# Fundamental SQL Using SELECT Statement

## Chapter 1

### Apa itu SQL?

SQL yang merupakan singkatan dari Structured Query Language, yaitu bahasa komputer standar yang digunakan untuk berinteraksi dengan suatu sistem database - atau lebih tepatnya sistem manajemen database relasional. Jadi, user dapat menambahkan, mengubah, mengupdate, mencari dan menghapus data dari suatu sistem database dengan menggunakan SQL.

SQL dilafalkan dengan membaca tiap karakternya S Q L (es kiu el) atau sikuel.

### Contoh komunikasi SQL

Terdapat dua kategori dari interaksi SQL:

* **Data Definition Language** (DDL), yaitu berbagai perintah yang berfungsi lebih kepada memanipulasi struktur database, seperti Membuat (CREATE), meubah (ALTER), dan menghapus (DROP) struktur penyimpanan data, yaitu database, table, kolom dan tipe data.
* **Data Manipulation Language** (DML), yaitu berbagai perintah yang digunakan untuk Menyisipkan data (INSERT), Mengambil data atau query (SELECT), Meubah data (UPDATE) dan Menghapus data (DELETE).

Berikut adalah satu contoh query atau perintah untuk mengambil data:

**SELECT** nama\_produk **FROM** ms\_produk;

nama\_produk                          
------------------------------------  
Kotak Pensil DQLab                   
Flashdisk DQLab 64 GB                
Gift Voucher DQLab 100rb             
Flashdisk DQLab 32 GB                
Gift Voucher DQLab 250rb             
Pulpen Multifunction + Laser DQLab   
Tas Travel Organizer DQLab           
Gantungan Kunci DQLab                
Buku Planner Agenda DQLab            
Sticky Notes DQLab 500 sheets

Terlihat ada sepuluh nama peralatan kantor dengan label DQLab yang ditampilkan. Ini menunjukkan bahwa interaksi antara SQL dan sistem database telah berjalan dengan baik.

“Nih coba kamu baca. Selain yang tadi kujelaskan, penguasaan SQL **akan membantu perusahaan pada area berikut,**” tunjuk Senja pada salah satu halaman buku catatannya:

* Manajemen memerlukan laporan dengan informasi yang semakin beragam, seperti: tren penjualan bulan ke bulan, pertumbuhan pelanggan, apakah perusahaan mencapai target, dan lain-lain. Dan ini membutuhkan keahlian SQL yang mumpuni.
* Programmer yang membangun sistem aplikasi hampir dipastikan selalu bergantung kepada sistem database SQL agar aplikasinya berjalan dengan baik. Dengan demikian, penguasaan SQL adalah hal mutlak.
* Bisa meningkatkan kinerja perusahaan karena informasi yang kaya dapat dihasilkan melalui SQL.

“Jadi, untuk beberapa alasan inilah, maka tidak heran SQL menjadi keterampilan utama yang diminta oleh banyak perusahaan?” ujarku menyimpulkan.

### Dimana saja SQL Digunakan?

Perusahaan – perusahaan yang sudah menerapkan sistem IT pasti memiliki sistem database dan bisa dipastikan menyimpan datanya dalam suatu database. Contohnya perusahaan berbasis teknologi, seperti e-commerce, menyimpan data baik itu data profile user, data transaksi pembelian dan penjualan, data produk dan data traffic kunjungan user ke halaman website di sistem database - atau lebih tepatnya sistem manajemen database atau database management system (DBMS).

Semua informasi ataupun analisa yang dibutuhkan oleh manajemen, umumnya bersumber dan diolah dari data DBMS ini. Dan di perusahaan, sistem database biasanya tidak hanya satu, bisa dua, tiga bahkan puluhan. Oleh karena itu, SQL sangat berperan disini, karena dengan menggunakan SQL dapat memenuhi kebutuhan manajemen tersebut. Tanpa penguasaan SQL  akan kesulitan memperoleh data yang dibutuhkan, dan akan kesulitan dalam melakukan analisa dan menghasilkan informasi yang dibutuhkan manajemen dan perusahaan.

Akan tetapi, perlu diketahui bahwa tidak semua sistem database mendukung SQL. Hanya sistem database berbasis relational database management system (RDBMS) yang mendukung bahasa ini. Untuk RDBMS sendiri akan dijelaskan kemudian.

**SQL hanya digunakan di sistem database berbasis Relational Database Management System.**

### Kesimpulan

SQL singkatan dari Structured Query Language, adalah sebuah **bahasa komputer** sederhana yang menjadi standar untuk memungkinkan seseorang berkomunikasi dengan suatu **sistem database manajemen relasional (RDBMS)**.

Karena RDBMS bisa dikatakan digunakan oleh sistem IT oleh seluruh perusahaan di dunia dan di Indonesia, maka pengetahuan SQL menjadi aset yang penting di perusahaan. Walaupun sederhana, aspek SQL sangat luas.

Aku semangat sekali, karena pada module Fundamental SQL using SELECT statement, aku akan mempelajari proses pengambilan data dengan pendekatan best practice yang dibutuhkan oleh mayoritas perusahaan di dunia dan Indonesia.

## Chapter 2

### Apa itu RDBMS?

**R**elational **D**atabase **M**anagement **S**ystem yang biasa disingkat dengan RDMBS adalah suatu program yang memungkinkan untuk Membuat, Memperbarui, dan Mengelola suatu basis data relasional (Relational Database). Nah, Umumnya RDMBS ini menggunakan SQL untuk mengakses database.

Basis data relasional sendiri merupakan suatu jenis database dimana data – data umumnya disimpan dalam bentuk yang terstruktur berupa tabel (baris dan kolom) dan setiap tabel/ data yang terdapat dalam database memiliki relasi (relational) satu sama lain. Seperti terlihat pada gambar berikut

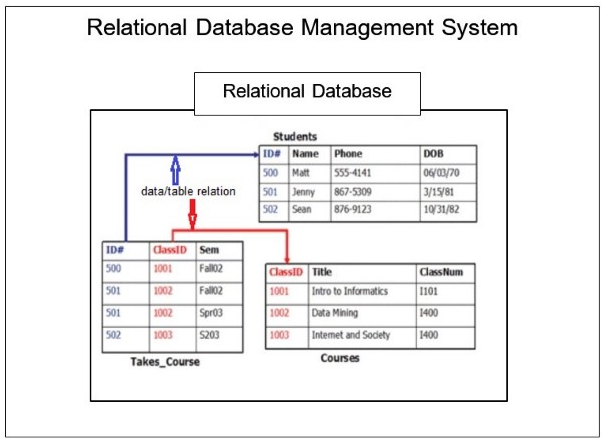


Image 1 RDBMS

Basis data relasional sangat popular dan banyak digunakan oleh perusahaan – perusahaan karena jenis database ini mudah dikelola terlebih jika memiliki banyak data atau informasi yang perlu disimpan, scalable dan flexibel.

* Basis data rasional cukup mudah dikelola. Setiap tabel/data dapat diupdate atau dimodifikasi tanpa mengganggu tabel/data yang lain.
* Flexible : jika perlu memperbarui data, hanya perlu melakukannya sekali saja - jadi tidak perlu lagi mengubah banyak file satu per satu. Selain itu, basis data rasional juga cukup mudah untuk di-extend. Misalnya saat data sudah semakin banyak, dapat dengan mudah memperbesar kapasitas dari database yang dimiliki.

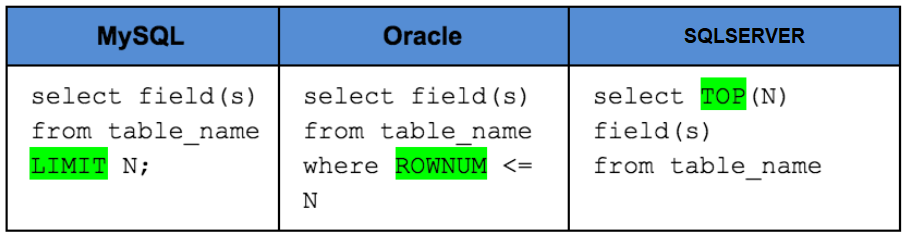
### Produk-produk RDBMS di Pasaran

Selain MySQL, masih ada produk lain RDBMS, baik yang berbayar (proprietary) maupun open source. Berikut adalah sebagian produk yang cukup populer di pasaran :

1. **MySQL**Open-source SQL database yang cukup populer. Umumnya digunakan untuk pengembangan aplikasi web.
2. **PostgreSQL**Open-source RDBMS product, dan juga umumnya digunakan untuk pengembangan aplikasi web. Akan tetapi secara kinerja, postgreSQL lebih lambat dibandingkan MySQL.
3. **Oracle DB**Produk RDBMS yang dimiliki oleh Oracle Corporation dan produk ini bersifat proprietary atau tidak open source. Oracle DB umumnya digunakan di industri perbankan.
4. **Microsoft SQL Server**   
   SQL Server adalah produk RDBMS yang dimiliki oleh Microsoft dan sama seperti Oracle DB, SQL Server bersifat proprietary atau tidak open source, SQL Server umumnya digunakan di perusahaan skala besar yang juga menggunakan produk keluaran Microsoft lainnya.
5. **SQLite**Open source RDBMS, umumnya digunakan sebagai database di handphone, MP3 player, and perangkat lainnya.

Selain itu, juga ada MariaDB yang juga gratis atau open source, IBM DB2, Microsoft Access, dan masih banyak lainnya.

Umumnya RDBMS menggunakan SQL untuk mengakses database dan produk RDBMS tidak hanya satu macam saja tetapi ada berbagai macam produk, maka SQL syntax pun bisa jadi sedikit berbeda untuk setiap produk tersebut. Berikut contoh perbandingan MySQL, Oracle, dan SQLSERVER untuk menampilkan beberapa baris data dari suatu tabel :



### Struktur Penyimpanan RDBMS

Sebagai penyimpan data, sistem database relasional memiliki struktur hirarki objek penyimpanan sebagai berikut:

* Database
* Tabel (table)
* Kolom (column) atau Field

Dari sini aku belajar informasi menarik yaitu setiap database bisa berisi beberapa tabel, dan setiap tabel bisa terdiri dari beberapa kolom. Di setiap database, tabel dan kolom memiliki nama sendiri sebagai identitas mereka. Tabel dan kolom inilah yang akan diisi data yang kemudian membentuk row (baris data).

### Tabel dan Kolom

Gambar berikut adalah contoh suatu Tabel dalam database. Karena setiap tabel dalam database memiliki nama, maka, nama tabel ini adalah **ms\_produk**.

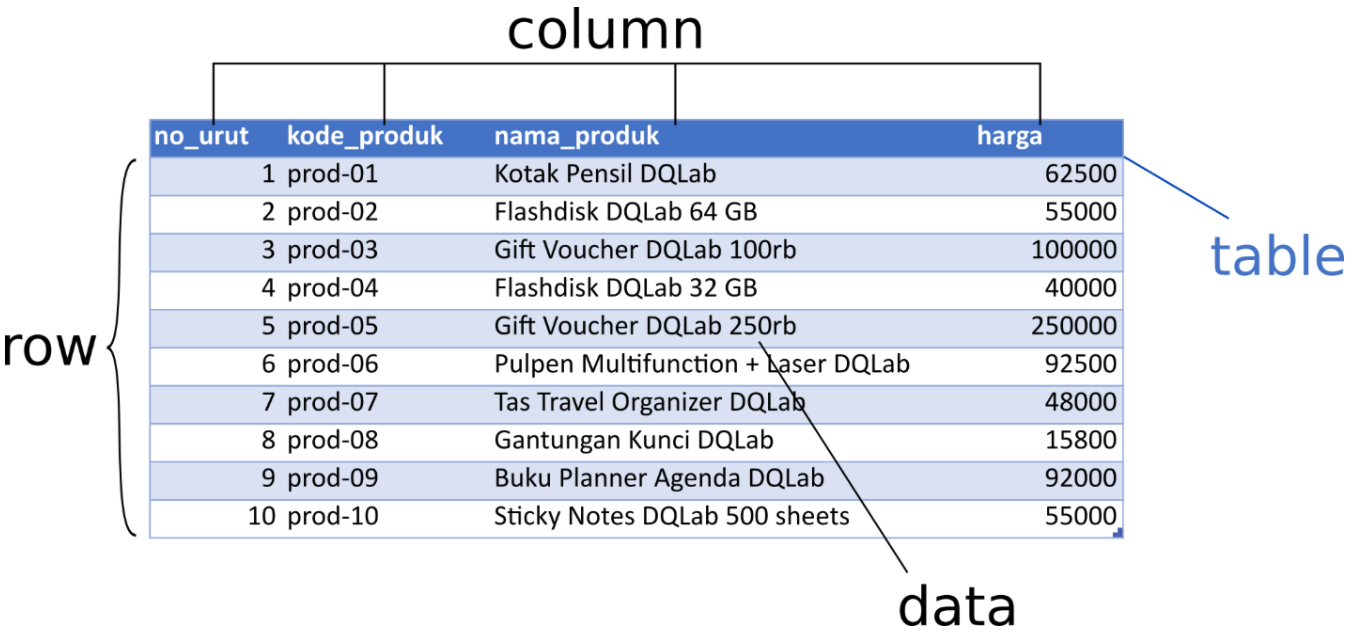


Image 2 Tabel ms\_produk

Jika aku perhatikan, struktur tabel **ms\_produk** terdiri dari empat kolom (column), masing-masing dengan nama berikut:

* no\_urut
* kode\_produk
* nama\_produk
* harga

Dan dalam tabel tersebut terdapat 10 baris data (row) dengan isi data yang bervariasi, contoh isi data untuk kolom "**nama\_produk**" pada baris kelima adalah "**Gift Voucher DQLab 250rb**".

### Kesimpulan

Wah ternyata seru sekali belajar SQL!

Aku mulai membuka catataanku, mengambil pena, dan menuliskan apa yang aku pelajari:

1. Sistem relasional database atau relational database management system (RDBMS) adalah sistem database paling populer di dunia saat ini dan menggunakan bahasa SQL untuk pengolahannya.
2. Beberapa produk RDBMS yang terkenal antara lain adalah Microsoft SQL Server, Oracle, MySQL, PostgreSQL, IBM DB2, dan masih banyak lainnya.
3. Struktur penyimpanan data di RDBMS menggunakan hirarki:
   1. Database
   2. Tabel (Table)
   3. Kolom (Column)
4. Data akan diisi ke dalam table dalam bentuk Baris (Row) data

## Chapter 3

### Mengambil Seluruh Kolom dalam suatu Tabel

“Untuk mengakses data di database, kita dapat menggunakan SELECT statement. Pada SELECT statement kita menyatakan kolom - kolom mana saja yang ingin kita tampilkan dari suatu tabel di database. SELECT statement tidak berdiri sendiri. Setelah menyatakan kolom - kolom yang ingin ditampilkan, kita melanjutkan dengan FROM. Di FROM inilah kita menyatakan dari tabel mana data yang ingin kita tampilkan berada. **SELECT… FROM…** adalah statement paling sederhana di SQL, dan merupakan bagian utama dari query. Kita tidak bisa meng-query data tanpa menggunakan statement ini,” jelas Senja.



Image 3 Statement SELECT

* Kata awal, yaitu **SELECT** digunakan untuk menginformasikan kepada sistem bahwa kita ingin mengambil data.
* Tanda **\*** (bintang) artinya seluruh kolom perlu diambil dari tabel yang dirujuk. Tanda ini sering juga disebut sebagai **wildcard**.
* **FROM [NAMA\_TABLE]**, artinya table yang akan diambil datanya.
* Tanda **;** (titik koma) adalah tanda yang menyatakan akhir dari perintah SELECT atau SQL lain.

#### Tugas :

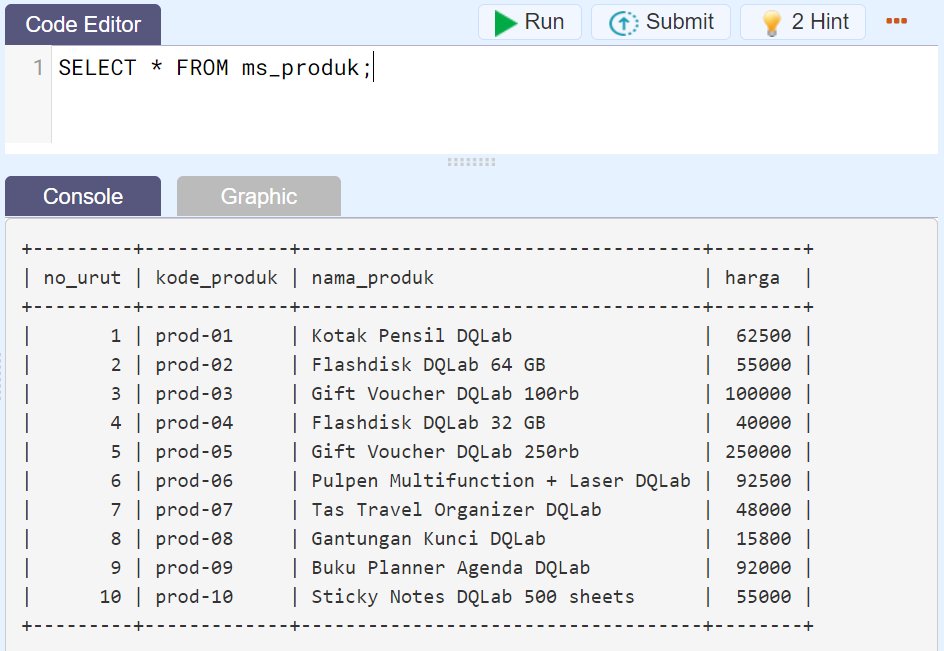


Image 4 Run SELECT \* (All Kolom)

### Mengambil Satu Kolom dari Tabel

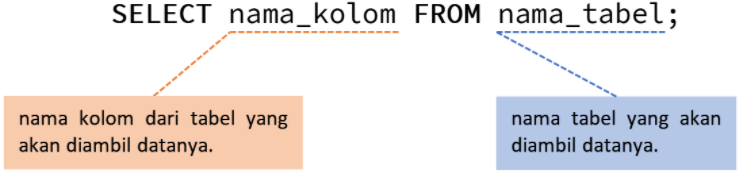


Image 5 Statement Select 1 Kolom

“Kita coba ya dengan menampilkan data pelanggan yang ada di database. Kita sudah menggunakan perintah **SELECT** sebelumnya untuk mengambil seluruh kolom. Nah, berikut adalah contoh query untuk mengambil satu kolom saja yaitu **nama\_produk**,” tambah Senja.

#### Tugas :

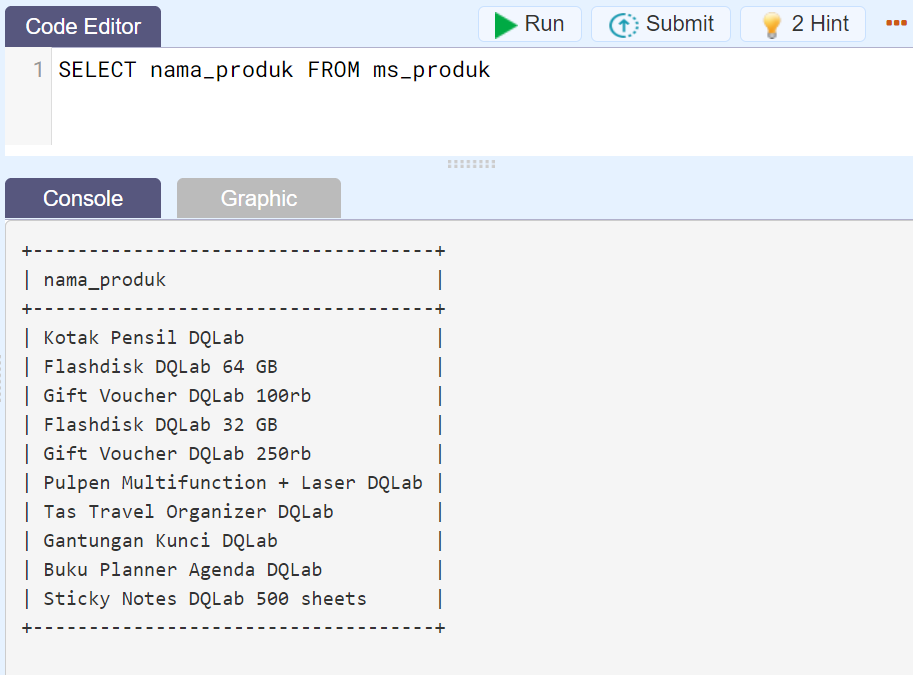


Image 6 Run Select 1 Kolom

### Mengambil Lebih dari Satu Kolom dari Tabel

Tabel **ms\_produk** memiliki lebih dari satu kolom data. Kalau aku ingin mengambil kolom lainnya, aku hanya perlu menuliskan tiap kolom yang ingin ditampilkan dipisahkan dengan tanda koma, seperti contoh berikut untuk dua kolom.

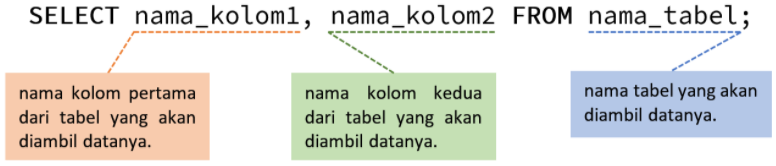


Image 7 SELECT Lebih dari Satu Kolom dari Tabel

Menggunakan tabel **ms\_produk**, aku menggunakan perintah SELECT berikut untuk menampilkan dua kolom, **kode\_produk** dan **nama\_produk**.

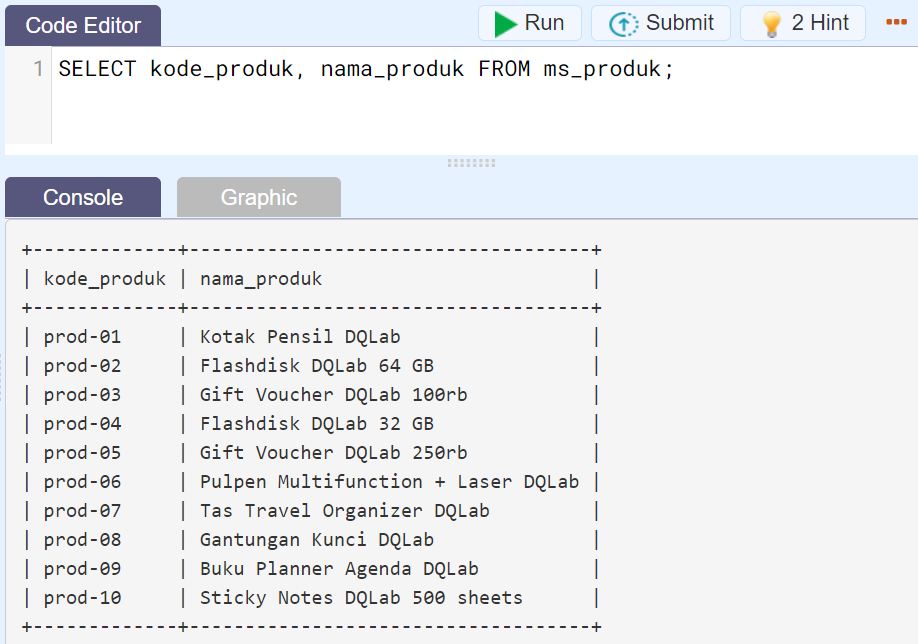


Image 8 Hasil Select lebih dari 1 Kolom terhadap 1 Tabel

**Ingat:**

**Untuk MySQL nama kolom sensitif terhadap huruf besar maupun kecil.**

#### Tugas:

Sekarang gantilah perintah SELECT di code editor untuk menampilkan nama\_produk dan harga dari tabel yang sama. **Ingat** untuk memisahkan setiap kolom dengan comma (,).

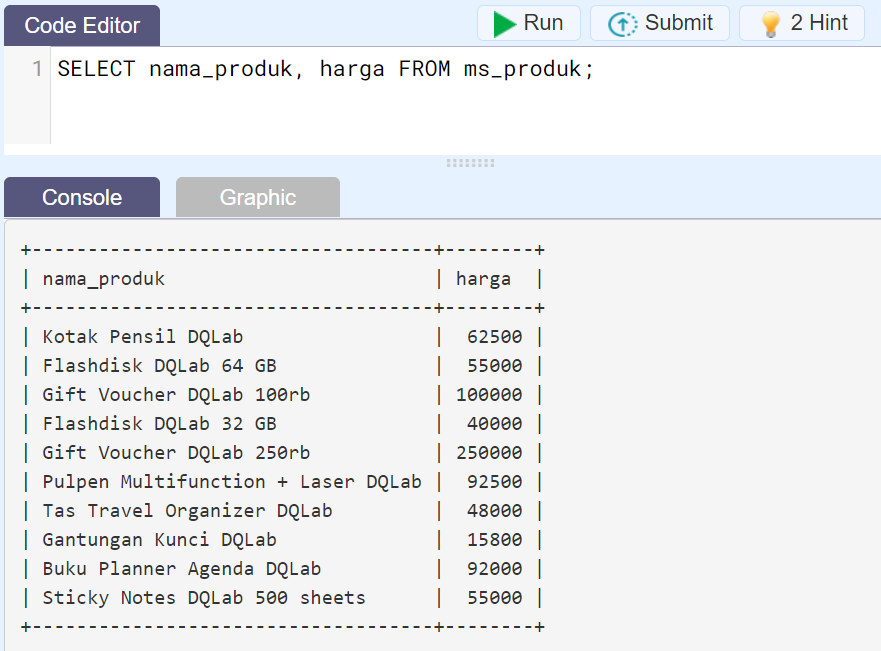


Image 9 Hasil Tugas

### Membatasi Pengambilan Jumlah Row Data

Selain pembatasan kolom, aku bisa membatasi jumlah baris data yang diambil. Seperti yang aku pelajari di materi RDMS sebelumnya, bahwa untuk tiap produk RDBMS, caranya agak berbeda-beda. Untuk MySQL dan PostgreSQL, aku dapat menggunakan LIMIT. Secara umum syntaxnya dinyatakan sebagai berikut

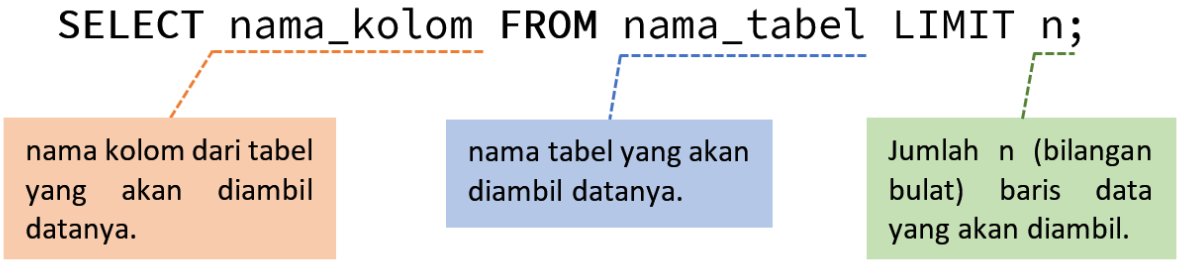


Image 10 Statement membatasi Select

#### Tugas :

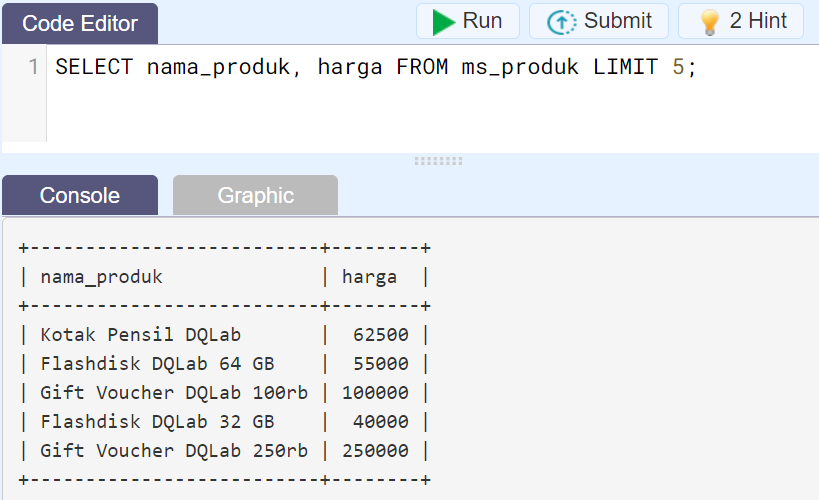


Image 11 Hasil Limit

### Penggunaan SELECT DISTINCT statement

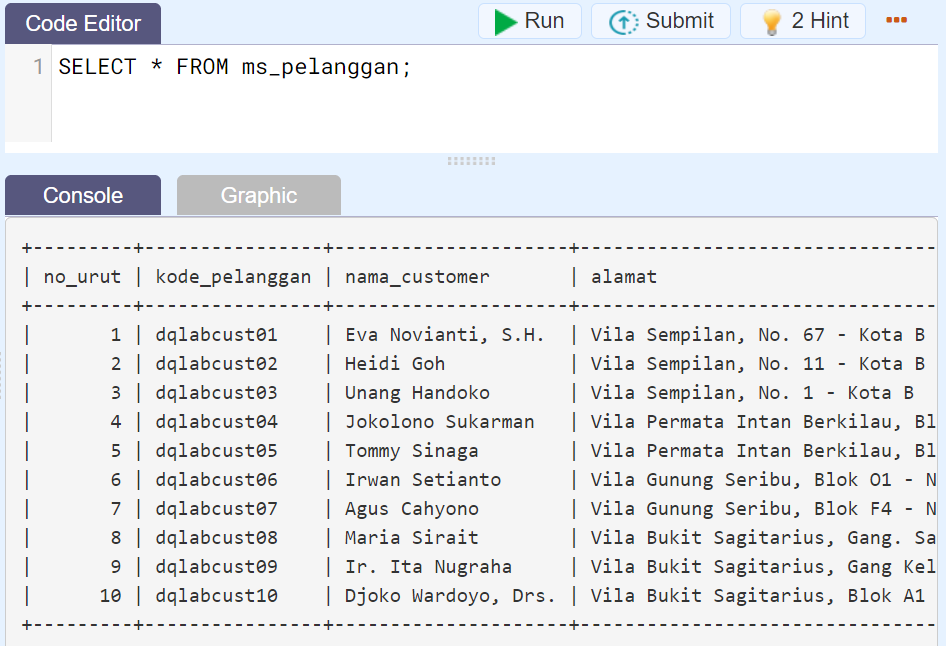


Image 12 Tabel ms\_pelanggan

Ternyata, dari data pelanggan, aku menemukan duplikasi data, dalam nama\_customer dan alamat untuk no\_urut 3 & 11, serta 5 & 12 yang sama persis dengan kode\_pelanggan yang berbeda. Tentunya ini akan berdampak pada hasil analisaku nantinya.

Untuk menghilangkan data duplikasi, aku bisa menggunakan **SELECT DISTINCT** statement. Dengan **SELECT DISTINCT**, data yang sama atau duplikat akan dieliminasi dan akan ditampilkan data yang unik saja.

Berikut syntax-nya:

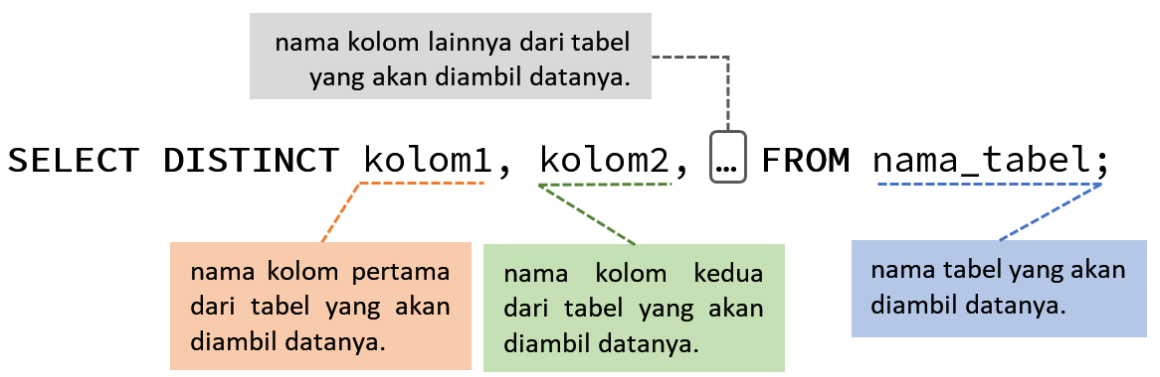


Image 13 Syntax DISTINCT

#### Tugas :

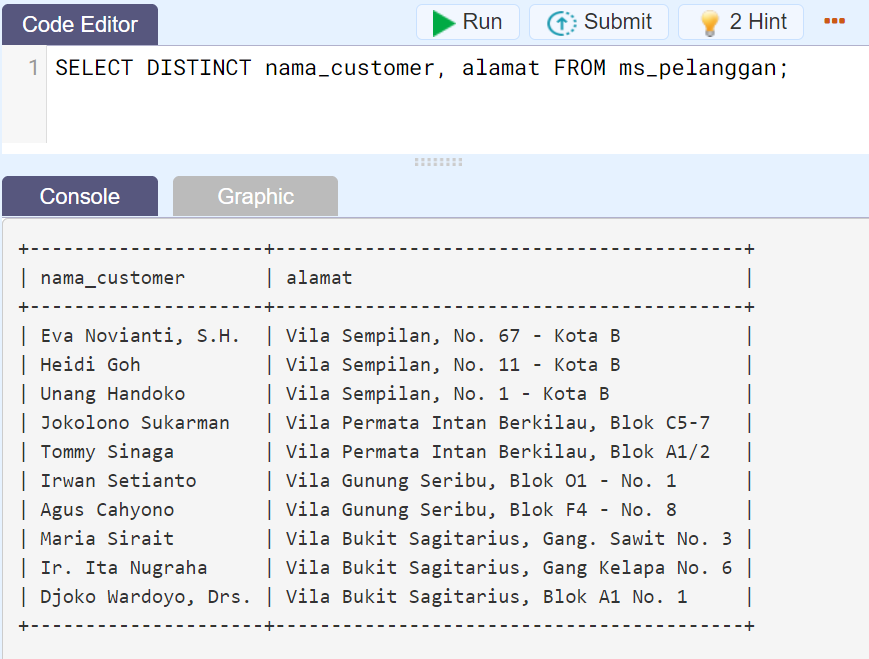


Image 14 Hasil Distinct

### Kesimpulan

Aku mengambil catatanku, dan mulai menulis apa yang aku pelajari, sebelum aku melanjutkan belajarku:

1. Perintah SELECT dapat digunakan untuk menentukan apa saja kolom yang akan diambil dengan menuliskan nama-nama kolom yang diinginkan menggunakan pemisah tanda koma.
2. Perintah SELECT juga dapat digunakan untuk membatasi jumlah data yang dikeluarkan. Namun untuk berbagai produk bisa berbeda penulisannya. MySQL menggunakan LIMIT untuk tujuan tersebut.
3. Perintah SELECT DISTINCT dapat digunakan untuk menghilangkan duplikasi baris dalam tabel dan hanya menampilkan baris data yang unik tanpa duplikasi.

## Chapter 4

### Pendahuluan

Sejauh ini, aku sudah cukup paham cara menggunakan SQL dengan perintah SELECT, terutama untuk mengakses data dari database. Aku pun berniat untuk mengambil dan menganalisis data produk dengan syntax yang ada:

SELECT t1.kode\_produk AS product\_code, t1.nama\_produk AS product\_name, t1.harga AS price FROM ms\_produk AS t1;

Beberapa detik aku termenung melihat syntax ini. Mengapa ada **‘t1’,** dan **‘AS’?** Apa maksudnya? Aku terdorong untuk bertanya kembali pada Senja.

“Nja, sorry gangguin kamu lagi. Aku lagi mau nyoba latihan nih. Tapi aku nemu syntax ini dan agak bingung terutama karena ada ‘t1’, dan ‘AS’. Maksudnya?”

“Oh, ini hanya variasi penggunaan nama table dan kolom pada bagian SELECT. Sini saya kasih tahu.”

Aku pun mencatat penjelasan Senja, walau sederhana tapi penting:

* **Prefix**, dimana kita akan menambahkan nama tabel di depan nama kolom.
* **Alias**, dimana kita memberikan alias atau nama lain untuk tabel maupun kolom.

### Menggunakan Prefix pada Nama Kolom

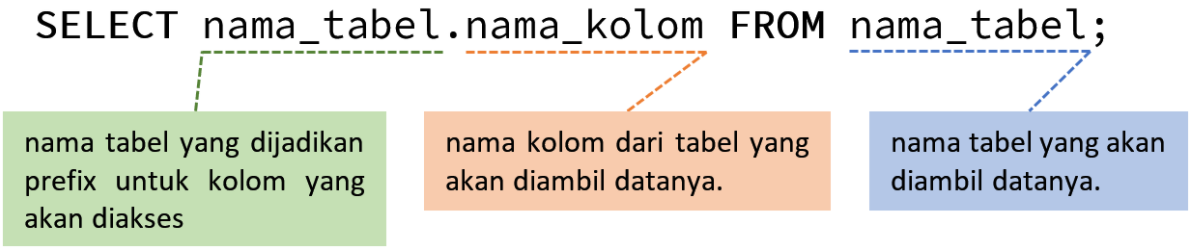


Image 15 Syntax Prefix

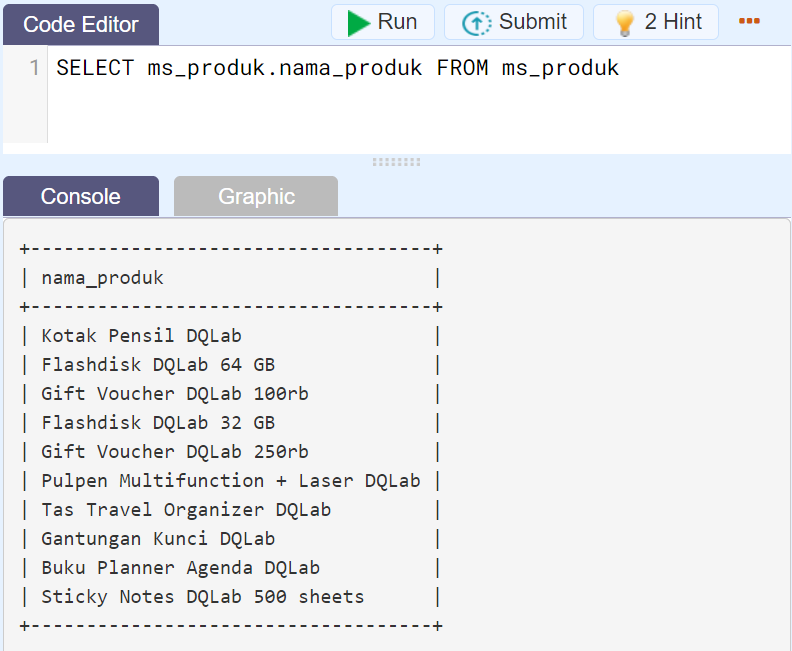


Image 16 Contoh prefix jika dari 2 table join agar tidak ambigu

#### Tugas :

Gantilah seluruh code yang ada pada code editor dengan perintah SELECT untuk menampilkan kolom **kode\_produk** dari tabel **ms\_produk** dengan penulisan menggunakan prefix nama tabel.

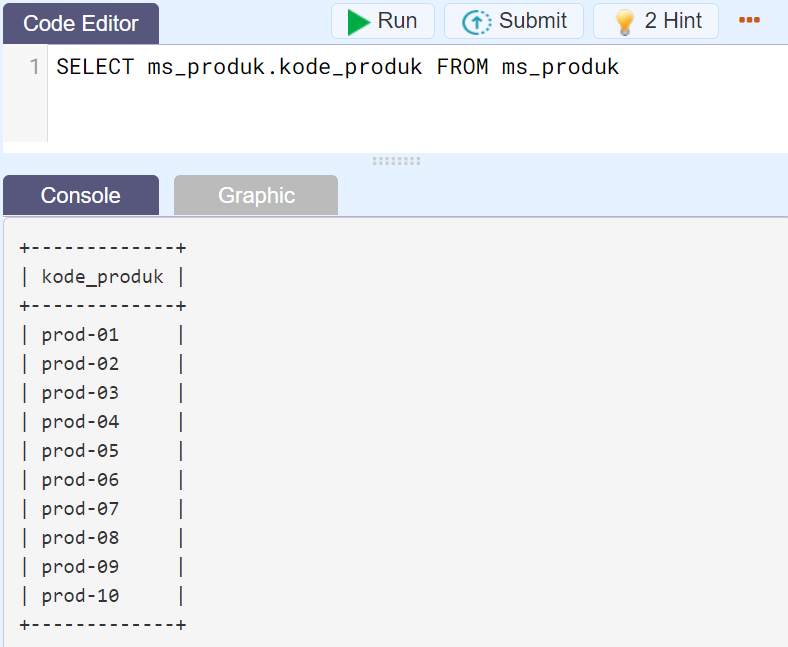


Image 17 Hasil Tugas

### Menggunakan Alias pada Kolom

Selain prefix, aku dapat merubah identitas nama kolom yang diambil dengan SELECT dengan menggunakan keyword **AS**. Ini dinamakan alias. Perubahan nama tabel bersifat temporary, artinya hanya berubah ketika mengambil/meng-query data, sedangkan nama kolom di tabel dalam database tidak akan berubah.

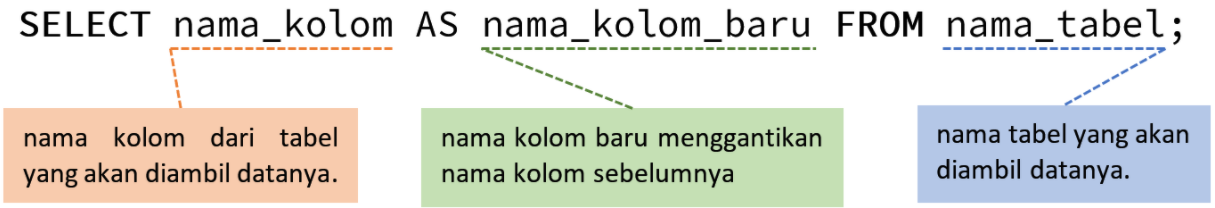


Image 18 Syntax Alias

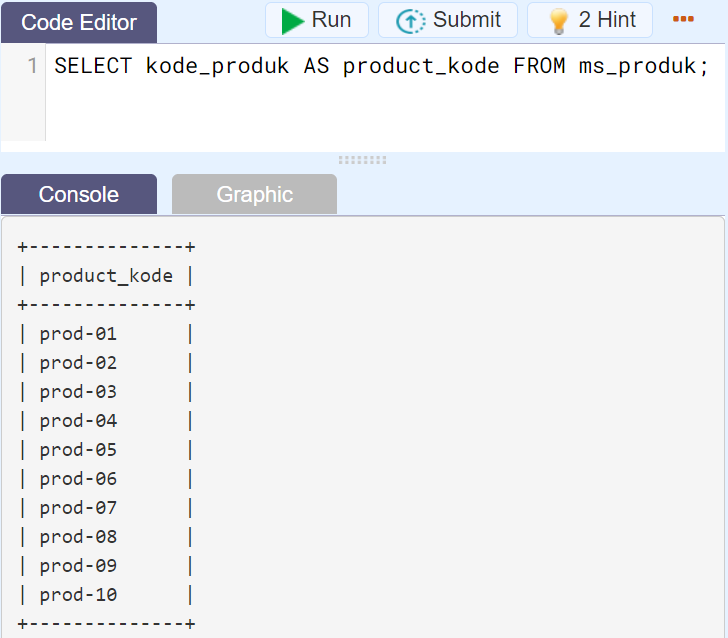


Image 19 Contoh Alias

Terlihat kolom **kode\_produk** telah berubah nama menjadi **product\_code**.

**Catatan:** **alias tidak bisa digunakan untuk wildcard (\*)**

#### Tugas :

Cobalah ubah perintah SELECT tersebut di atas untuk merubah nama kolom dengan details berikut:

* **no\_urut** menjadi **nomor**.
* **nama\_produk** menjadi **nama**.

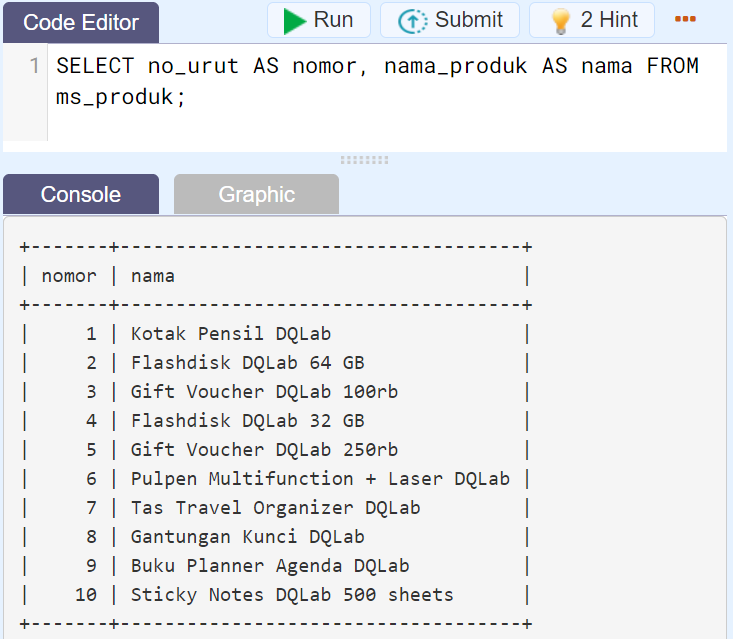


Image 20 Hasil Tugas

### Menghilangkan Keyword 'AS'

Keyword **AS** yang digunakan sebagai penanda alias pada kolom dapat dihilangkan dengan syntax:

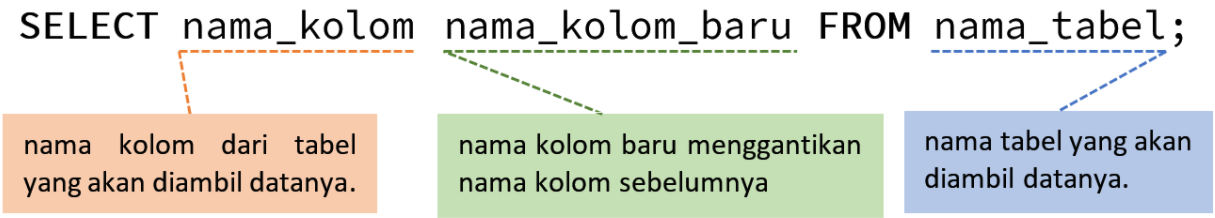


Image 21 Syntax menghilangkan 'AS'

Berikut adalah contoh yang sama dari sub-chapter sebelumnya, dimana untuk merubah nama kolom dari **kode\_produk** menjadi **product\_code** dari tabel ms\_produk dapat dilakukan tanpa menggunakan alias.

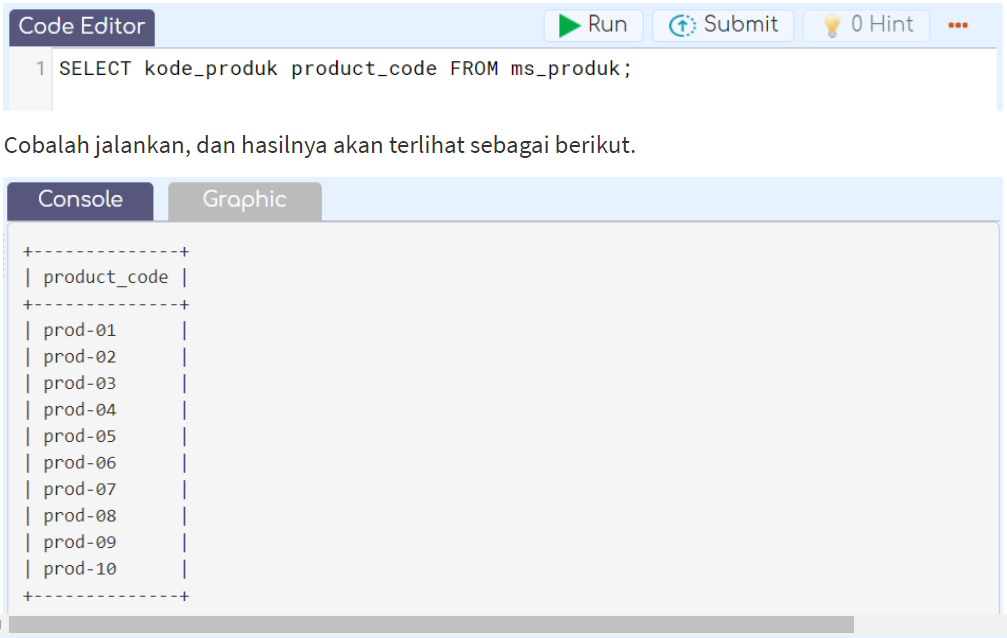


Image 22 Contoh tanpa 'AS'

#### Tugas :

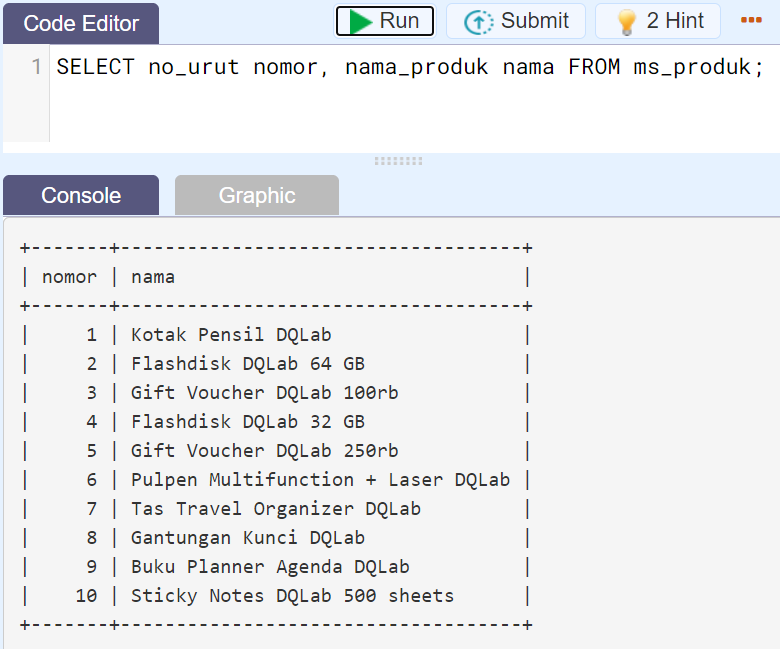


Image 23 Hasil Tugas

### Menggabungkan Prefix dan Alias

Prefix dan alias juga dapat digunakan secara bersamaan.

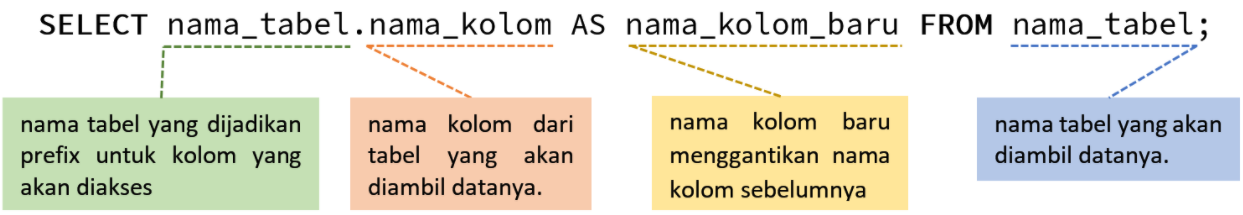


Image 24 Syntax HYBRID (PREFIX & ALIAS)

Aku menerapkannya dengan tabel ms\_produk, menggunakan prefix nama tabel dan alias untuk merubah nama\_produk menjadi nama.

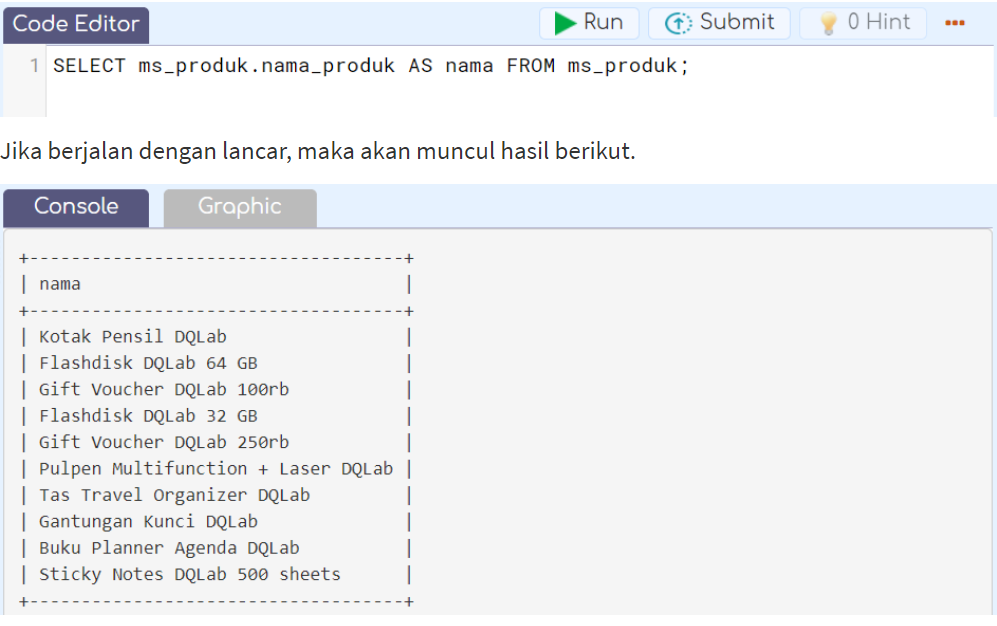


Image 25 Contoh Hybrid

#### Tugas :

Tampilkan kolom **harga** dari tabel **ms\_produk** dengan nama alias **harga\_jual** lengkap dengan prefix.

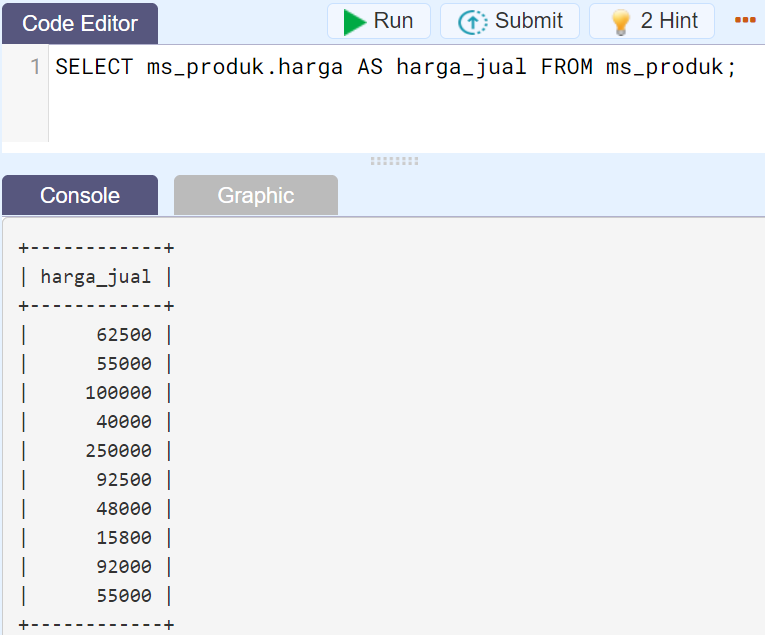


Image 26 Hasil Tugas

### Menggunakan Alias pada Tabel

Selain kolom, nama alias juga bisa digunakan untuk tabel dengan menggunakan keyword **AS** setelah nama tabel. Dan, keyword ini juga bisa digunakan atau tidak. Umumnya penggunaan alias pada tabel jika nama tabel tersebut cukup panjang dan muncul atau dirujuk beberapa kali dalam query. Sehingga dengan menggunakan alias pada tabel, dapat menghemat waktu dalam menuliskan query, khususnya untuk query yang cukup rumit, panjang dan melibatkan banyak tabel.

Penulisannya adalah sebagai berikut.

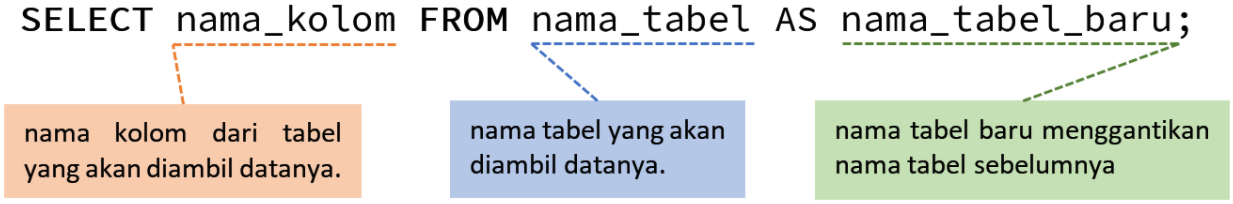


Image 27 Syntax Alias pada Tabel

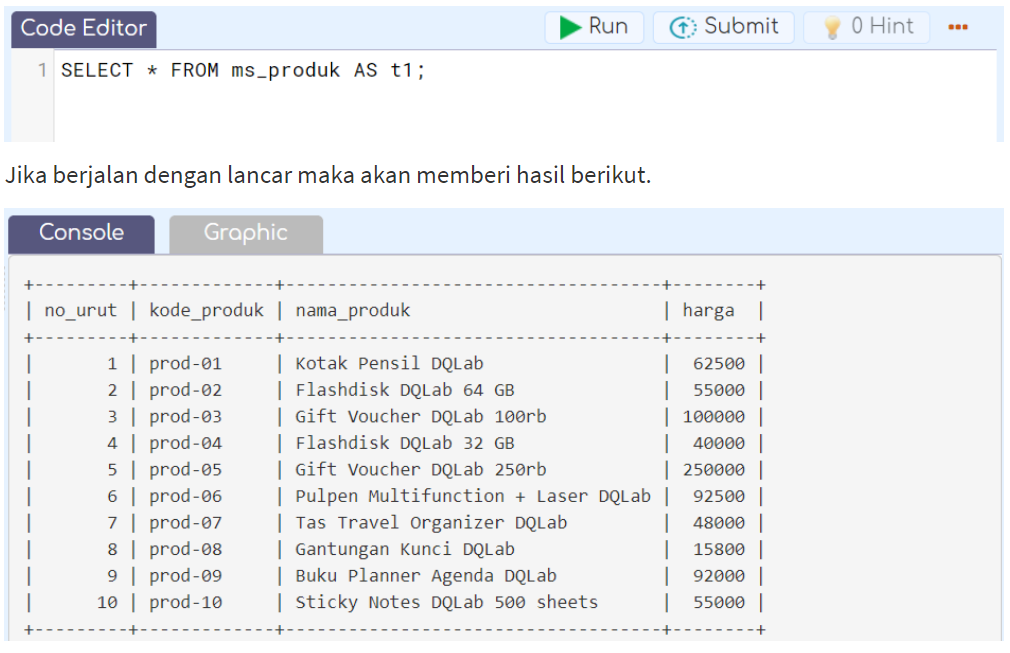


Image 28 Contoh Syntax Alias pada Table

#### Tugas :

Ganti nama tabel **ms\_produk** menjadi **t2** dan tampilkan seluruh isinya tanpa menggunakan keyword **AS**.

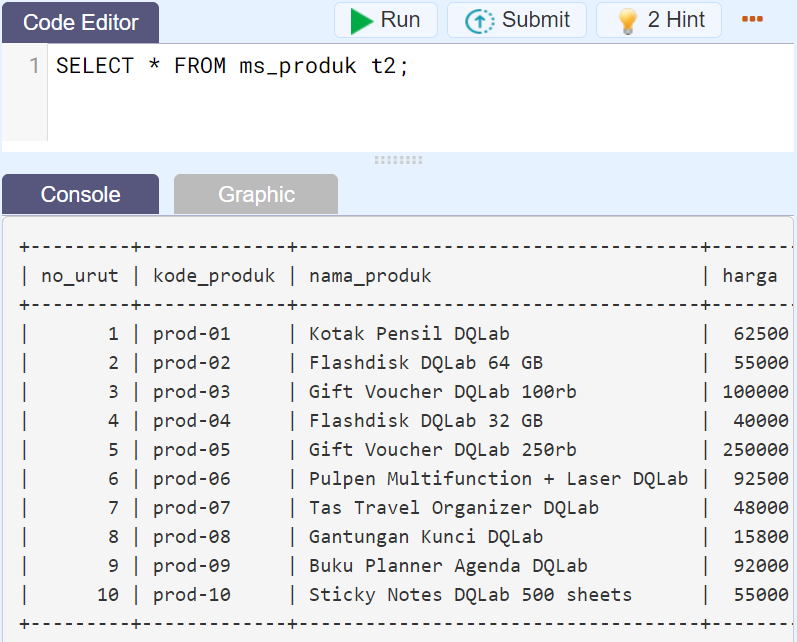


Image 29 Tanpa 'AS'

### Prefix dengan Alias Tabel

Aku menyela sebentar penjelasan Senja karena masih penasaran mengenai Prefix ini.

“Nja, kalau kita menggunakan alias tabel, maka nama prefix yang digunakan untuk kolom adalah alias tabel dan bukan nama original tabel, seperti yang ditunjukkan berikut ini. Gimana hasilnya?”

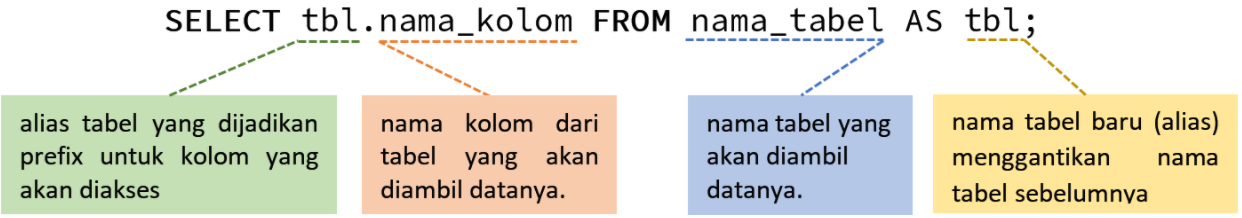


Image 30 Syntax Prefix dengan Alias Tabel

“Penggunaan nama original tabel sebagai prefix akan menimbulkan error saat query dijalankan karena dengan penggunaan alias, nama tabel secara temporary sudah di-gantikan oleh alias, Aksara,” jawab Senja lugas.

Aku mengangguk. Senja pun kembali melanjutkan penjelasannya.

Jika kolom dan tabel memiliki alias, dapat dilakukan dengan mengetikkan perintah berikut di code editor

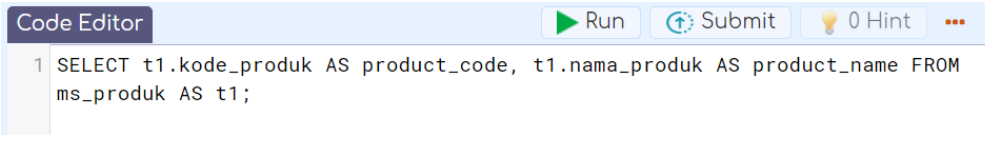


Image 31 Syntax kolom dan tabel dengan Alias

#### Tugas :

Gantilah perintah pada code editor dengan nama alias **t2** - **tanpa menggunakan keyword AS** - untuk tabel ms\_produk dan menampilkan kolom nama\_produk dan harga, lengkap dengan prefix alias.

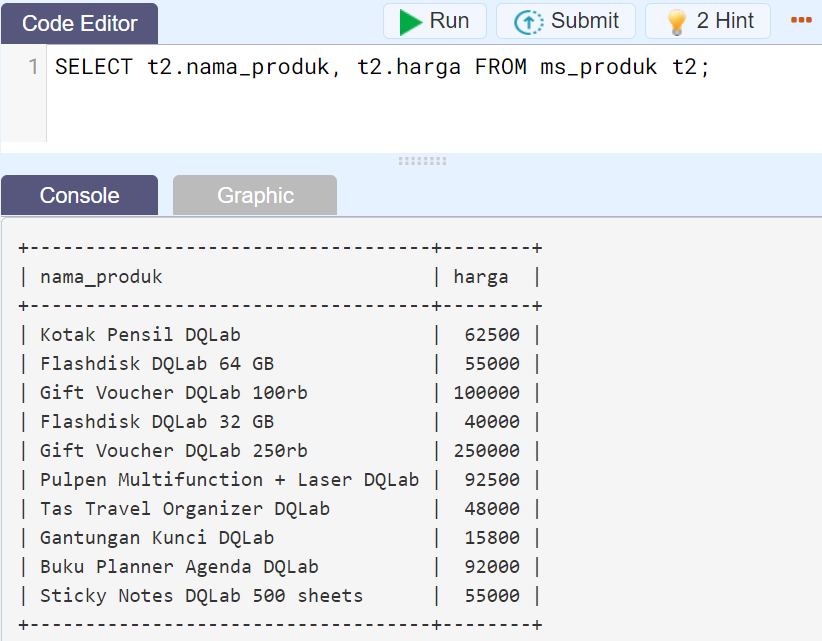


Image 32 Hasil Tugas

### Kesimpulan

Aku kembali mengambil catatanku dan menuliskan apa yang telah aku pelajari:

1. Perintah SELECT dapat ditulis dengan variasi identitas kolom dan tabel berupa prefix dan alias.
   1. Penulisan lengkap untuk nama kolom adalah prefix berupa nama tabel disertai tanda titik sebelum nama kolom itu sendiri.
   2. Alias adalah nama lain yang diberikan untuk kolom maupun tabel.
   3. Alias dapat digunakan dengan keyword AS atau tanpa keyword AS setelah nama kolom dan tabel.
   4. Prefix nama tabel bisa menggunakan alias.

Pemahaman mengenai prefix dan alias akan mendorong kemampuan identifikasi tabel maupun kolom yang terlibat untuk perintah SELECT yang lebih kompleks.

## Chapter 5

### Pendahuluan

 “Nja, sampai sini aku sudah paham bagaimana menggunakan alias dan prefix. Tapi aku masih ada pertanyaan. Terkadang kita hanya ingin menampilkan data berdasarkan kondisi tertentu, jadi kita tidak butuh semua data dari tabel. Kalau seperti itu, berarti kita hanya ingin mengambil data produk dengan nama produk tertentu. Nah, di SQL caranya gimana ya?” tanyaku. Hal ini sedari tadi membuatku penasaran.

 “Untuk case dimana kita ingin mengambil data berdasarkan kondisi tertentu saja, kita bisa menggunakan filter. SQL memiliki fungsi filter dengan menggunakan klausul WHERE. Jika kondisi WHERE terpenuhi, maka hasil query hanya akan menampilkan data yang sudah terfilter.”

Seperti biasa, aku akan lebih memahami Senja jika ada praktik yang menyertai penjelasannya. Jadi kusampaikan saja, “Nja, boleh sekalian praktik enggak?”

Senja hanya tersenyum dan hafal kebiasaanku. “Oke, biar lebih mudah dipahami, selanjutnya kita akan mempraktikkan bagaimana menggunakan klausul WHERE.”

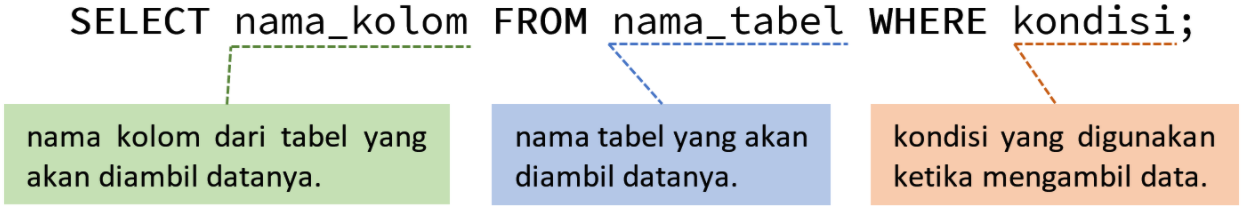
Aku pun segera membuka modul bagian klausul WHERE:

Klausul WHERE untuk:

* Filter data dengan kondisi teks tertentu.
* Filter data dengan nilai angka tertentu.
* Filter data dengan dua kondisi menggunakan operator AND dan OR.

### Menggunakan WHERE

Klausul WHERE dari SELECT digunakan untuk memfilter data berdasarkan kondisi tertentu. Untuk syntax lengkapnya adalah sebagai berikut.



Kondisi paling sederhana memiliki format sebagai berikut.

[nama\_kolom] = 'nilai\_untuk\_filter'

Biar lebih jelasnya, mari langsung contohkan dengan perintah berikut.

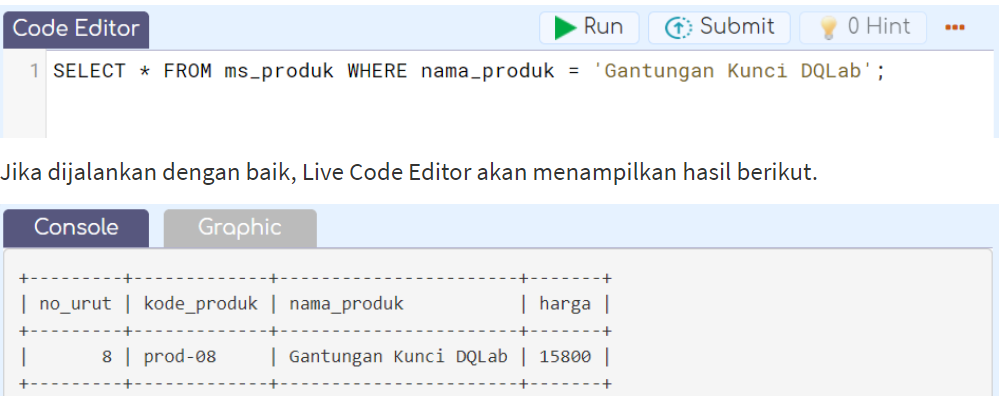


Image 33 Contoh

#### Tugas :

Cobalah ubah perintah SELECT pada code editor untuk mengeluarkan data dengan nama\_produk bernilai 'Tas Travel Organizer DQLab'.

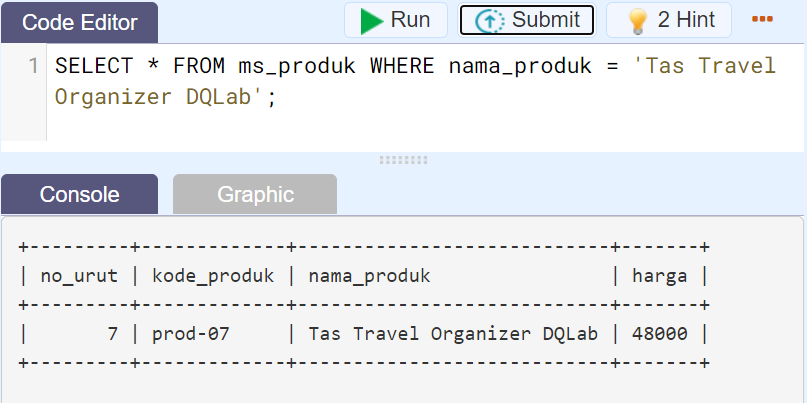


Image 34 Hasil Tugas

### Menggunakan Operand OR

Pada subbab sebelumnya, aku telah menggunakan filter teks sederhana untuk mengeluarkan data masing-masing dengan nama\_produk 'Gantungan Kunci DQLab' dan 'Tas Travel Organizer DQLab'.

Pertanyaannya, bagaimana jika ingin mengeluarkan keduanya sekaligus? Aku bisa menggunakan Operand OR.

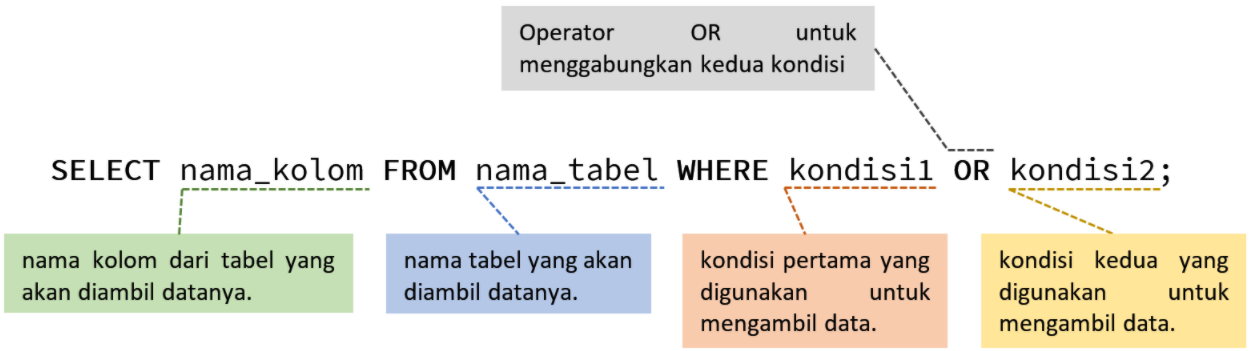


Image 35 Syntax with OR

Untuk memunculkan hasil query yang memuat data produk dengan nama\_produk 'Gantungan Kunci DQLab' dan 'Tas Travel Organizer DQLab', aku dapat menggunakan logika sederhana, yaitu: Aku perlu mengambil data dengan kondisi nama\_produk itu bernilai 'Gantungan Kunci DQLab' ATAU 'Tas Travel Organizer DQLab'. Logika ini bisa dinotasikan dengan menggunakan logika **OR.**

Sehingga, dengan menggunakan logika **OR**, aku dapat menggabungkan dua atau lebih kondisi untuk memfilter data. Jadi, untuk menyelesaikan problem yaitu memunculkan data dengan kondisi kolom nama\_produk bernilai 'Gantungan Kunci DQLab' ATAU 'Tas Travel Organizer DQLab', dapat menggunakan syntax berikut:

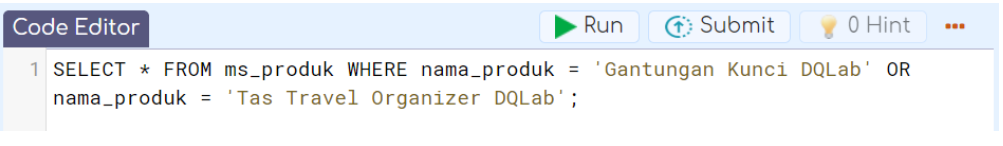


Image 36 Contoh Syntax OR

**Catatan**: Perhatikan bahwa perintah SELECT ini cukup panjang, dan riil-nya bisa dibagi menjadi beberapa baris. Namun untuk sistem DQLab, ini masih perlu dijadikan satu baris.

Jika dijalankan dengan baik maka hasilnya akan tampak sebagai berikut.

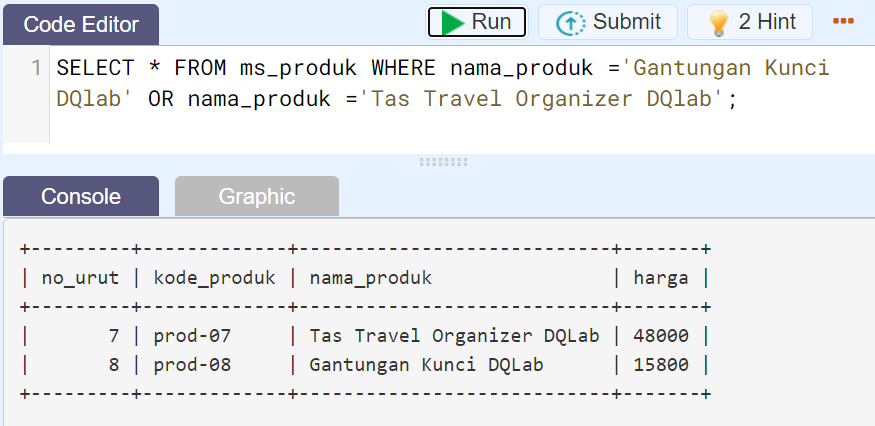


Image 37 Hasil Contoh

#### Tugas :

Tambahkan nama\_produk 'Flashdisk DQLab 64 GB' ke dalam tabel.

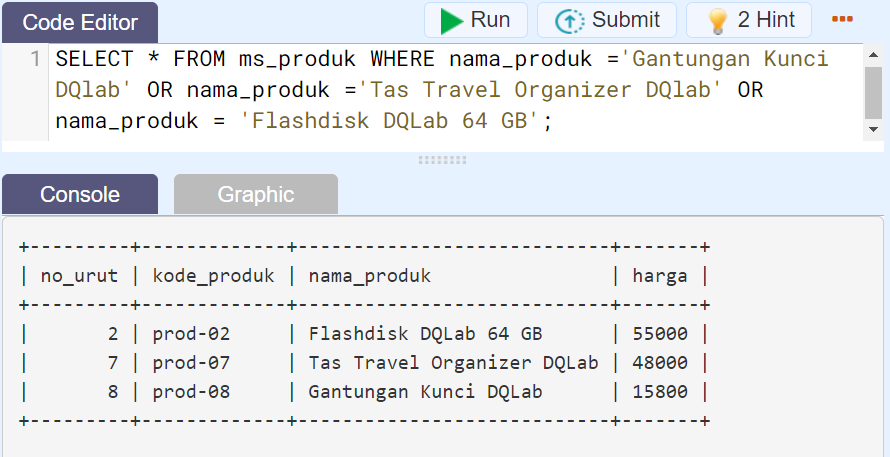


Image 38 Hasil Tugas

### Filter untuk Angka

Sebelumnya, aku telah melakukan filtering untuk teks, namun WHERE tidak terbatas untuk tipe data teks saja tapi malah umumnya untuk angka.

Berikut adalah contoh filter dimana kolom harga harus memiliki nilai di bawah 50000.

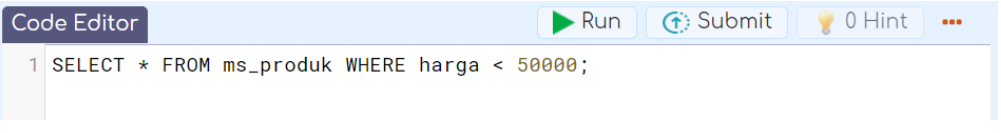


Image 39 Contoh Syntax Filter untuk angka

Jika dijalankan, maka aku akan mendapatkan tiga baris data sebagai berikut.

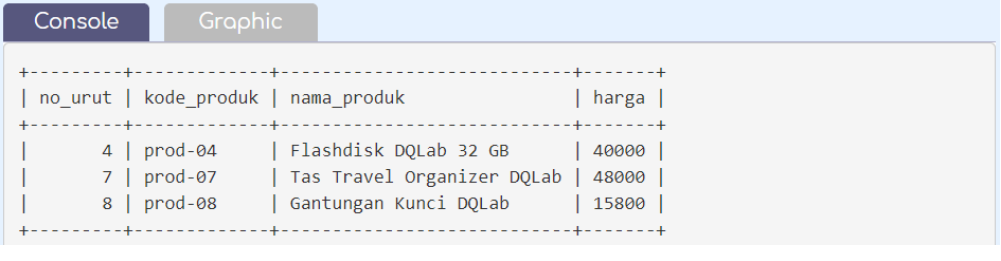


Image 40 Run Syntax Filter untuk angka

#### Tugas:

Tampilkan informasi dengan harga diatas 50000.

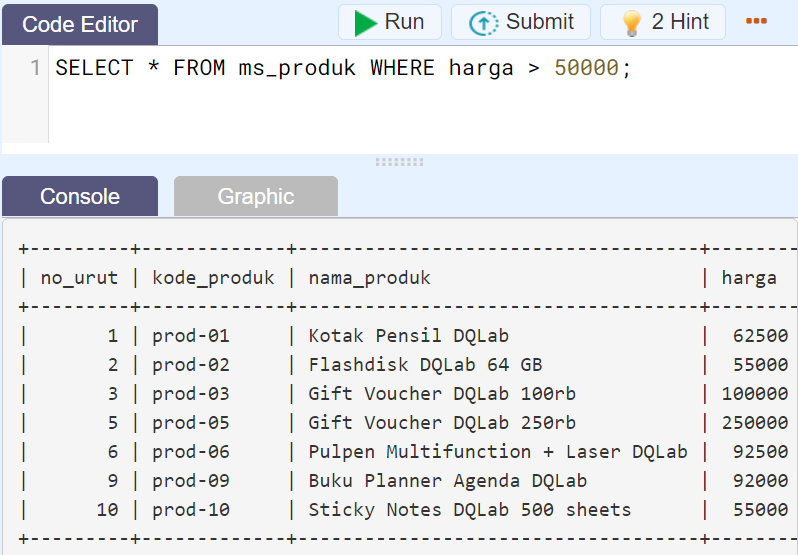


Image 41 Hasil Tugas

### Menggunakan Operand AND

Jika sebelumnya aku mempelajari Operand OR, aku juga bisa menggunakan operand AND agar dua atau lebih kondisi terpenuhi semuanya. Jika salah satu kondisi tidak terpenuhi, data tidak akan diambil. Secara umum syntaxnya diilustrasikan berikut ini

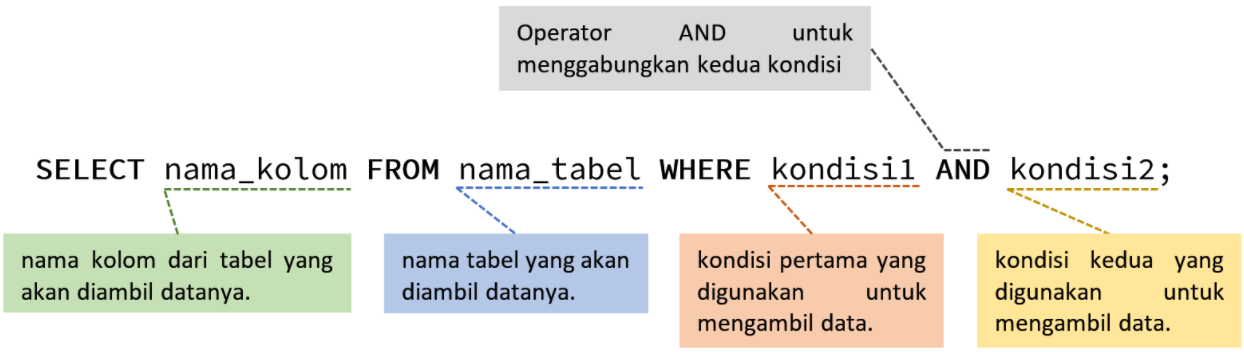


Image 42 Syntax Operand AND

Berikut adalah contoh dimana kedua kondisi digunakan dengan penghubung AND.

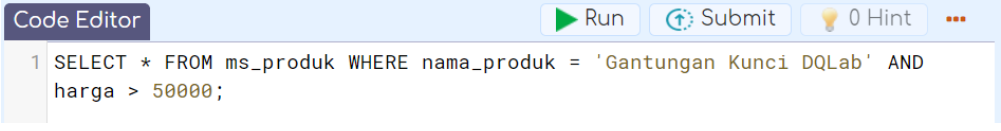


Image 43 Contoh Syntax Operand AND

Jika dijalankan dengan baik, maka akan menampilkan

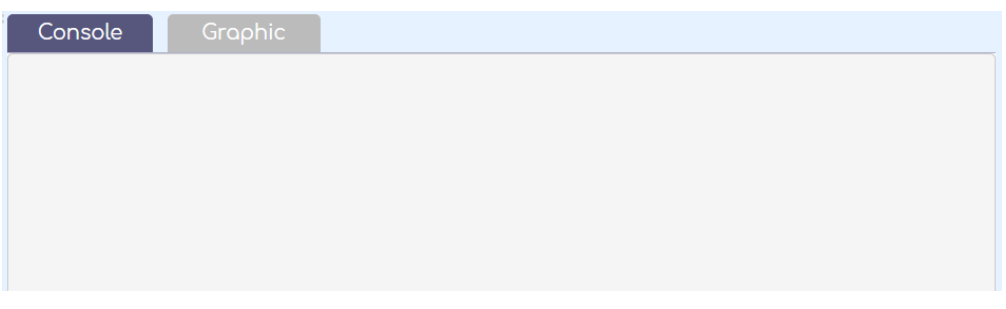


Image 44 Hasil Contoh

Tidak ada hasil yang keluar! Loh? Bingung?

Iya, karena **tidak ada** nama\_produk yang mengandung “Gantungan Kunci DQLab” dan dengan harga di atas 50,000, sehingga tidak ada hasil yang keluar karena kedua kondisi tersebut tidak terpenuhi.

#### Tugas:

Cobalah ganti kondisi contoh di atas dimana harga menjadi lebih kecil dari 50000, dengan nama\_produk yang sama.

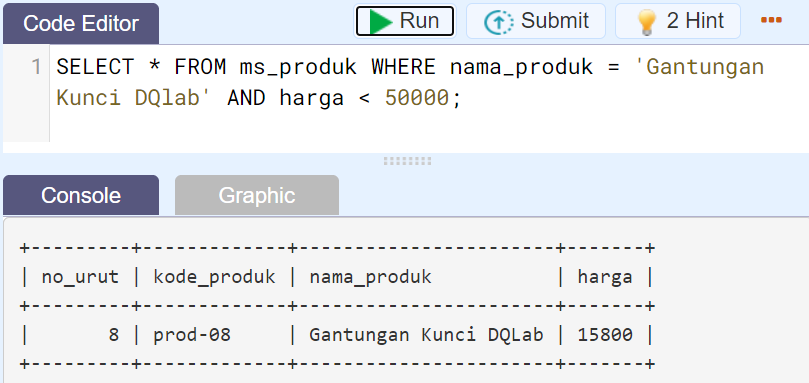


Image 45 Hasil Tugas

### Kesimpulan

Wah, semakin seru saja nih belajar SQL! Sebelum aku lupa, aku mencatat apa yang aku pelajari tadi. Siapa yang punya kebiasaan seperti aku? Dengan mencatat apa yang aku pelajari, aku merasa ilmu yang aku pelajari tadi lebih mudah untuk aku pahami.

1. Filter di SQL diimplementasikan dengan menggunakan WHERE diikuti dengan satu atau lebih kondisi logis.
2. Kondisi logis ini memiliki format nama kolom diikuti dengan nilai yang akan difilter. Untuk teks sederhana bisa menggunakan tanda sama dengan, sedangkan untuk nilai angka bisa menggunakan operator perbandingan matematika.
3. Aku juga bisa menggunakan operand OR dan AND untuk menggabungkan beberapa kondisi menjadi satu kondisi baru yang harus terpenuhi untuk pengambilan data.

## Chapter 6

### Pendahuluan

“Oke, selamat Aksara! Materi SQL untuk hari ini sudah selesai dan saya lihat kamu bisa mengikutinya dengan baik,” puji Senja.

“Ini juga berkat penjelasan dan bimbinganmu, Nja.” Dalam hati aku kembali menobatkan Senja sebagai mentor terbaik!

“Agar kemampuan di SQL lebih teruji lagi, bagaimana kalau kamu membantu saya menangani proyek baru dari cabang A ini?”

Aku menggeser bangku ke sebelah Senja untuk melihat tabel yang ditunjukkannya.

### Proyek dari Cabang A

“Jadi, apakah kamu bisa menyiapkan data transaksi penjualan dengan total revenue >= IDR 100.000?

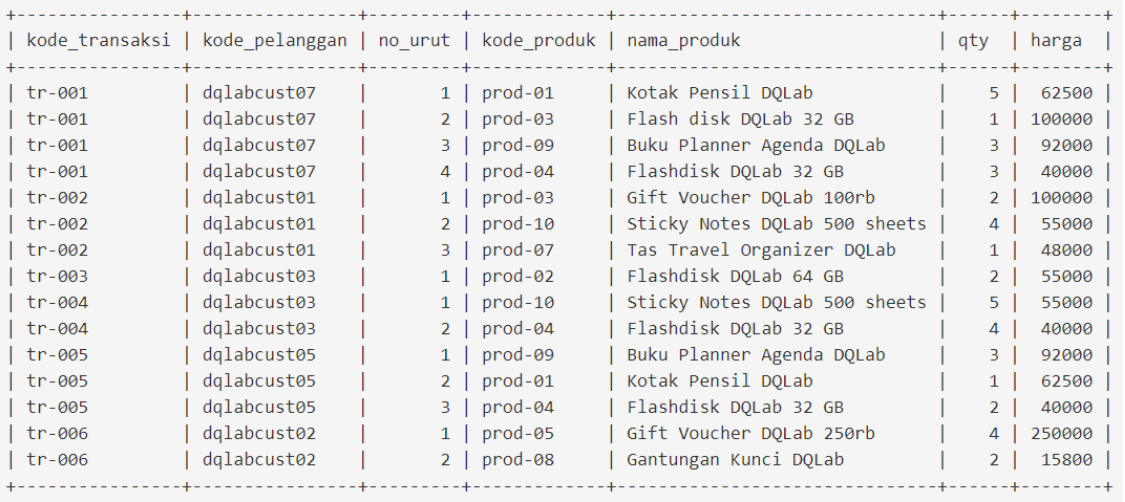
Format datanya yang akan kamu tampilkan adalah: **kode\_pelanggan, nama\_produk, qty, harga**, dan **total,** serta diurutkan mulai dari total revenue terbesar,” pinta Senja padaku.

Kalau kasusnya seperti ini, berarti aku perlu meng-query data tersebut dari tabel **tr\_penjualan** yang terdapat di database perusahaan.

Aku dapat melakukan

* perkalian antara kolom qty dan harga untuk memperoleh total revenue setiap kode pelanggan yang dinyatakan ke dalam kolom total, dan
* menggunakan “ORDER BY total DESC” pada akhir query untuk mengurutkan data.

Aku pun menerima tantangan proyek ini! Senja pun segera mengirim detailnya melalui email yang berisi contoh tabel sebagai berikut untuk segera kukerjakan.



#### Hasil Proyek

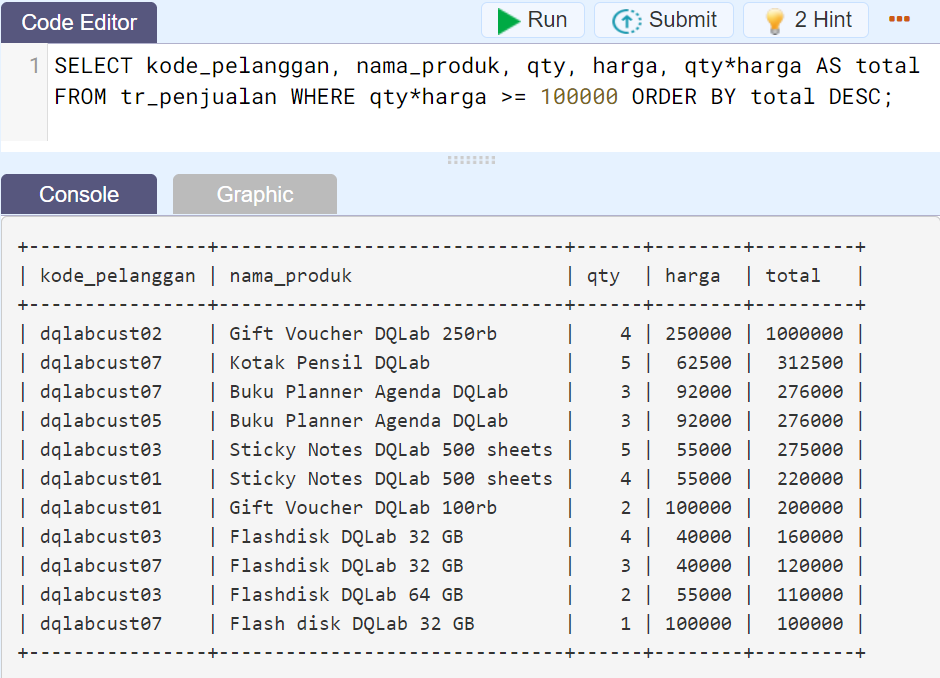


Image 46 Hasil proyek Cabang A

### Hasil Belajarku

Wah seru sekali bagian pertama dari Module SQL ini! Dari materi yang telah aku pelajari dalam 'Fundemental SQL using SELECT Statement', aku telah memahami dan mampu mempraktekkan:

1. Konsep SQL, yaitu:
   1. Konsep Sistem Database Relasional atau Relational Database Management System (RDBMS).
   2. Struktur penyimpanan RDBMS yang terdiri dari database, tabel, kolom (column) dan baris (row).
   3. Pengenalan perintah SELECT untuk mengambil data dari tabel.
2. Teknik SELECT, dimana aku dapat:
   1. Mengambil kolom tertentu.
   2. Mengambil jumlah data tertentu.
   3. Menggunakan prefix dan alias.
   4. Menggunakan filter.

Dengan kemampuan ini, aku telah siap untuk mengambil dan mengolah data secara sederhana. Keterampilan ini sendiri adalah 60% aktivitas awal yang akan dilakukan seorang analis.