**Apa Kabar Jawa Barat?**

Anda telah berhasil menjawab pertanyaan mengenai kondisi COVID-19 di Indonesia dengan menggunakan data yang langsung diambil menggunakan API. Namun bagaimana jika Anda ingin berfokus dengan data COVID-19 di provinsi tempat menetap saat ini?

covid19.go.id menyediakan data kasus COVID-19 tingkat provinsi di alamat API yang berbeda. Sebagai contoh data mengenai COVID-19 Jawa Barat, tempat tinggal saya sekarang, tersedia di <https://data.covid19.go.id/public/api/prov_detail_JAWA_BARAT.json> dan dapat diakses menggunakan baris kode berikut:

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

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

Sekarang jalankanlah fungsi names() pada cov\_jabar\_raw tersebut untuk mengetahui nama-nama elemen utama yang tersedia dan jawablah pertanyaan berikut:

* Berapa jumlah total kasus COVID-19 di Jawa Barat?
* Berapa persentase kematian akibat COVID-19 di Jawa Barat?
* Berapa persentase tingkat kesembuhan dari COVID-19 di Jawa Barat?

## Memperoleh Informasi yang Lebih Lengkap

Informasi umum mengenai COVID-19 di Jawa Barat telah Anda dapatkan. Namun informasi akan lebih lengkap jika Anda memiliki data perkembangan COVID-19 dari waktu ke waktu, apakah Anda setuju?

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\_jabar\_raw dan simpanlah hasilnya sebagai obyek bernama cov\_jabar! Amati struktur cov\_jabar menggunakan fungsi str() dan head().

**Menjinakkan Data**

Setelah mengekstrak dan mengamati cov\_jabar, Anda menemukan beberapa kejanggalan pada data tersebut. Diantaranya adalah kejanggalan data pada kolom tanggal dan format penulisan kolom yang tidak konsisten. Sekarang Anda akan mencoba melakukan beberapa tahapan untuk menjinakan data tersebut sehingga dapat diolah dan dianalisis dengan lebih mudah.

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\_jabar, 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\_jabar.

## Menunjukkan Melalui Gambar

Akhirnya Anda berhasil menjinakan data cov\_jabar sehingga lebih mudah untuk diolah, selamat! Memiliki data yang rapi memang menyenangkan, apakah Anda setuju dengan hal tersebut?

Setelah memiliki data yang jinak, sekarang saatnya Anda mengekspresikan data tersebut dalam bentuk lain yang harapannya lebih mudah dicerna: grafik. Anda akan memulai merancang visualisasi yang memiliki estetika dengan menggunakan **paket ggplot2**dan**paket hrbrthemes.** Aktifkanlah kedua paket tersebut!

Berikut merupakan templat kode untuk membuat visualisasi menggunakan ggplot2:

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

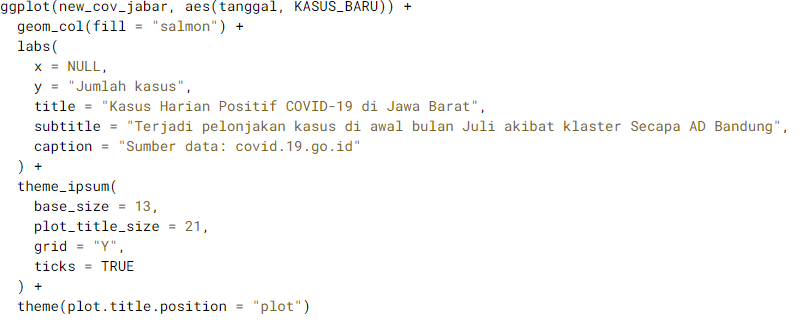
geom\_xxx()

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()).

Lengkapi baris kode berikut untuk membuat bar-chart jumlah kasus baru harian COVID-19 di Jawa Barat menggunakan data new\_cov\_jabar! Pergunakan kolom “tanggal” sebagai sumbu-x.

## Menunjukkan Melalui Gambar - Pt 2

Anda telah berhasil membuat bar-chart yang diminta. Namun grafik tersebut belum memiliki informasi yang jelas serta cenderung membosankan untuk dilihat. Sekarang silakan buat kembali grafik tersebut dengan menggunakan baris kode yang telah dimodifikasi berikut:



Hasil dari potongan coding di atas adalah sebagai berikut.

## Grafik untuk Kasus Sembuh

Sekarang buatlah grafik serupa dengan menggunakan data kasus sembuh  Pergunakan warna “olivedrab2” untuk grafik kasus sembuh!

Hasil yang diharapkan adalah sebagai berikut :

## Apakah Pekan ini Lebih Baik?

Setelah mengamati grafik perkembangan kasus Anda menyadari bahwa terjadi fluktuasi pertambahan kasus harian. Dilandasi hal tersebut Anda kemudian ingin mencoba mengamati bagaimana perkembangan kasus dalam rentang waktu pekanan. Bagaimanakah caranya?

Anda dapat dengan mudah bekerja dengan data tanggal apabila menggunakan paket **lubridate.** Adapun yang akan digunakan untuk mengekstrak informasi pekan dalam satu tahun adalah fungsi week().

Lengkapilah baris kode berikut untuk menghitung pertambahan kasus pekanan dan simpanlah hasilnya sebagai **cov\_jabar\_pekanan**! Anda juga diminta untuk menggunakan fungsi glimpse() dari dplyr untuk melakukan inspeksi data.

**Menjawab Pertanyaan**

Pertanyaan baru muncul di benak Anda setelah melakukan inspeksi terhadap data cov\_jabar\_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()!

## Membuat Bar Chart

Anda sungguh luar biasa! Dengan menggunakan data hasil perhitungan tersebut maka sekarang Anda dapat membuat bar-chat penambahan kasus pekanan yang ditambahkan informasi baru untuk menjawab pertanyaan: “Apakah pekan ini lebih baik?”

Salin dan jalankan kode berikut di konsol Anda!



Adapun hasil yang diharapkan adalah sebagai berikut.

## 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!

## 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()

## Kabar Buruk dan Kabar Baik

Sebagai penutup analisis dalam proyek ini Anda diminta untuk membuat grafik komparasi antara akumulasi kasus aktif, kasus sembuh, dan kasus meninggal.

Ada dua pilihan cara yang dapat Anda tempuh untuk membuat grafik tersebut:

1. Menggunakan data cov\_jabar\_akumulasi, kemudian buat tiga layer geom\_line() untuk setiap kolom satu per satu dengan warna garis yang berbeda; atau

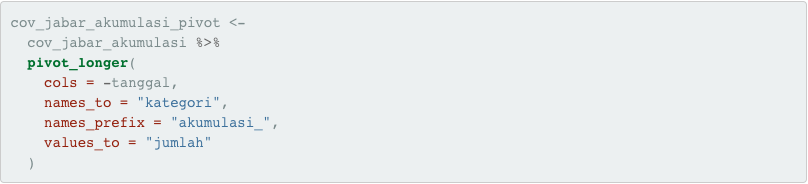
2. Melakukan pivot pada data cov\_jabar\_akumulasi, kemudian buat satu layer geom\_line() dengan menggunakan colour aesthetic

Anda akan diminta untuk menempuh cara nomor 2, yaitu dengan cara melakukan pivot pada data terlebih dahulu. Ketik TRUE pada konsol jika Anda setuju!

## Transformasi Data

Anda akan menggunakan fungsi gather() dari paket tidyr untuk mentransformasi data cov\_jabar\_akumulasi. Data tersebut akan dirubah dari yang semula berformat wide menjadi format long. AKtifkanlah paket tidyr terlebih dahulu dan lengkapilah baris kode berikut untuk melakukan pivot data. Simpan hasil transformasi data tersebut sebagai cov\_jabar\_akumulasi\_pivot! Perhatikan hasil dari fungsi dim() sebelum dan setelah Anda melakukan transformasi data.

## 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\_jabar\_akumulasi menjadi cov\_jabar\_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?