# [**Analisis Data COVID19 di Indonesia**](https://academy.dqlab.id/main/package/practice/253)

## Bermain API

Saat Anda menjalankan fungsi GET(), pada dasarnya hal yang terjadi adalah Anda membuat sebuah permintaan kepada server penyedia API. Permintaan Anda tersebut selanjutnya diproses dan dijawab oleh server sebagai sebuah respon. Objek resp yang telah Anda buat memiliki informasi respon oleh server.

Ada tiga informasi utama dalam sebuah respon API, yaitu status, headers, dan body. Status memiliki informasi apakah permintaan Anda berhasil atau tidak dan dinyatakan dalam status code, headers umumnya mengandung informasi metadata, sedangkan body berisikan konten atas permintaan yang telah dibuat.

## Status Code

Ada beberapa jenis *status code* yang umumnya dijumpai, antara lain:

* 1. Artinya permintaan sukses dipenuhi.
  2. Artinya berkas yang diminta tidak dapat ditemukan.
  3. Artinya akses permintaan ditolak.
  4. Artinya terjadi kesalahan pada server.

library(httr)

resp <- GET("https://data.covid19.go.id/public/api/update.json")

status\_code (resp)

Informasi lengkap status API :

<https://restfulapi.net/http-status-codes/>

## Mengekstrak isi respon :

Respon API dengan *status code* 200 menyatakan bahwa permintaan Anda berhasil dipenuhi dan konten yang diminta tersedia untuk diekstrak. Selain itu Anda juga telah mengetahui lewat fungsi headers() bahwa konten yang diminta tersedia dalam bentuk *application/json*, yaitu berkas JSON.

Selanjutnya Anda dapat mengekstrak konten tersebut dengan menggunakan fungsi content(). Fungsi content() tersebut secara cerdas akan menyesuaikan output sesuai dengan jenis berkas yang tersedia, dalam hal ini adalah berkas JSON. Jalankan fungsi tersebut pada obyek resp dan tambahkan argumen as = "parsed" dan simplifyVector = TRUE. Simpanlah hasilnya sebagai cov\_id\_raw!

library(httr)

resp <- GET("https://data.covid19.go.id/public/api/update.json")

cov\_id\_raw <- content(resp, as ="parsed", simplifyVector = TRUE)

## Mengekstrak isi Respon - Pt 2

Dengan menggunakan fungsi length() dan names() cobalah amati ada berapa komponen serta apa saja nama komponen dalam obyek cov\_id\_raw tersebut! Kemudian ekstraklah komponen ke-2 dan simpan dengan nama cov\_id\_update.

length(cov\_id\_raw)

names(cov\_id\_raw)

cov\_id\_update <- cov\_id\_raw$update

# [**Analisis Data COVID19 di Jawa**](https://academy.dqlab.id/main/package/practice/253) **Tengah**

## Apa Kabar Jawa Tengah?

library(httr)

resp\_jateng <- GET("https://data.covid19.go.id/public/api/prov\_detail\_JAWA\_BARAT.json")

cov\_jateng\_raw <- content(resp\_jateng, as = "parsed", simplifyVector = TRUE)

names(cov\_jateng\_raw)

cov\_jateng\_raw$kasus\_total

cov\_jateng\_raw$meninggal\_persen

cov\_jateng\_raw$sembuh\_persen

## Memperoleh Informasi yang Lebih Lengkap

Kabar baiknya adalah informasi tersebut juga disediakan oleh covid19.go.id melalui permintaan API yang telah Anda buat sebelumnya. Data historis perkembangan COVID-19 tersebut tersimpan dengan nama list\_perkembangan. Silakan Anda ekstrak data tersebut dari cov\_jateng\_raw dan simpanlah hasilnya sebagai obyek bernama cov\_jateng! Amati struktur cov\_jateng menggunakan fungsi str() dan head().

cov\_jateng <- cov\_jateng\_raw$list\_perkembangan

str(cov\_jateng)

head(cov\_jateng)

## Menjinakkan Data

Sebelum itu, silakan Anda **aktifkan paket dplyr** yang akan dipergunakan untuk melakukan pengolahan data.

Ada beberapa tahapan yang akan Anda lakukan untuk menjinakan data cov\_jateng, yaitu:

1. Menghapus kolom “DIRAWAT\_OR\_ISOLASI” dan “AKUMULASI\_DIRAWAT\_OR\_ISOLASI”
2. Menghapus semua kolom yang berisi nilai kumulatif
3. Mengganti nama kolom “KASUS” menjadi “kasus\_baru”
4. Merubah format penulisan kolom berikut menjadi huruf kecil
   * kolom MENINGGAL
   * kolom SEMBUH
5. Memperbaiki data pada kolom tanggal

Lengkapilah baris kode berikut untuk melakukan menjinakan data sesuai dengan tahapan yang telah Anda rencanakan! Anda akan menggunakan operator *pipe* (%>%) untuk merangkai fungsi menjadi sebuah *pipeline*. Simpan hasil pengolahan Anda dengan nama new\_cov\_jateng.

library(dplyr)

new\_cov\_jateng <-

cov\_jateng %>%

select(-contains("DIRAWAT\_OR\_ISOLASI")) %>%

select(-starts\_with("AKUMULASI")) %>%

rename(

kasus\_baru = KASUS,

meninggal = MENINGGAL,

sembuh = SEMBUH

) %>%

mutate(

tanggal = as.POSIXct(tanggal / 1000, origin = "1970-01-01"),

tanggal = as.Date(tanggal)

)

str(new\_cov\_jateng)

## Menunjukkan Melalui Gambar

Berdasarkan templat tersebut, komponen utama untuk membuat visualisasi antara lain adalah tabel data, kolom data, serta bentuk geometri untuk mempresentasikan data. Sebagai contoh untuk membuat scatter-plot yang diperlukan adalah bentuk geometri titik (geom\_col()), line-chart memerlukan geometri garis (geom\_line()), sedangkan bar-chart memerlukan bentuk geometri batang atau kolom (geom\_bar() atau geom\_col())

Library

(ggplot2)

library(hrbrthemes)

ggplot(new\_cov\_jateng, aes(x = tanggal, y = kasus\_baru)) +

geom\_col()

## Menunjukkan Melalui Gambar - Pt 2

library(ggplot2)

library(hrbrthemes)

ggplot(new\_cov\_jateng, aes(tanggal, kasus\_baru)) +

geom\_col(fill="salmon")+

labs(

x=NULL,

y="Jumlah kasus",

title = "Kasus Harian Positif COVID-19 di Jawa Tengah",

subtitle = "Terjadi pelonjakan kasus di awal bulan Juli akibat klaster Secapa AD Bandung", caption = "Sumber data : covid.19.go.id"

) +

theme\_ipsum(

base\_size = 13,

plot\_title\_size = 21,

grid = "Y",

ticks = TRUE

) +

theme(plot.title.position = "plot")

## Grafik untuk Kasus Sembuh

library(ggplot2)

library(hrbrthemes)

ggplot(new\_cov\_jateng, aes(tanggal, sembuh)) +

geom\_col(fill = "olivedrab2") +

labs(

x = NULL,

y = "Jumlah kasus",

title = "Kasus Harian Sembuh Dari COVID-19 di Jawa Tengah",

caption = "Sumber data: covid.19.go.id"

) +

theme\_ipsum(

base\_size = 13,

plot\_title\_size = 21,

grid = "Y",

ticks = TRUE

) +

theme(plot.title.position = "plot")

## Grafik untuk Kasus Meninggal

library(ggplot2)

library(hrbrthemes)

ggplot(new\_cov\_jateng, aes(tanggal, meninggal)) +

geom\_col(fill ="darkslategray4") +

labs(

x = NULL,

y = "Jumlah kasus",

title = "Kasus Harian Meninggal Akibat COVID-19 di Jawa Tengah",

caption = "Sumber data: covid.19.go.id"

) +

theme\_ipsum(

base\_size = 13,

plot\_title\_size = 21,

grid = "Y",

ticks = TRUE

) +

theme(plot.title.position = "plot")

## Apakah Pekan ini Lebih Baik?

library(dplyr)

library(lubridate)

cov\_jateng\_pekanan <- new\_cov\_jateng %>%

count(

tahun = year(tanggal),

pekan\_ke = week(tanggal),

wt = kasus\_baru,

name = "jumlah"

)

glimpse(cov\_jateng\_pekanan)

## Menjawab Pertanyaan

Pertanyaan baru muncul di benak Anda setelah melakukan inspeksi terhadap data cov\_jateng\_pekanan tersebut: “Apakah pekan ini lebih baik dari pekan kemarin?”.

Demi menjawab hal tersebut Anda melakukan kalkulasi sederhana dengan tahapan berikut:

* Membuat kolom baru yang berisi jumlah kasus baru dalam satu pekan sebelumnya. Kolom ini diberi nama “jumlah\_pekanlalu”.
* Mengganti nilai NA pada kolom “jumlah\_pekanlalu” dengan nilai 0
* Melakukan komparasi antara kolom “jumlah” dengan kolom “jumlah\_pekanlalu”. Hasil komparasi ini disimpan dalam kolom baru dengan nama “lebih\_baik”, isinya adalah TRUE apabila jumlah kasus baru pekan ini lebih rendah dibandingkan jumlah kasus pekan lalu

Lengkapilah baris kode berikut untuk mengerjakan tahapan yang telah Anda rencanakan! Anda akan menggunakan fungsi lag() dari dplyr untuk membuat kolom “jumlah\_pekanlalu”. Perhatikan bahwa disini fungsi tersebut dituliskan sebagai dplyr::lag() untuk menghindari konflik dengan fungsi lag() dari paket stats. Inspeksi hasil pekerjaan Anda dengan menggunakan fungsi glimpse()!

library(dplyr)

cov\_jateng\_pekanan <-

cov\_jateng\_pekanan %>%

mutate(

jumlah\_pekanlalu = dplyr::lag(jumlah, 1),

jumlah\_pekanlalu = ifelse(is.na(jumlah\_pekanlalu), 0, jumlah\_pekanlalu),

lebih\_baik = jumlah < jumlah\_pekanlalu

)

glimpse(cov\_jateng\_pekanan)

## Membuat Bar Chart

library(ggplot2)

library(hrbrthemes)

ggplot(cov\_jateng\_pekanan, aes(pekan\_ke, jumlah, fill = lebih\_baik)) +

geom\_col(show.legend = FALSE) +

scale\_x\_continuous(breaks = 9:29, expand = c(0, 0)) +

scale\_fill\_manual(values = c("TRUE" = "seagreen3", "FALSE" = "salmon")) +

labs(

x = NULL,

y = "Jumlah kasus",

title = "Kasus Pekanan Positif COVID-19 di Jawa Tengah",

subtitle = "Kolom hijau menunjukan penambahan kasus baru lebih sedikit dibandingkan satu pekan sebelumnya",

caption = "Sumber data: covid.19.go.id"

) +

theme\_ipsum(

base\_size = 13,

plot\_title\_size = 21,

grid = "Y",

ticks = TRUE

) +

theme(plot.title.position = "plot")

## Pola dan Dinamika

Ada yang akhirnya sembuh, namun tak sedikit pula yang meninggal akibat COVID-19. Sementara itu penambahan kasus baru terus terjadi di masyarakat. Hal ini mungkin memicu pertanyaan lain di diri Anda: “Hingga saat ini ada berapa kasus yang masih aktif?”. Aktif dalam artian sedang dalam perawatan atau isolasi.

Informasi ini sebenarnya telah disediakan di dalam respon API covid19.go.id yang Anda minta. Namun tidak ada salahnya jika Anda mencoba menghitungnya sendiri, apakah Anda setuju?

Jumlah kasus aktif dapat dihitung dengan cara mengurangi jumlah akumulasi positif dengan jumlah akumulasi sembuh dan jumlah akumulasi meninggal. Anda dapat menggunakan fungsi cumsum() untuk menghitung nilai akumulasi dari suatu vektor numerik. Lengkapilah baris kode berikut untuk menghitung jumlah akumulasi kasus aktif, kasus sembuh, dan kasus meninggal!

library(dplyr)

cov\_jateng\_akumulasi <-

new\_cov\_jateng %>%

transmute(

tanggal,

akumulasi\_aktif = cumsum(kasus\_baru) - cumsum(sembuh) - cumsum(meninggal),

akumulasi\_sembuh = cumsum(sembuh),

akumulasi\_meninggal = cumsum(meninggal)

)

tail(cov\_jateng\_akumulasi)

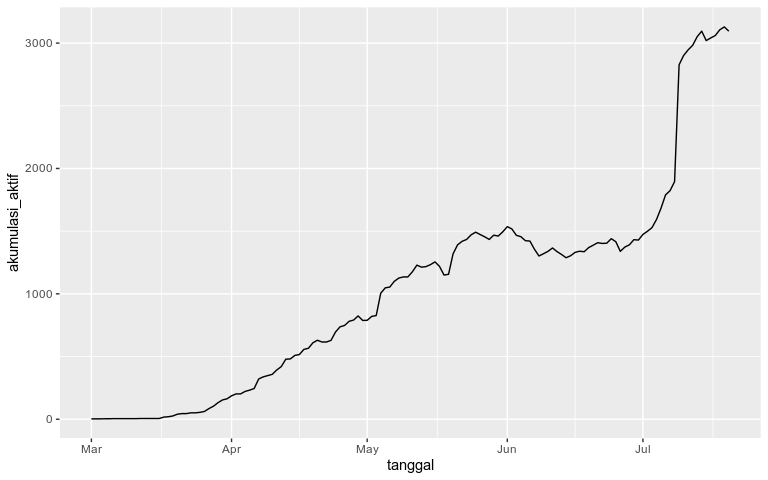
## Membuat Line Chart

Sekarang cobalah Anda buat *line-chart* pola kasus aktif dengan menggunakan fungsi geom\_line(). Sebagai pengingat, Anda dapat menggunakan templat kode berikut untuk membuat grafik menggunakan ggplot2():

ggplot(data = ..., aes(x = ..., y = ...)) +

geom\_xxx()

Adapun hasil yang diharapkan adalah sebagai berikut.



library(ggplot2)

ggplot(data = cov\_jateng\_akumulasi, aes(x = tanggal, y = akumulasi\_aktif)) +

geom\_line()

## Transformasi Data

library(dplyr)

library(tidyr)

dim(cov\_jateng\_akumulasi)

cov\_jateng\_akumulasi\_pivot <-

cov\_jateng\_akumulasi %>%

gather(

key = "kategori",

value = "jumlah",

-tanggal

) %>%

mutate(

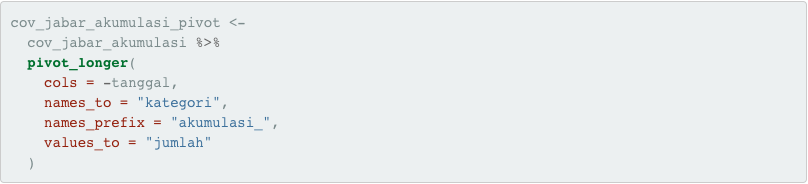
kategori = sub(pattern = "akumulasi\_", replacement = "", kategori)

)

dim(cov\_jateng\_akumulasi\_pivot)

glimpse(cov\_jateng\_akumulasi\_pivot)

## pivot\_longer()

Semenjak tidyr versi 1.0.0, Anda disarankan untuk menggunakan fungsi pivot\_longer() sebagai pengganti gather() dan pivot\_wider() sebagai pengganti spread(). pivot\_longer() dan pivot\_wider() memiliki fitur yang lebih lengkap dibandingkan gather() dan spread(). Proses transformasi cov\_jateng\_akumulasi menjadi cov\_jateng\_akumulasi\_pivot dapat dikerjakan dengan menggunakan pivot\_longer() sebagai berikut:

Anda dapat mencoba mempraktekkan potongan coding tersebut pada R Studio pada desktop masing-masing dan apakah Anda menemukan perbedaan signifikan antara baris kode di atas dan baris kode sebelumnya yang Anda gunakan?

## Tahap Terakhir

Anda hampir selesai! Pada tahap terakhir ini Anda cukup salin dan jalankan baris kode berikut di konsol untuk membuat grafik komparasi antara akumulasi kasus aktif, kasus sembuh dan kasus meninggal:

library(ggplot2)

library(hrbrthemes)

ggplot(cov\_jateng\_akumulasi\_pivot, aes(tanggal, jumlah, colour = (kategori))) +

geom\_line(size = 0.9) +

scale\_y\_continuous(sec.axis = dup\_axis(name = NULL)) +

scale\_colour\_manual(

values = c(

"aktif" = "salmon",

"meninggal" = "darkslategray4",

"sembuh" = "olivedrab2"

),

labels = c("Aktif", "Meninggal", "Sembuh")

) +

labs(

x = NULL,

y = "Jumlah kasus akumulasi",

colour = NULL,

title = "Dinamika Kasus COVID-19 di Jawa Tengah",

caption = "Sumber data: covid.19.go.id"

) +

theme\_ipsum(

base\_size = 13,

plot\_title\_size = 21,

grid = "Y",

ticks = TRUE

) +

theme(

plot.title = element\_text(hjust = 0.5),

legend.position = "top"

)