp. Qua đó, các em sẽ tìm ra những con số thú vị, có ý nghĩa đối với các dữ liệu thực tế trong quá khứ của hệ thống chấm bài online. Những kết quả mà các em tìm ra sẽ là bước khởi đầu cho việc khai phá nguồn dữ liệu của hệ thống sau này, nhằm đạt tới mục tiêu nâng cao kỹ năng lập trình, kỹ năng giải quyết vấn đề cho người học cũng như hướng tới mục tiêu cao hơn khi tích hợp với các hệ thống quản lý và cải thiện chất lượng dạy và học.

1.2 Vector

1.2.1 Dữ liệu số

Class	*Exams*	*Homework*	*Projects*
Math	92	87	85
Chemistry	90	81	92
Writing	84	95	79
Art	95	86	93
Music	92	90	91
Physical Education	85	88	95

Ta có điểm cuối cùng cho mỗi lớp và gán chúng cho các biến:

```
math <- 88
chemistry <- 87.66667
writing <- 86
art <- 91.33333
history <- 84
music <- 91
physical_education <- 89.33333
```

Tạo một vector điểm lớp học cuối cùng.

```
final_scores <- c(math, chemistry, writing, art, history, music, physical_education)
```

Hiển thị kiểu dữ liệu của **final_scores**

Source: typeof(final_scores)

Output: "double"

Một số thao tác thêm cho vector trong phân tích dữ liệu

- Một tập con các giá trị trong vectors
- Gán tên đến một thành phần trong vector
- Dùng các toán tử để trả lời các câu hỏi về dữ liệu trong được chứa trong vector

Với một vector, mỗi thành phần có một **vị trí**. Trong ngôn ngữ lập trình R, chỉ số được bắt đầu từ 1 (**1-indexed**), nghĩa là thành phần đầu tiên trong vector được gán 1 vị trí là một.

Position	1	2	3	4	5	6	7
Final_scores	88	87.6667	86	91.33333	84	91	89.3333

Trích các giá trị từ vector bằng cách chỉ ra vị trí của chúng trong [ ]
Source: final_scores[3]
Output: 86
Trích nhiều thành phần bằng cách dùng c()
Source: final_scores[c(1,3,7)]
Output: 88.00000 86.00000 89.33333
Chọn ra một dãy các thành phần kế nhau bằng cách dùng dấu 2 chấm :
Source: final_scores[1:4]
Output: 88.00000 87.66667 86.00000 91.33333

Practices

Điểm trong lớp STEM gồm có (science, technology, engineering, and math), còn lài là lớp non-STEM.

1. Viết code tao một vector mới, **stem_grades** chỉ chứa điểm toán và hóa
2. Tao một vector mới, **stem_grades** chứa điểm cuối cùng trong lớp khác lớp STEM
3. Tính giá tri trung bình mỗi vector và lưu giá trị vào các biến **avg_stem_grades**, **avg_non_stem_grades**

1.2.2 Dữ liệu ký tự

Tạo một vector chứa tên của 2 lớp bằng cách dùng dấu nháy đơn hoặc đôi (" or "")
math_chemistry <- c("math", "chemistry")
Source: typeof(math_chemistry)
Output: "character"

Trong R, ta có thể gán thêm thuộc tính đến một vector. Thuộc tính sẽ cung cấp thêm thông tin, như tên, về các giá trị trong một vector. Để gán tên đến các thành phần của vector ta dùng hàm name()

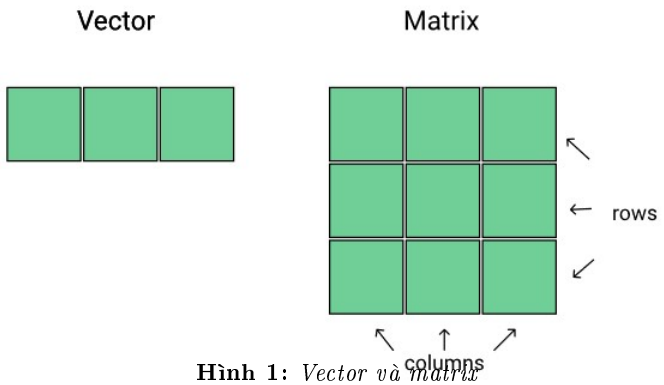
Source:
```
math_chemistry <- c(88, 87.66667)
class_names <- c("math", "chemistry")
names(math_chemistry) <- class_names
print(math_chemistry)\\
```

Output:

math	chemistry
88.00000	87.66667

1.3 Ma trận - Matrices

Trong phần này ta học thêm về cấu trúc dữ liệu về ma trận trong R. Như vector, ma trận giữ dữ liệu kiểu số hoặc kiểu luận lý. Một ma trận bao gồm nhiều dòng và nhiều cột, ma trận thể hiện một không gian 2 chiều, còn vector là một chiều.



Hình 1: *Vector và matrix*

Ta sẽ học dùng ma trận để chứa thông tin về xếp hạng các trường đại học.

Nguồn dữ liệu đến từ tập dữ liệu xếp hạng đại học thới giới. 100 trường hàng đầu trên thới giới được xếp theo chuẩn. Đây là vài dòng dữ liệu mẫu.

University	world_rank	quality_of_education	influence	broad_impact	patents
Harvard	1	1	1	1	3
Stanford	2	9	3	4	10
Stanford	2	9	3	4	10
MIT	3	3	2	2	1
Cambridge	4	2	6	13	48
Oxford	5	7	12	9	15
Oxford	6	13	13	12	4

Đầu tiên ta dùng hàm c() để tạo các vector:
harvard <- c(1,1,1,1,3) stanford <- c(2,9,3,4,10) MIT <- c(3,3,2,2,1) cambridge <- c(4,2,6,13,48) oxford <- c(5,7,12,9,15) columbia <- c(6,13,13,12,4)

Tiếp theo ta dùng hàm rbind() để tổ hợp các vector thành một ma trận. Với "r" trong rbind() đại diện cho dòng.

Source:

```
harv_stan <- rbind(harvard, stanford)
```

Output:

	[,1]	[,2]	[,3]	[,4]	[,5]
Harvard	1	1	1	1	3
Stanford	2	9	3	4	10

Ta có thể dùng hàm rbind() để gắn 1 dòng đơn đến một ma trận đang tồn tại hoặc tổ hợp 2 ma trần bằng cách chồng 2 ma trận lên nhau.

Source:

```
rbind(harv_stan, MIT)
```

Output:

	[,1]	[,2]	[,3]	[,4]	[,5]
Harvard	1	1	1	1	3
Stanford	2	9	3	4	10
MIT	3	3	2	2	1

Chúng ta đã biết cách gắn thuộc tính tên đến vector, đối với ma trận thêm tên vào hàng hoặc cột của ma trận tương tự như đặt tên các phần tử của vector. Trong trường hợp là vector ta sử dụng hàm name(), trường hợp ma trận ta sử dụng hàm

- Rows: rownames()
- Columns: colnames()

. Đặt tên các cột cho ma trận:
Source:

```
harv_stan <- rbind(harvard, stanford)
colnames(harv_stan) <- c("world_rank", "quality_of_education","influence","broad_impact",
"patents")
```

Output:

	world_rank	quality_of_education	influence	broad_impact	patents
Harvard	1	1	1	1	3
Stanford	2	9	3	4	10

1.3.1 Chiều của ma trận

Xác định chiều của ma trận (số dòng, số cột) bằng cách sử dụng hàm dim()

- dim(harv_stan): 2 5 # 2 dòng, 5 cột
- dim(harv_stan)[1]: 2
- dim(harv_stan)[2]: 5

1.3.2 Thêm dòng, cột

- rbind(): tổ hợp các vector hoặc ma trận bằng dòng
- cbind(): tổ hợp các vector hoặc ma trận bằng cột

Source:
```
harv_stan_tuition <- c(43280, 45000)
cbind(harv_stan, harv_stan_tuition)
```
Output:

	world_rank	quality_of_education	influence	broad_impact	patents	harv_stan_tuition
harvard	1	1	1	1	3	43280
stanford	2	9	3	4	10	45000

1.3.3 Chỉ số ma trận

- Để chọn một thành phần của ma trận ta cần chỉ số dòng và chỉ số cột hoặc tên của dòng và cột trương ứng

Source:
```
harv\_stan[2, 5]
harv\_stan["stanford", "patents"]
```
Output:
```
10
10
```

- Để chọn nhiều hơn một thành phần trong ma trận ta có thể chỉ ra một khoảng dòng hoặc cột tương ứng
- dùng dấu :

Source:
```
harv_stan[2, 4:5]
```
Output:
```
broad_impact    patents
4               10
```

- Chúng ta cũng có thể lập chỉ mục ma trận để trích xuất các phần tử không nằm cạnh nhau. Bằng cách sử dụng c() để chỉ ra dòng và cột ta cần trích ra

Source:
```
harv_stan[c(1,2), c(1,3)]
harv_stan[c("harvard", "stanford"),
c("world_rank", "influence")]
```
Output:

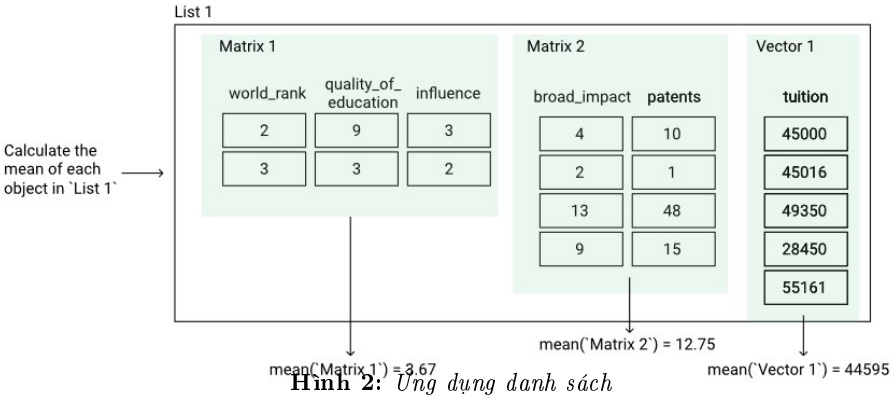
world_rank	influence
harvard 1	1
stanford 2	9

- để trích xuất toàn bộ hàng hoặc cột
  - harv_stan["harvard",]
  - harv_stan[,"quality_of_education"]

1.4 Danh sách - List

- **Vectors:** Cấu trúc dữ liệu một chiều chứa một kiểu dữ liệu.
- **Matrices:** Cấu trúc dữ liệu hai chiều chứa một kiểu dữ liệu.
- **List:** Cấu trúc dữ liệu chứa nhiều kiểu đối tượng. Các đối tượng có thể gồm các cấu trúc dữ liệu khác nhau như các phần tử dữ liệu đơn, vectors, các ma trận

Sức mạnh của danh sách: Thực hiện cùng một thao tác trên từng đối tượng trong danh sách của bạn, điều này có thể cho phép bạn tránh lặp lại việc gõ cùng một chức năng.
Tạo một danh sách: uni_clubs <- list("ballroom dancing", "rugby", "bird watching", "pottery")



Hình 2: *Ứng dụng danh sách*

```
[[1]]
[1] "ballroom dancing"

[[2]]
[1] "rugby"

[[3]]
[1] "bird watching"

[[4]]
[1] "pottery"
```

Tổ chức dữ liệu câu lạc bộ cờ trong một danh sách với các mẫu thông tin sau:

- Tên câu lạc bộ (vector kiểu dữ liệu ký tự)
- Mô tả câu lạc bộ (vector kiểu dữ liệu ký tự)
- Lệ phí tham gia (vector kiểu dữ liệu số)
- Ngày hợp (vector kiểu dữ liệu ký tự)
- Thời gian hợp (vector kiểu dữ liệu ký tự)

source

```
club_title <- c("Chess Club")
club_description <- c("Meets two nights a week for members to play chess. Snacks are provided.")
club_dues <- c(50, 20, 15)
meeting_days <- c("Monday", "Wednesday")
meeting_times <- c("6:00 pm", "8:00 pm")
club_meetings <- rbind(meeting_days,meeting_times)
chess_club <- list(club_title,club_description,club_dues,club_meetings)
club_meetings
chess_club
```

Output

```
[[1]]
[1] "Chess Club"

[[2]]
[1] "Meets two nights a week for members to play chess. Snacks are provided."

[[3]]
[1] 50 20 15

[[4]]
```

```
[,1]      [,2]
meeting_days  "Monday"  "Wednesday"
meeting_times "6:00 pm" "8:00 pm"

[,1]    [,2]
meeting_days  Monday  Wednesday
meeting_times 6:00 pm 8:00 pm
```

- Đặt tên cho đối tượng

Ta có thể gán tên cho đến các đối tượng trong danh sách là tương tự như đặt tên cho các thành phần trong một vector. Chúng ta sẽ dùng hàm names()

source

```
ballroom_dancing <- list(c("Ballroom Dancing Club"), c("Practices waltz, salsa, and tango dancing for competitions with local university dance teams"), c(150))
names(ballroom_dancing) <- c("club_title", "club_description", "club_dues")
```

Output

```
$club_title
[1] "Ballroom Dancing Club"

$club_description
[1] "Practices waltz, salsa, and tango dancing for competitions with local university dance teams"

$club_dues
[1] 150
```

Để đánh chỉ mục danh sách, thông thường ta sẽ dùng 2 cách đánh chỉ mục:

- Dấu ngoặc đơn để trả về **danh sách** các thành phần được chọn ([ ])
- Dấu ngoặc kép để trả về **một** phần tử ([[ ]])

Trích xuất đối tượng thứ hai trong danh sách dùng [] và chúng ta viết đoạn mã sau:

Source:

```
rugby_club[2]

typeof(rugby_club[2])
```

Output:

```
$club_description
[1] "Plays matches against clubs from local universities"
[1]"list"
```

Trích xuất đối tượng thứ hai trong danh sách dùng [[ ]]:

Source:

```
rugby_club[[2]]

typeof(rugby_club[[2]])
```

Output:

```
[1] "Plays matches against clubs from local universities"

[1] "character"
```

Thực hành

Một danh sách chess_club như sau:

```
$club_title
[1] "Chess Club"

$club_description
[1] "Meets two nights a week for members to play chess. Snacks are provided."

$club_dues
[1] 50 20 15

$club_meetings
```

```
[,1]       [,2]
meeting_days  "Monday"   "Wednesday"
meeting_times "6:00 pm" "8:00 pm"
```

- Đánh chỉ mục chess_club list trả lại phần tử thứ 2 của club_dues

Code chess_club[[3]][2] hoặc chess_club[[c(3,2)]]
Để thay thế các giá trị club_dues trong rugby_club rugby_club$club_dues <- c(110, 60) Để thay thế giá trị 110 lại thành 60: rugby_club[[c(3, 1)]] <- 60

- Viết mã để thay thế phần tử thứ ba của club_dues thành 5.
- Hiển thị phần tử thứ ba của club_dues trong danh sách Chess_club để kiểm tra xem bạn đã thay thế nó đúng chưa.

Code
chess_club[[c(3,3)]]<-5
chess_club[[c(3,3)]]

Thông tin dưới dạng dữ liệu số trong một ma trận có tên Member_years_rugby. Chúng ta muốn thêm ma trận này làm đối tượng vào danh sách rugby_club, hiện có ba đối tượng:

chúng ta có thể chỉ định vị trí mà chúng ta muốn phần tử mới xuất hiện: rugby_club[[4]] <- member_years_rugby

Output

```
$club_title
[1] "Rugby Club"

$club_description
[1] "Plays matches against clubs from local universities"

$club_dues
[1] 80 60

[[4]]
first_yr second_yr third_yr fourth_yr
fall          8        12       13         2
spring        6        11       14         3
```

Thực hành

- Thêm một vector first_year như một đối tượng, có cùng tên, đến cuối danh sách chess_club
  - Sử dụng cú pháp chess_club [["object_name"]] <- object để thêm vectơ với tên được gán
- Đánh chỉ mục chess_ club để hiển thị số lượng sinh viên năm thứ nhất tham gia câu lạc bộ cờ vua trong học kỳ mùa xuân năm ngoái.

Code
first_years <- c(12, 15)
names(first_years) <- c("fall", "spring")
chess_club[["first_years"]] <- first_years
chess_club$first_years[[2]]

Tổ hợp các danh sách vào một danh sách đơn

- Để tổ hợp 2 danh sách rugby_club và ballroom_dancing vào một danh sách mới ta có thể dùng hàm c()
  - uni_club_data <- c(rugby_club, ballroom_dancing)
- kết hợp các danh sách bảo tồn tổ chức của từng danh sách (vì có thể cùng tên giữa các thành phần). Ta dùng list để tạo 1 danh sách của các danh sách
  - uni_club_list <- list(rugby_club = rugby_club, ballroom_dancing = ballroom_dancing)

2 The tidyverse

2.1 Khung dữ liệu - Data frame

Data frame có thể là cấu trúc phổ biến nhất mà bạn sẽ làm việc khi phân tích dữ liệu trong R, vì vậy chúng tôi sẽ giúp bạn xây dựng sự hiểu biết cơ bản mạnh mẽ về cách thao tác sau đó. Giống như danh sách, data frame có thể chứa nhiều loại dữ liệu. Tuy nhiên, không giống như danh sách, tất cả các phần tử của khung dữ liệu là các vectơ có độ dài bằng nhau.

Khi làm việc dữ liệu với R, ta sẽ nhập dữ liệu vào R và lưu nó như 1 data frame. Thay vì dùng các hàm cơ bản của R cho nhập liệu.

Cài đặt và load gói readr

Ta sẽ dùng 1 gói readr gói này chứa các hàm cho nhập dữ liệu với nhiều định dạng khác nhau.

Gói readr là phần của tidyverse có tập các gói để cải tiến quy trình khoa học dữ liệu trong vài cách:

- **Tốc độ** nhập vai dữ liệu càng lớn các hàm readr thường nhanh hơn các hàm cơ bản trong R
- **Khả năng dùng lại** readr không phụ thuộc lên hệ điều hành và các biến môi trường.
- **Tính nhất quán** gói readr tương thích với các gói tinyverse

Để cài đặt một gói R, ta dùng hàm install.packages(): install.packages("readr"). Sau khi đã cài đặt các gói, chúng ta cần load gói mà chúng ta muốn làm việc với nó khi bắt đầu một phiên làm việc mới. Để load các gói ta dúng hàm library(): library(readr)

Thí dụ:

```
library(readr)
recent_grads <- read_csv("recent_grads.csv")
```

Cài đặt và load gói xlsx

install.packages("xlsx")
library("xlsx")

Đọc một excel file:
read.xlsx2(file, sheetIndex, header=TRUE, colClasses="character")

- **file**: đường dẫn đến file để đọc
- **sheetIndex**: một số chỉ định một sheet để đọc; sheetIndex=1 đọc sheet đầu tiên
- **header**: Nếu là TRUE, dòng đầu tiên được dùng như tên biến
- **colClasses**: một vector ký tự thể hiện lớp mỗi cột

Thí dụ:

```
library(xlsx)
recent_grads <- read.xlsx2("DATA\\test_import.xlsx", sheetIndex = 1)
head(res[, 1:6])
```

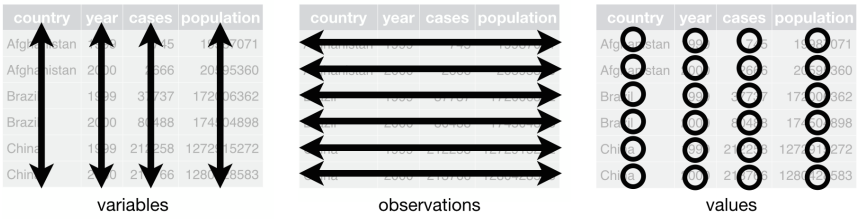
2.2 Các thao tác trên data frame

Mặc dù ta có thể đánh chỉ mục data frame để trả lại 1 tập con của các cột, ta nên dùng một công cụ trong gói tidyverse được thiết kế cụ thể cho phân tích dữ liệu trong data frame dplyr

Cài đặt và load gói:
install.packages("dplyr")
library(dplyr)
Thao tác dữ liệu liên quan đến việc đi từ dữ liệu thô đến dữ liệu được xử lý.

Thô:	Đã xử lý:
Dữ liệu gốc.	Sẵn sàng cho phân tích.
Thường khó dùng trực tiếp cho phân tích dữ liệu.	Dữ liệu sau khi đã chuyển đổi, hợp nhất,...
Ta không nên xử lý dữ liệu gốc.	Tất cả các bước đưa ta đi từ dữ liệu thô đã được xử lý nên được soạn sẵn.

Dữ liệu: Các quan sát nằm trong các hàng, các biến nằm trong các cột. dplyr::glimpse(grads)



Hình 3: *Dữ liệu*

```
Rows: 722
Columns: 16
$ Mã.s.U.1ED1..ID                          <chr> "1911478", "1914405", "1914405", "1911478", "1912
$ Tình.tr.U.1EA1.ng                        <chr> "Đã hoàn thành", "Đã hoàn thành", "Đã hoàn thành
$ Đã.b.U.1EAF.t.d.U.1EA7.u.vào.lúc         <chr> "20 March 2020  9:13 AM", "20 March 2020  9:16 AM
$ Đã.hoàn.thành                            <chr> "20 March 2020  9:24 AM", "20 March 2020  9:18 AM
$ Th.U.1EDD.i.gian.th.U.1EF1.c.hi.U.1EC7.n <chr> "10 phút 56 giây", "2 phút 17 giây", "1 phút 6 gi
$ Đi.U.1EC3.m.10.00                        <chr> "8,50", "8,50", "9,50", "10,00", "10,00", "9,00",
$ Q..1..1.00                               <chr> "1,00", "1,00", "1,00", "1,00", "1,00", "1,00",
$ Q..2..1.00                               <chr> "1,00", "1,00", "1,00", "1,00", "1,00", "1,00",
$ Q..3..1.00                               <chr> "1,00", "1,00", "1,00", "1,00", "1,00", "1,00",
$ Q..4..1.00                               <chr> "1,00", "1,00", "1,00", "1,00", "1,00", "1,00",
$ Q..5..1.00                               <chr> "1,00", "1,00", "1,00", "1,00", "1,00", "1,00",
$ Q..6..1.00                               <chr> "1,00", "1,00", "1,00", "1,00", "1,00", "1,00",
$ Q..7..1.00                               <chr> "0,50", "0,50", "0,50", "1,00", "1,00", "1,00",
$ Q..8..1.00                               <chr> "1,00", "1,00", "1,00", "1,00", "1,00", "1,00",
$ Q..9..1.00                               <chr> "0,00", "0,00", "1,00", "1,00", "1,00", "0,00",
$ Q..10..1.00                              <chr> "1,00", "1,00", "1,00", "1,00", "1,00", "1,00",
```

Ống %>%:

- Toán tử ống - pipe operator %>%
  - Để ta viết chuỗi liên tiếp các xử lý với nhau
  - Đọc %>% như "then"
  - Làm cho code dễ hiểu, viết, dễ đọc hơn
- x %>% f(y) -> f(x,y)
- x %>% f(z,.) -> f(z, x)

Các hàm:

- filter: chọn hàng
- arrange: sắp thứ tự dòng
- select: chọn cột
- rename: đổi tên cột
- distinct: tìm các hàng riêng biệt

- mutate: thêm các biến mới
- summarise: tóm tắt trên một tập dữ liệu
- sample_n: lấy mẫu từ tập dữ liệu.

Thí dụ:

- filter

Source:
grads %>% filter(Đã.hoàn.thành == '-')
Output

```
Mã.s.U.1ED1..ID              Tình.tr.U.1EA1.ng Đã.b.U.1EAF.t.d.U.1EA7.u.vào.lúc Đã.hoàn.thành
1         1912916 Chua bao gi<U+1EDD> g<U+1EED>i           20 March 2020  2:20 PM             -
Th.U.1EDD.i.gian.th.U.1EF1.c.hi.U.1EC7.n Đi.U.1EC3.m.10.00 Q..1..1.00 Q..2..1.00 Q..3..1.00 Q..4..
1                                        -                 -          -          -          -
Q..6..1.00 Q..7..1.00 Q..8..1.00 Q..9..1.00 Q..10..1.00
1          -          -          -          -           -
```

- select

Source:
grads %>% select(Mã.s.U.1ED1..ID, Đi.U.1EC3.m.10.00) %>% head(3)
Output

```
Mã.s.U.1ED1..ID Đi.U.1EC3.m.10.00
1         1911478              8,50
2         1914405              8,50
```

Các tùy chọn cho mutate/sumarise

- mean: trung bình trong các nhóm
- sum: tổng trong các nhóm
- sd: độ lệch chuẩn trong các nhóm
- max: lớn nhất trong các nhóm
- ...

Thí dụ:
Source:
processed_samples %>% group_by(Mã.s.U.1ED1..ID) %>% summarise(Mã.s.U.1ED1..ID = n()) %>% arrange(-Mã.s.U.1ED1..ID)
Output

```
 Mã.s.U.1ED1..ID
<int>
1               3
2               2
3               2
4               1
5               1
6               1
```

3 Cấu trúc

3.1 Viết hàm

```
# Function template
my_function = function(input1, input2, ..., inputN) # Define inputs
{
# Define 'output' using input1,...,inputN
return(output) # Return this output
}
```

Source

```
add_numbers = function(x, y){
# Define z as sum of x and y
z = x + y
# Return z
return(z)
}
```

add_numbers(2, 4)
Output
6

3.2 Xử lý chuỗi

Các hàm hàm hữu dụng:

- str_length: chiều dài chuỗi
- str_c: tổ hợp chuỗi
- str_sub: tập con chuỗi
- str_to_lower: chuỗi đến chữ thường

Thí dụ:
# New vector of strings
x = c("apple", "banana", "pear", "213")

- str_replace

Source x %>% str_replace("[0-9]", "\\?")
Output
[1] "apple" "banana" "pear" "?13"

- str_replace_all

Source x %>% str_replace_all("[0-9]", "\\?")
Output
[1] "apple" "banana" "pear" "???"

3.3 Xử lý ngày

Tạo đối tượng ngày giờ từ chuỗi

strptime Chuyển đổi chuỗi ký tự thành ngày giờ

```
strptime("02/27/92 23:03:20", format="%m/%d/%y %H:%M:%S")
[1] "1992-02-27 23:03:20 +07"
```

Xử lý thời gian

```
earlier <- strptime("2000-01-01 00:00:00","%Y-%m-%d %H:%M:%S")
later <- strptime("2000-01-01 00:00:20","%Y-%m-%d %H:%M:%S")
later - earlier
```

Output
Time difference of 20 secs

4 Trực quan dữ liệu

4.1 ggplot2

Chúng ta sẽ trực quan dữ liệu bằng gói ggplot2
library(ggplot2)
ggplot2 là một công cụ có hệ thống, nhất quán và hiệu quả về thời gian mà bạn có thể sử dụng để tạo trực quan hóa chất lượng cao.

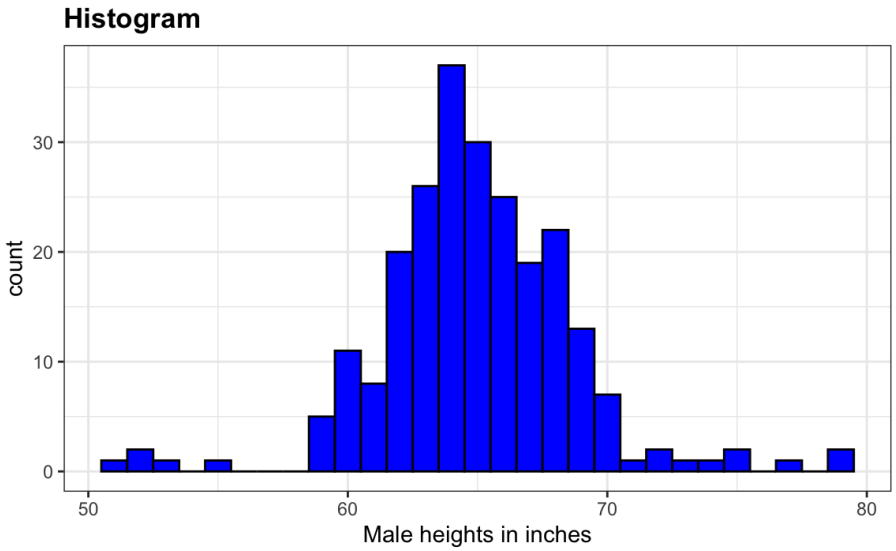
Bạn bắt đầu với dữ liệu của mình và sau đó bạn chỉ định hình học cho các thành phần của dữ liệu đó, chẳng hạn dùng vòng tròn thể hiện cho dân số, sau đó vẽ những hình học đó dựa trên tỷ lệ dữ liệu của bạn.

- Tạo một biểu đồ và định nghĩa các biến cần ánh xạ đến đến đồ thị
  - ggplot(data = data_frame, aes(x = variable_1, y = variable_2))
- Gắn đường thẳng để đồ thị
  - ggplot(data = data_frame, aes(x = variable_1, y = variable_2)) + geom_line()
- Gắn tựa đề biểu đồ và nhãn các trục
  - ggplot(data = data_frame, aes(x = variable_1, y = variable_2)) + geom_line() + labs(title = "Title of Graph", x = "new x label", y = "new y label")
- Thay đổi màu nền
  - ggplot(data = data_frame, aes(x = variable_1, y = variable_2)) + geom_line() + labs(title = "Title of Graph", x = "new x label", y = "new y label") +theme(panel.background = element_rect(fill = "background color")

Một số đồ thị:

- geom_point()
- geom_line()
- geom_bar()
- geom_boxplot()
- geom_histogram()
- geom_density()

```
heights %>%
filter(sex == "Female") %>%
ggplot(aes(height)) +
geom_histogram(binwidth = 1, fill = "blue", col = "black") +
xlab("Male heights in inches") +
ggtitle("Histogram")
```



Tài liệu

[Dal] Dalgaard, P. *Introductory Statistics with R.* Springer 2008.

[K-Z] Kenett, R. S. and Zacks, S. *Modern Industrial Statistics: with applications in R, MINITAB and JMP,* 2nd ed., John Wiley and Sons, 2014.

[Ker] Kerns, G. J. *Introduction to Probability and Statistics Using R,* 2nd ed., CRC 2015.