**Menganalisis Data untuk Menjawab Pertanyaan**

**Fungsi dengan beberapa syarat**

Dalam bacaan ini, Anda akan belajar lebih banyak tentang fungsi bersyarat dan bagaimana membangun fungsi dengan beberapa kondisi. Ingat bahwa fungsi dan formula bersyarat melakukan kalkulasi sesuai dengan kondisi tertentu. Sebelumnya, Anda belajar bagaimana menggunakan fungsi seperti SUMIF dan COUNTIF yang memiliki satu kondisi. Anda dapat menggunakan SUMIFS dan COUNTIFS jika Anda memiliki dua atau lebih kondisi. Anda akan mempelajari syntax dasar mereka di Google Sheets, dan mempelajari contohnya.

**SUMIF ke SUMIFS**

Syntax dasar dari fungsi SUMIF adalah: **=SUMIF(range, criterion, sum\_range)**

Range atau rentang pertama adalah di mana fungsi itu mencari kondisi yang telah Anda tetapkan. Criterion adalah kondisi yang Anda terapkan dan sum\_range adalah rentang sel yang akan dimasukkan dalam kalkulasi.

Misalnya, Anda mungkin memiliki tabel yang berisi daftar pengeluaran, biaya, dan tanggal pengeluaran tersebut.



Anda dapat menggunakan SUMIF untuk menghitung total harga bahan bakar dalam tabel, seperti ini:

Formula bar for Cell A11. =SUMIF(A1:A9, "Fuel", B1:B9)

Tapi, Anda juga bisa membuat beberapa kondisi dengan menggunakan fungsi SUMIFS. SUMIF dan SUMIFS sangat mirip, tetapi SUMIFS dapat mencakup beberapa kondisi.

Syntax dasarnya adalah: **=SUMIFS(sum\_range, criteria\_range1, criterion1, [criteria\_range2, criterion2,...])**

Tanda kurung persegi memberi tahu Anda bahwa ini opsional. Tanda kurung tutup pada akhir pernyataan memberitahu bahwa Anda dapat memiliki pengulangan parameter ini sebanyak yang diperlukan. Misalnya, jika Anda ingin menghitung jumlah biaya bahan bakar untuk satu tanggal dalam tabel ini, Anda dapat membuat pernyataan SUMIFS dengan beberapa kondisi, seperti ini:

Formula bar for Cell A12. =SUMIFS(B1:B9, A1:A9, "Fuel", C1:C9, "12/15/2020")

Formula ini memberi Anda biaya total setiap biaya bahan bakar dari tanggal yang tercantum dalam kondisi. Dalam contoh ini, C1:C9 adalah criterion\_range kedua kita dan tanggal 15/12/2020 adalah kondisi kedua. Selama Anda mengikuti syntax dasar, Anda dapat menambahkan hingga 127 kondisi untuk pernyataan SUMIFS!

**COUNTIF ke COUNTIFS**

Sama seperti fungsi SUMIFS, COUNTIFS memungkinkan Anda untuk membuat fungsi COUNTIF dengan beberapa kondisi.

Syntax dasar untuk COUNTIF adalah: **=COUNTIF(range, criterion)**

Sama seperti SUMIF, Anda mengatur rentang dan kemudian kondisi yang perlu dipenuhi. Misalnya, jika Anda ingin menghitung berapa kali Makanan muncul di kolom Pengeluaran, Anda dapat menggunakan fungsi COUNTIF seperti ini:

Formula bar for Cell A13. =COUNTIF(A1:A9, "Food")

COUNTIFS memiliki syntax dasar yang sama dengan SUMIFS:

**=COUNTIFS(criteria\_range1, criterion1, [criteria\_range2, criterion2, ...])**

Criteria\_range dan criterion berada dalam urutan yang sama, dan Anda dapat menambahkan lebih banyak kondisi hingga akhir fungsi tersebut. Jadi, jika Anda ingin menemukan berapa kali Kopi muncul di kolom Pengeluaran pada 15/12/2020, Anda bisa menggunakan COUNTIFS untuk menerapkan kondisi tersebut, seperti ini:

Formula bar for Cell A14. =COUNTIFS(A1:A9, "Coffee", C1:C9, "12/15/2020")

Formula ini mengikuti syntax dasar guna menciptakan kondisi untuk “Kopi” dan tanggal tertentu. Sekarang kita dapat menemukan setiap data di mana kedua kondisi ini terpenuhi.

**Informasi lebih lanjut**

* [Cara menggunakan fungsi Excel IFS](https://exceljet.net/excel-functions/excel-ifs-function)
* [VLOOKUP di Excel dengan beberapa kriteria](https://exceljet.net/formula/vlookup-with-multiple-criteria)
* [INDEX dan MATCH di Excel dengan beberapa kriteria](https://exceljet.net/formula/index-and-match-with-multiple-criteria)
* [Menggunakan IF dengan fungsi AND, OR, dan NO di Excel](https://support.microsoft.com/en-us/office/using-if-with-and-or-and-not-functions-d895f58c-b36c-419e-b1f2-5c193a236d97)

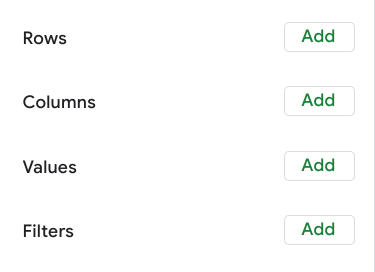
**Fungsi SUMPRODUCT**

SUMPRODUCT adalah salah satu dari sekian banyak trik yang para analis pelajari sendiri atau dapat dari sumber lain. Anda juga bisa menganggapnya sebagai jalan pintas, untuk mengerjakan kalkulasi yang rumit. Kita akan menunjukkan kepada Anda bagaimana SUMPRODUCT bekerja dan kapan Anda mungkin menggunakannya untuk menyederhanakan pekerjaan. SUMPRODUCT adalah fungsi yang mengalikan nilai himpunan dan menghasilkan jumlah dari produk perkalian tersebut.

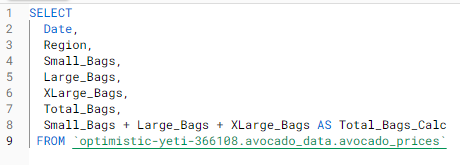


**Elemen tabel pivot**

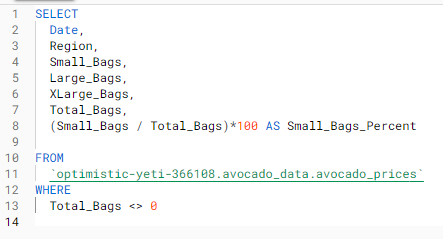
Tabel pivot memungkinkan untuk melihat data dalam berbagai cara untuk mengidentifikasi wawasan dan tren. Mereka dapat membantu Anda dengan cepat memahami dataset atau kumpulan data yang lebih besar dengan membandingkan metrik, melakukan kalkulasi, dan menghasilkan laporan. Mereka juga berguna untuk menjawab pertanyaan spesifik tentang data Anda.

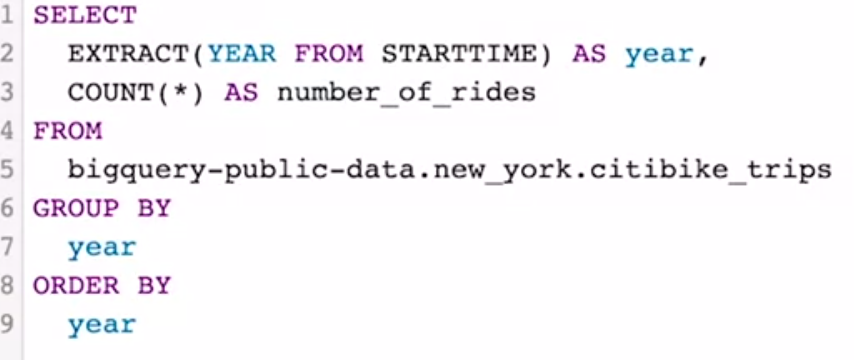


* **Baris tabel pivot** mengatur dan mengelompokkan data yang Anda pilih secara horizontal.
* **Kolom** mengatur dan menampilkan nilai dari data Anda secara vertikal. Mirip dengan baris, kolom dapat ditarik langsung dari dataset atau dibuat dengan menggunakan nilai-nilai. Nilai digunakan untuk menghitung data. Di sinilah Anda memasukkan variabel yang ingin Anda ukur. Inilah cara Anda membuat field yang dihitung, calculated field dalam tabel pivot Anda. Sebagai penyegaran, calculated field adalah field baru dalam tabel pivot yang melakukan kalkulasi tertentu berdasarkan nilai field lainnya.
* **Filter** tabel pivot memungkinkan Anda untuk menerapkan filter berdasarkan kriteria tertentu — seperti halnya filter dalam spreadsheet biasa



Contoh kalkulasi dalam SQL





**Jenis validasi data**

1. **Data Type**

* **Tujuan:** Periksa apakah data cocok dengan jenis data yang ditetapkan untuk suatu field.
* **Contoh**: Nilai data untuk kelas sekolah 1-12 harus berupa jenis data numerik.
* **Keterbatasan**: Nilai data 13 akan melewati validasi jenis data tetapi nilai itu tidak akan dapat diterima. Untuk kasus ini, validasi rentang data juga diperlukan.

1. **Data Range**

* **Tujuan:** Memeriksa apakah data berada dalam rentang nilai yang dapat diterima yang didefinisikan untuk suatu field.
* **Contoh:** Nilai data untuk kelas sekolah harus bernilai antara 1 dan 12.
* **Keterbatasan:** Nilai data 11,5 akan berada di kisaran data dan juga akan diterima sebagai tipe data numerik. Tapi, nilai tidak dapat diterima karena tidak ada nilai setengah. Untuk kasus ini, validasi batasan data juga diperlukan.

1. **Data Constraints**

* **Tujuan:** Memeriksa apakah data memenuhi kondisi atau kriteria tertentu untuk suatu field. Termasuk jenis data yang dimasukkan serta atribut lain dari field tersebut, seperti jumlah karakter.
* **Contoh:** Batasan konten: Nilai data untuk nilai sekolah 1-12 harus berupa bilangan bulat.
* **Keterbatasan**: Nilai data 13 adalah bilangan bulat dan akan melewati validasi batasan konten. Tapi, nilai itu tidak dapat diterima karena 13 bukan kelas sekolah yang diakui. Untuk kasus ini, validasi rentang data juga diperlukan.

1. **Data Consistency**

* **Tujuan**: Memeriksa apakah data dapat diterima dalam konteks data lainnya yang terkait.
* **Contoh**: Nilai data untuk tanggal pengiriman produk tidak bisa lebih awal dari tanggal produksi produk.
* **Keterbatasan**: Data mungkin konsisten tetapi masih bisa salah atau tidak akurat. Tanggal pengiriman bisa lebih lambat dari tanggal produksi dan masih bisa salah.

1. **Data Structure**

* **Tujuan**: Memeriksa apakah data mengikuti atau sesuai dengan struktur yang ditentukan.
* **Contoh**: Laman web harus mengikuti struktur yang ditentukan untuk bisa ditampilkan dengan benar.
* **Keterbatasan**: Struktur data mungkin benar tetapi datanya masih salah atau tidak akurat. Konten pada laman web dapat saja ditampilkan dengan benar namun masih berisi informasi yang salah.

1. **Code Validation**

* **Tujuan**: Memeriksa apakah kode aplikasi secara sistematis melakukan salah satu validasi yang disebutkan sebelumnya ketika data dimasukkan pengguna.
* **Contoh**: Masalah umum yang ditemukan selama validasi kode meliputi: lebih dari satu jenis data yang diperbolehkan, pemeriksaan rentang data tidak dilakukan, atau akhir string teks tidak didefinisikan dengan baik.
* **Keterbatasan**: Validasi kode mungkin tidak memvalidasi semua kemungkinan variasi dengan masukan data.

**Temporary Tabel SQL**

Tabel sementara persis seperti apa yang terdengar - tabel sementara dalam database SQL yang tidak disimpan secara permanen. Dalam bacaan ini, Anda akan mempelajari metode untuk membuat tabel sementara menggunakan perintah SQL. Anda juga akan mempelajari beberapa praktik terbaik untuk diikuti saat bekerja dengan tabel sementara.

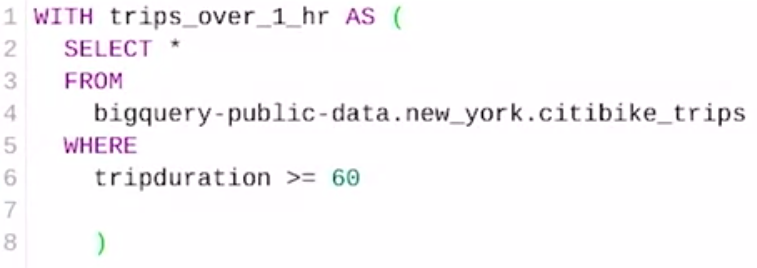
* Tabel sementara secara otomatis dihapus dari database ketika Anda mengakhiri sesi SQL Anda.
* Mereka dapat digunakan sebagai holding area untuk menyimpan nilai-nilai jika Anda membuat serangkaian kalkulasi. Hal ini kadang-kadang disebut sebagai pra-pengolahan data.
* Mereka dapat mengumpulkan hasil dari beberapa kueri terpisah. Hal ini kadang-kadang disebut sebagai data staging. Staging berguna jika Anda perlu melakukan kueri pada data yang dikumpulkan atau menggabungkan data yang dikumpulkan.
* Mereka dapat menyimpan subset yang disaring dari database. Anda tidak perlu memilih dan memfilter data setiap kali Anda bekerja dengan subset itu. Selain itu, menggunakan perintah SQL yang lebih sedikit membantu menjaga data Anda tetap bersih.

**Temp Tabel Dengan Syntax WITH**

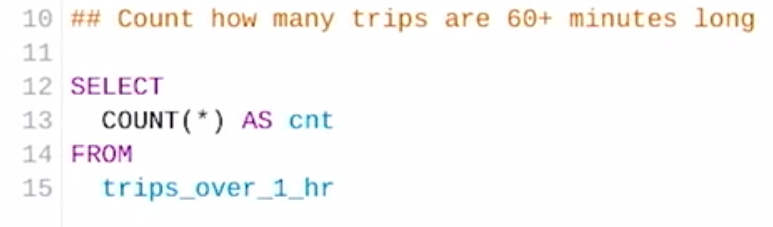
Tabel sementara dapat dibuat dengan menggunakan klausa yang berbeda. Dalam BigQuery, klausa WITH dapat digunakan untuk membuat tabel sementara. Syntax umum untuk metode ini adalah sebagai berikut:

Mengurai permintaan ini sedikit, perhatikan hal berikut:

* Pernyataan dimulai dengan klausa WITH diikuti dengan nama tabel sementara baru yang ingin Anda buat
* Klausa AS muncul setelah nama tabel baru. Klausa ini menginstruksikan database untuk menempatkan semua data yang diidentifikasi di bagian berikutnya dari pernyataan tersebut ke dalam tabel baru.
* Tanda kurung pembukaan setelah klausa AS menciptakan sub-kueri yang menyaring data dari tabel yang ada. Sub-kuerinya adalah pernyataan SELECT biasa bersama dengan klausa WHERE untuk menentukan data yang akan disaring.
* Kurung penutup mengakhiri sub-kueri yang dibuat oleh klausa AS.



Contoh Tabel temporary menggunakan WITH



Contoh penggunaan pada query baru

**Temp Tabel Dengan Klausa SELECT dan INTO**

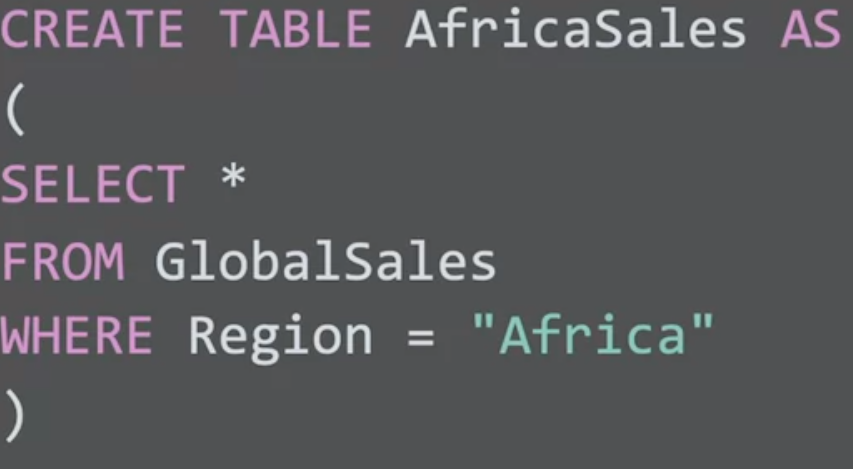
Metode berikut tidak didukung di BigQuery, tetapi sebagian besar versi database SQL lainnya mendukungnya, termasuk SQL Server dan MySQL. Menggunakan SELECT dan INTO, Anda dapat membuat tabel sementara berdasarkan kondisi yang didefinisikan oleh klausa WHERE untuk menemukan informasi yang Anda butuhkan untuk tabel sementara. Syntax umum untuk metode ini adalah sebagai berikut:



Pernyataan SELECT ini menggunakan klausa standar seperti FROM dan WHERE , tetapi klausa INTO memberitahu database untuk menyimpan data yang diminta dalam tabel sementara baru bernama, dalam hal ini, “AfricaSales.”

**Temp Tabel Dengan Klausa CREATE TABLE**

Sejauh ini, kita telah mengeksplorasi cara-cara membuat tabel sementara di mana database bertanggung jawab untuk mengelola. Tapi, Anda juga dapat membuat tabel sementara yang dapat dikelola sebagai pengguna. Sebagai seorang analis, Anda mungkin memutuskan untuk membuat tabel sementara untuk analisis Anda yang dapat Anda kelola sendiri. Anda akan menggunakan pernyataan CREATE TABLE untuk membuat semacam tabel sementara. Setelah Anda selesai bekerja dengan tabel itu, Anda kemudian akan menghapus atau melepaskannya dari database di akhir sesi Anda.



Setelah Anda selesai bekerja dengan tabel sementara Anda, Anda dapat menghapus tabel dari database menggunakan klausa DROP TABLE. Syntax umumnya adalah sebagai berikut:

