**#INTRODUCTION TO DATA TABLE**

**-DT[i(row), j(column), by(group)]**

-Creating data.table :

1.data.table()

2.as.data.table()

3.fread()

-Example create data.table :

**\*x\_dt <- data.table(id = 1:2, name = c(“a”, “b”))**

-to convert existing R Object to data.table :

**\*as.data.table(y)**

-cek jumlah row : nrow(x)

-cek jumlah column : ncol(x)

-cek dimensi data :dim(x)

-data table doesn’t automatically convert to a factor

-data table never sets or used rownames()

-Creating a data table :

# Load data.table

library(data.table)

# Create the data.table X

X <- data.table(id = c("a", "b", "c"), value = c(0.5, 1.0, 1.5))

# View X

X

-get the number of column :

# Get number of columns in batrips

col\_number <- ncol(batrips)

-print top 8 rows :

# Print the first 8 rows

head(batrips, 8)

-print the last 8 row :

# Print the last 8 rows

tail(batrips, 8)

-print the structure :

# Print the structure of batrips

str(batrips)

-Filter and subset

-filter : **batrips[3:4] or batrips[3:4,]**

-subset everything except : **batrips[-(1:5)] or batrips[!(1:5)]**

-to get the last row : **batrips[.N]**

-Jumlah row dikurang 10 : batrips[1:(.N-10)]

-Subset logical expression : batrips[subscription\_type==”Subscriber”]

-if batrips was only a data frame : batrips[batrips$subscription\_type ==”Subscriber”,]

-Another subset : batrips[start\_terminal == 58 & end\_terminal !=65]

-filter 3rd row :

# Filter third row

row\_3 <- batrips[3]

row\_3

\*tidak usah pakai koma seperti data.frame

-Filter row 10:20

# Filter rows 10 through 20

rows\_10\_20 <- batrips[10:20]

rows\_10\_20

-filter row ke 1, 6 , and 10 :

# Filter the 1st, 6th and 10th rows

rows\_1\_6\_10 <- batrips[c(1,6,10)]

rows\_1\_6\_10

-select all rows, except :

# Select all rows except the first two

not\_first\_two <- batrips[-(1:2)]

not\_first\_two

# Select all rows except 1 through 5 and 10 through 15

exclude\_some <- batrips[-c(1:5, 10:15)]

exclude\_some

# Select all rows except the first and last

not\_first\_last <- batrips[-c(1, .N)] # Or batrips[-c(1, nrow(batrips))]

not\_first\_last

# Filter all rows where start\_station is "MLK Library"

trips\_mlk <- batrips[start\_station == "MLK Library"]

trips\_mlk

# Filter all rows where start\_station is "MLK Library" AND duration > 1600

trips\_mlk\_1600 <- batrips[start\_station == "MLK Library" & duration > 1600]

trips\_mlk\_1600

# Filter all rows where `subscription\_type` is not `"Subscriber"`

customers <- batrips[!(subscription\_type=="Subscriber")]

customers

# Filter all rows where start\_station is "Ryland Park" AND subscription\_type is not "Customer"

ryland\_park\_subscribers <- batrips[start\_station=="Ryland Park" &!(subscription\_type=="Customer")]

ryland\_park\_subscribers

-Helpers for filtering

-batrips[start\_station %like% “^San Francisco”] -> data.table

-batrips[grepl(“^San Francisco”, start\_station)] -> data,frame

-batrips[duration %between% c(2000,3000)] -> data.table

-batrips[duration >= 2000 & duration <= 3000] -> data.frame

-batrips[start\_station %chin% c(“Japantown”, “Mezes Park”, “MLK Library”)] -> data.table

-batrips[start\_station %in% c(“Japantown”, “Mezes Park”, “MLK Library”)] -> data.frame

# Filter all rows where end\_station contains "Market"

any\_markets <- batrips[end\_station %like% "Market"]

any\_markets

# Filter all rows where end\_station ends with "Market"

end\_markets <- batrips[end\_station %like% "Market$"]

end\_markets

# Filter all rows where trip\_id is 588841, 139560, or 139562

filter\_trip\_ids <- batrips[trip\_id %in% c(588841, 139560, 139562)]

filter\_trip\_ids

# Filter all rows where duration is between [5000, 6000]

duration\_5k\_6k <- batrips[duration %between% c(5000, 6000)]

duration\_5k\_6k

# Filter all rows with specific start stations

two\_stations <- batrips[start\_station %chin% c("San Francisco City Hall", "Embarcadero at Sansome")]

two\_stations

**#SELECTING COLUMNS FROM A DATA.TABLE**

-Select Column :

**\*ans <- batrips[, c(“trip\_id”, “duration”)]** -> select column “trip\_id” and “duration”

-the difference between data frame and data table : when subset data.frame, type of data no longer data.frame, but vector. Meanwhile in data.table consistency in data.table format.

-Using number when subsetting column is not recommended because column can change in the future.

-Deselecting column with character vectors :

**\*batrips[, -c(“start\_date”, “end\_date”, “end\_station”)] or**

**\*batrips[, !c(“start\_date”, “end\_date”, “end\_station”)]**

-Selecting column using a list and rename the column :

**\*batrips[, list(trip\_id, dur = duration)]** -> rename duration column to dur

-when selecting single column if you used list the result will return to data.table. when do not used list, the result will be a vector.

**-batrips[, list(trip\_id)]** : result will be data.table

**-batrips[, trip\_id]** : the result will be in vector

-another format writing for list

**-batrips[, list(trip\_id, duration)] = -batrips[, .(trip\_id, duration)]**

# Select bike\_id and trip\_id using a character vector

df\_way <- batrips[, c("bike\_id", "trip\_id")]

# Select start\_station and end\_station cols without a character vector

dt\_way <- batrips[, .(start\_station, end\_station)]

# Deselect start\_terminal and end\_terminal columns

drop\_terminal\_cols <- batrips[, -c("start\_terminal", "end\_terminal")]

-Computing on column in data.table way

\***ans <- batrips[,mean(duration)]** -> data.table

\***ans <- mean(batrips[,”duration”])** -> data.frame

-Computing on rows and columns

\***batrips[start\_station ==”Japan Town”, mean(duration)]**

-special symbol .N in j

\***batrips[start\_station == “Japan Town”, .N]** -> data.table

\***nrow(batrips[batrips$start\_station == “Japan Town”, ])** -> data.frame

-Computing in J Exercise :

# Calculate median duration using the j argument

median\_duration <- batrips[,median(duration)]

# Get median duration after filtering

batrips[end\_station == "Market at 10th" & subscription\_type == "Subscriber", median(duration)]

-computing in j (ii) exercise :  
# Compute duration of all trips

trip\_duration <- batrips[, difftime(end\_date, start\_date, units ="min")]

head(trip\_duration)

-Advanced computation in j :

-calculate more than 1 summary

\***batrips[, .(mn\_dur = mean(duration),**

**md\_dur = median(duration))]**

-combining with I in advanced j calculation :

\***batrips[start\_station==”Japan Town”, .(mn\_dur = mean(duration), md\_dur = median(duration))]**

-computing in j (iii) exercise :

# Calculate the average duration as mean\_durn

mean\_duration <- batrips[, .(mean\_durn= mean(duration))]

# Get the min and max duration values

min\_max\_duration <- batrips[, .(min(duration), max(duration))]

# Calculate the average duration and the date of the last ride

other\_stats <- batrips[, .(mean\_duration=mean(duration), last\_ride=max(end\_date))]

-combining I and j exercise :

batrips[start\_station=="Townsend at 7th" & duration<500,

                          .(min\_dur=min(duration),

                            max\_dur=max(duration))]

# Plot the histogram of duration based on conditions

batrips[start\_station == "Townsend at 7th" & duration < 500, hist(duration)]

**# COMPUTATION BY GROUPS**

-The by argument

-How many trips happened from each start station

**\*ans <- batrips[, .N, by = “start\_station”] ->** column will resulting N as a header

**\* ans <- batrips[, .N, by = .(start\_station)]** -> will be produce same result

-allow renaming grouping column on the fly

**\*ans <- batrips[, .(no\_trips = .N), by = .(start = start\_station)]**

-Double grouping :

**\*ans <- batrips[ , .N, by = .(start\_station, mon = month(start\_date))]**

-Computing stats by groups :

# Compute the mean duration for every start\_station

mean\_start\_stn <- batrips[, .(mean\_duration=mean(duration)), by = start\_station]

-computing stats by groups(ii) :

# Compute the mean duration for every start\_station

mean\_start\_stn <- batrips[, .(mean\_duration=mean(duration)), by = start\_station]

# Compute the mean duration grouped by start\_station and month

mean\_start\_station <- batrips[, .(mean\_duration=mean(duration)), by=.(start\_station, month(start\_date))]

-computing multiple stats :

# Compute mean of duration and total trips grouped by start and end stations

aggregate\_mean\_trips <- batrips[, .(mean\_duration=mean(duration), total\_trips=.N), by=.(start\_station, end\_station)]

# Compute min and max duration grouped by start station, end station, and month

aggregate\_min\_max <- batrips[, .(min\_duration=min(duration), max\_duration=max(duration)), .(start\_station, end\_station, month(start\_date))]

-Chaining data.table expression

\***batrips[duration > 3600][order(duration)][1:3]** -> filter duration > 3600, sort the duration and select top 3

-sort the lowest mean\_duration and take top 3 start\_station :

\***batrips[, .(mn\_dur = mean(duration)), by = “start\_station”][order(mn\_dur)][1:3]**

-uniqueN : to calculate number of unique variable in dataset

**\*id <- c(1,2,2,1)**; uniqueN(id) : the result is 2(number unique character)

\*id val

1 1

2 2

2 3

1. 4

**\*uniqueN(x)** : the result will be 4, because no duplicate value in all column

**\*uniqueN(x, by =”id”)** : the result will be, because unique in column id there are 2 variable duplicate

-uniqueN together with by :

**\*batrips[, uniqueN(bike\_id), by = month(start\_date)] -> jumlah unik bike id per month**

-Ordering Rows :

# Compute the total trips grouped by start\_station and end\_station

trips\_dec <- batrips[, .N, by = .(start\_station,

                                  end\_station)]

# Arrange the total trips grouped by start\_station and end\_station in decreasing order

trips\_dec <- batrips[, .N, by = .(start\_station,

                                  end\_station)][order(-N)]

-The top 5 destination :

# Top five most popular destinations

top\_5 <- batrips[, .N, by = end\_station][order(-N)][1:5]

-what is the most popular destination from each start station :

# Compute most popular end station for every start station

popular\_end\_station <- trips\_dec[, .(end\_station = end\_station[1]),

                                 by = start\_station]

-combining I, j, and by :

# Find the first and last ride for each start\_station

first\_last <- batrips[order(start\_date),

                      .(start\_date = start\_date[c(1, .N)]),

                      by = start\_station]