

Relational Databases, ORM and Database Operation (part 1)

Gold - Chapter 4 - Topic 3

Selamat datang di **Chapter 4 Topic 3**
online course **Back End Java** dari
Binar Academy!



Kita ketemu lagi, Binarian! ☺

Gimana nih belajar kamu di Spring Framework? Apakah beneran easy to use? Atau cukup challenging?

Setelah Spring Framework, pada topic ketiga ini, kita bakal belajar tentang **Relational Database dan ORM**. Mulai dari konsep Relational Database sampai CRUD Operation.

Yuk, langsung aja kita kepoin~



Dari sesi ini, kita bakal bahas hal-hal berikut:

- Konsep database dan relational database
- Teori relational database
- Petunjuk instalasi software PostgreSQL
- Syntax SQL berupa DDL & DML
- Bagaimana membuat CRUD operation



Jalan-jalan beli bolu.

Buat mengawali pembahasan, yuk kita cari **Konsep Database**-nya dulu!

Berangkattt~



Apa itu Database?

Database atau Basis Data terdiri dari dua kata, yaitu Basis dan Data. Dimana kata Basis diartikan sebagai gudang, markas, tempat berkumpul, dan sebagainya.

Dari situ kita bisa simpulkan kalau Database atau Basis Data ini secara sederhana bisa diartikan sebagai **tempat pengumpulan data**.



Data tersebut bisa diolah atau dimanipulasi menggunakan perangkat seperti misalnya software/program/aplikasi untuk menghasilkan informasi.



Database sendiri berfungsi sebagai gudang penyimpanan data untuk diolah lebih lanjut.

Kita bisa mengorganisasi data, menghindari duplikasi data, menghindari hubungan antar data yang nggak jelas serta update data yang rumit.



“Terus, cara kerja database itu kayak gimana?“

Kamu pernah kan belanja di minimarket?

Kalau kamu perhatikan, pas kita bayar belanjaan di kasir minimarket, si petugas kasir biasanya pakai alat khusus untuk scan barcode setiap produk yang dibeli.

Pas udah di-scan, nama dan harga produk-nya bisa langsung muncul tuh di layar komputer.



Cara ini bikin tugas kasir jadi lebih mudah, lho~

Kalau pake cara scan barcode ini, petugas kasir nggak perlu repot-repot lagi deh menghafal setiap nama dan harga produk satu per satu dan memasukkan datanya ke komputer setiap ada pembeli.

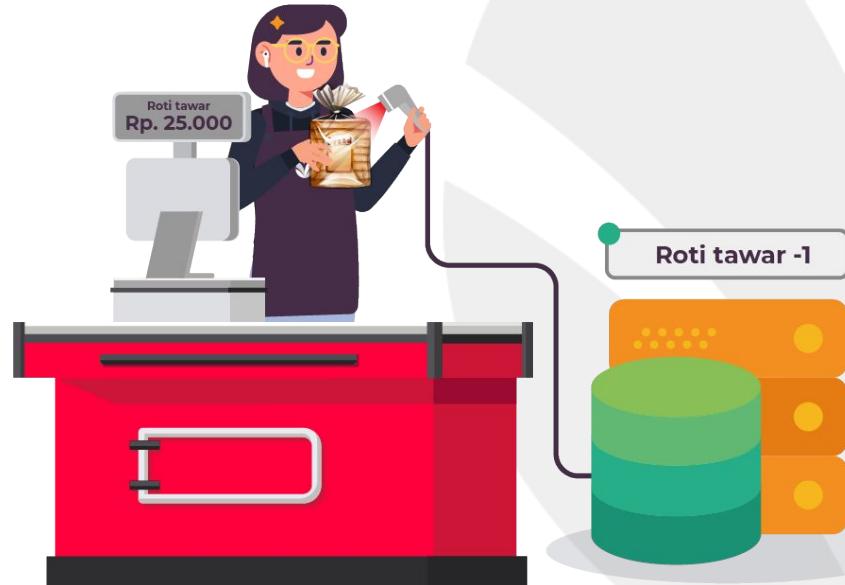
Ini semua terjadi berkat ada yang namanya database~



Sistem kasir yang pakai alat scan barcode ini ternyata memanfaatkan database stok dan harga produk yang udah dimiliki sama perusahaan minimarket tersebut.

Karena udah saling terhubung dalam sistem, setiap kali ada produk yang laku, otomatis jumlah stok di database pun bakal berkurang.

Canggih banget ya ternyata? □

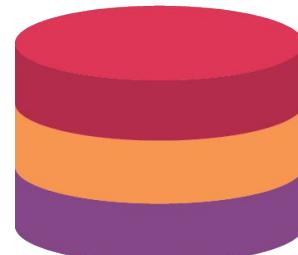


Kurang lebih, kayak gini cara kerja database~

Komputer kasir terhubung ke database yang menyimpan data semua produk. Jadi setiap kali ada produk yang laku, jumlah stok di database ikut berkurang.

Pas stoknya habis, informasinya otomatis bakalan langsung ditampilkan deh di komputer tadi!

Database Toko

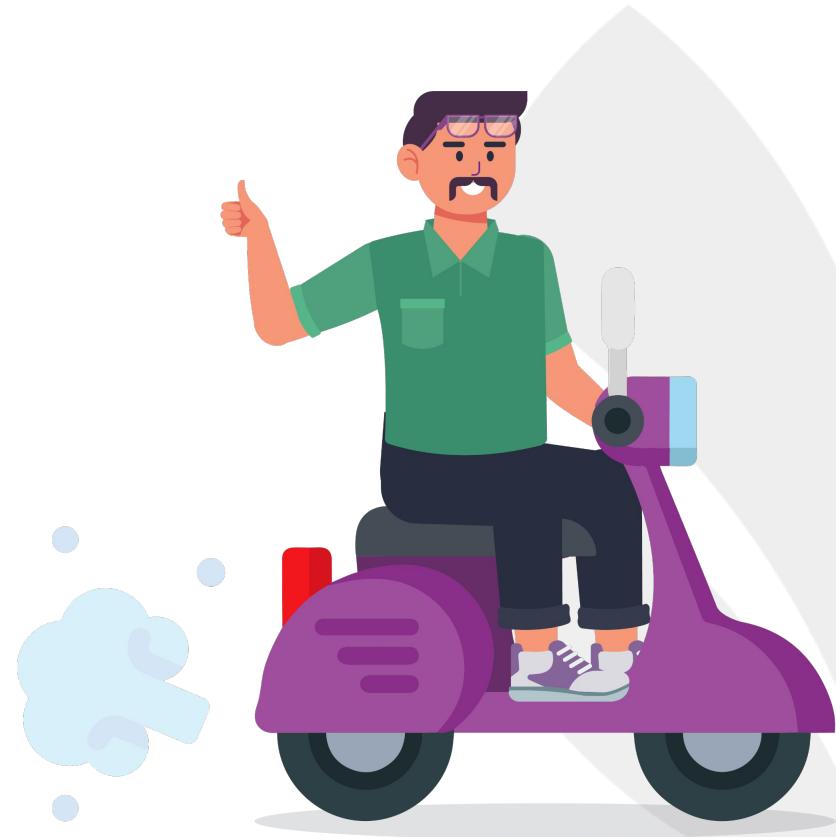


Mesin Kasir

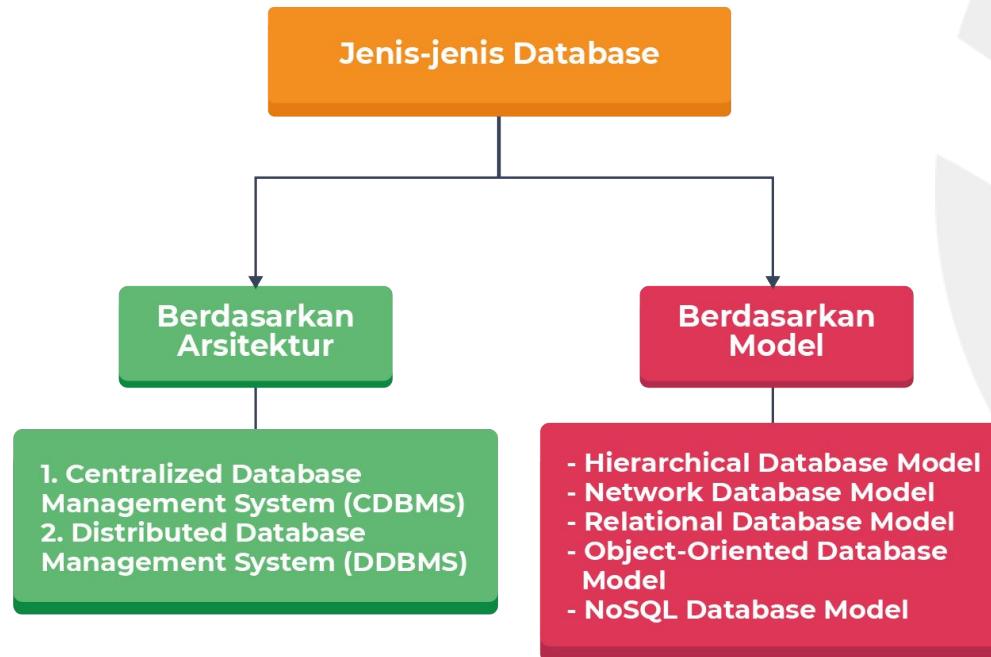
Iya, ternyata konsep dari database nggak serumit itu!

Karena kita udah memahami database itu apa, sekarang kita bakal cari tahu tentang jenis-jenis database.

Yuk gas pol! □



Berikut adalah jenis-jenis dari Database, gengs!



Sekarang kita fokus ke jenis database berdasarkan modelnya aja ya~

Model Database adalah suatu konsep yang terintegrasi dalam menggambarkan hubungan (relationships) antar data, serta batasan-batasan (constraint) data dalam suatu sistem database.



Berdasarkan bagan sebelumnya, tepatnya pada model data dilihat dari hubungan antar data dalam database, terdiri dari lima jenis, yaitu:

- **Hierarchical Database Model**
- **Network Database Model**
- **Relational Database Model**
- **Object-Oriented Database Model**
- **NoSQL Database Model**



Sipp! definisi database udah bisa dipahamin sambil merem, nih.

Kalau gitu sekarang saatnya kita meng-install software SQL yang bakal digunakan ketika pakai Relational Database Model.

Pertama, kita coba ke SQL dulu, yaitu install PostgreSQL.



Emang cara install PostgreSQL tuh gimana sih?

Simple, kok. Kamu bisa buka referensi berikut untuk melakukan instalasinya. Mudah kok caranya~

[How to install PostgreSQL](#)



Dari seluruh jenis Database berdasarkan modelnya tadi, yang paling sering dipakai oleh perusahaan adalah **Relational Database Model**.

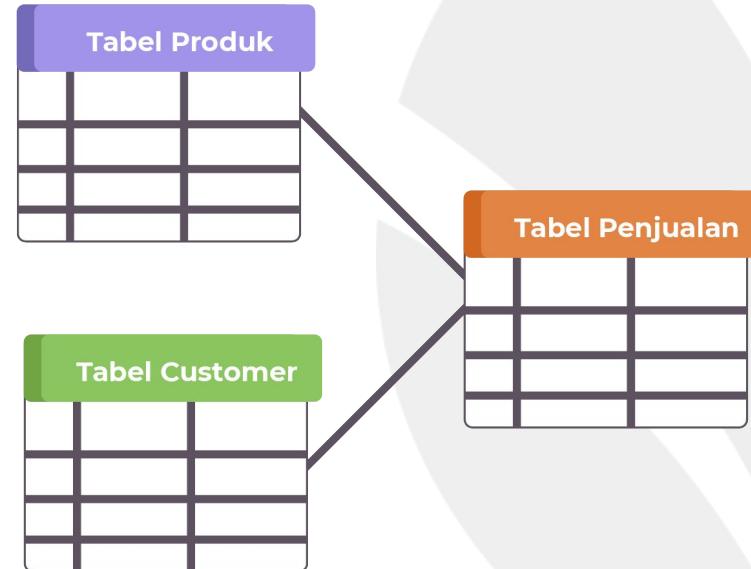
Kita coba bahas yang satu ini, ya!



Apa itu Relational Database Model?

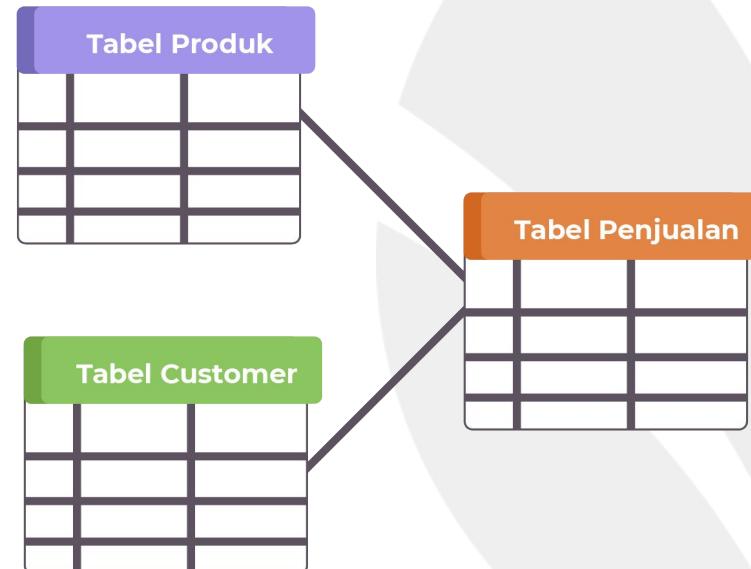
Relational Database Model adalah model yang **paling matang dan paling umum dipakai oleh banyak perusahaan** dari semua model database.

Di sini, data diorganisir berdasarkan relasi data.



Sekarang udah banyak banget perangkat lunak yang pakai sistem ini untuk mengatur dan memelihara database lewat hubungan setiap data.

Biasanya, semua sistem menggunakan **Structured Query Language (SQL)** sebagai bahasa pemrograman buat pemeliharaan relasi antar database.



Relasi data ini, ada hubungannya sama isi database, gengs!

Namanya adalah Relational Keys.

Relational Keys adalah satu atau sekelompok kolom yang nilainya bisa membedakan secara unik baris-baris tersebut.



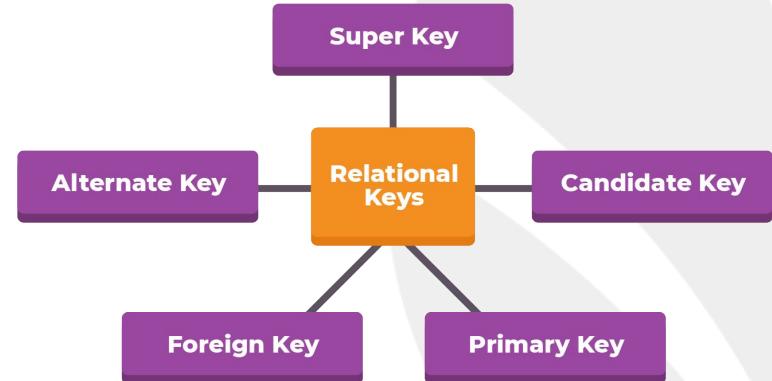
Relational Keys

Ada lima jenis Relational Keys!

Kelimanya bisa kamu cek pada gambar di samping yaaa~

“Terus arti dari setiap keys ini apa?”

Sabar, sabar. Biar lebih gampang, yuk langsung lihat contohnya~



Super Key

Super key adalah **kombinasi kolom yang secara unik menjadi identitas setiap baris dalam tabel**.

Coba kamu perhatikan contoh pada gambar di samping. Tiap baris dari Tabel Stok Produk terdiri dari kolom Kode Produk, Merek Produk, dan Stok.

Kombinasi dari semua kolom ini yang bakal menghasilkan keunikan dari tiap baris.

Tabel Stok Produk

Kode Produk	Merek Produk	Stok
B01	Susu L' zat	10
B02	Susu Cimora	12
B03	Tepung Segitiga Hijau	7

Setiap baris punya kombinasi kolom yang unik, jadi nggak ada baris yang datanya duplikat.

Candidate Key

Candidate Key adalah bagian dari Super Key. Bedanya, atribut pada Candidate Key nggak berulang.

Sederhananya begini, dari contoh pada tabel, Candidate Key-nya adalah kolom Kode Produk dan Merek Produk.

“Terus kenapa kolom Stok nggak termasuk?”

Tabel Stok Produk

Kode Produk	Merek Produk	Stok
B01	Selai L' zat	10
B02	Susu Cimora	12
B03	Tepung Segitiga Hijau	7

Kedua kolom ini punya data yang sifatnya unik dan nggak berulang.

Lebih jelasnya, ini adalah perbandingan antara Super Key dan Candidate Key ya, gengs~

Super Key	Candidate Key
Super Key adalah kumpulan kolom yang dipakai secara unik untuk mengidentifikasi semua baris dalam suatu tabel.	Candidate Key adalah bentuk yang lebih kecil dan spesifik dari Super Key.
Dalam sebuah tabel, jumlah Super Key lebih banyak dari pada jumlah Candidate Key.	Dalam suatu tabel, jumlah Candidate Key lebih sedikit dari jumlah Super Key.
Nggak semua Super Key bisa jadi Candidate Key.	Semua Candidate Key adalah Super Key.

Primary Key

Adalah suatu atribut (bisa satu atau lebih) yang dipakai untuk **memastikan bahwa setiap baris dalam tabel tersebut bersifat unik**.

Syarat kolom yang dijadikan Primary key itu kayak gini:

- Bersifat unik atau berbeda dengan nilai kolom lainnya.
- Nilai atribut nggak boleh Null (kosong, nggak diketahui, nggak bisa ditentukan)

Tabel Stok Produk

Kode Produk	Merek Produk	Stok
B01	Selai L' zat	10
B02	Susu Cimora	12
B03	Tepung Segitiga Hijau	7

Pada tabel di samping, contoh Primary Key adalah kolom Kode Produk karena bersifat unik untuk membandingkan setiap produk.

Alternatif lainnya bisa juga pakai kolom Merek Produk.

Bedanya dengan Candidate Key, di Primary Key kita **cuma menentukan satu kolom** yang benar-benar bisa membedakan setiap baris yang ada.

Tabel Stok Produk

Kode Produk	Nama Produk	Stok
B01	Selai Strawberry	10
B02	Keju Cheddar	12
B03	Tepung Gandum	7

Biar lebih kebayang sama Super Key, Candidate Key, dan Primary Key...

Bayangkan misalnya kita punya tiga anak kucing yang baru lahir~

Masing-masing anak kucing itu punya tiga identitas: **warna bulu**, **jenis kelamin**, dan **bentuk telinga**.

Ketiga cara ini bisa diibaratkan kayak **Super Key**, yaitu beberapa identitas yang jadi keunikan tiap baris data~



Tapiii, karena ada dua kucing yang berjenis kelamin betina, kita cuma bisa pakai dua cara untuk membedakan si kucing dengan mudah, yaitu lewat warna dan bentuk telinganya.

Ini mirip sama **Candidate Key**, yaitu beberapa **tanda keunikan yang nggak berulang** di tiap baris data.



Buat **Primary Key**, kita memilih **satu keunikan yang paling umum dan gampang dipakai** buat membedakan si kucing, yaitu warna bulu.

Walaupun setiap kucing punya warna bulu dan bentuk telinga yang berbeda, tapi membedakan mereka dengan melihat warnanya bakal jauh lebih gampang karena langsung kelihatan~

Sampai sini udah kebayang kan, gengs? □

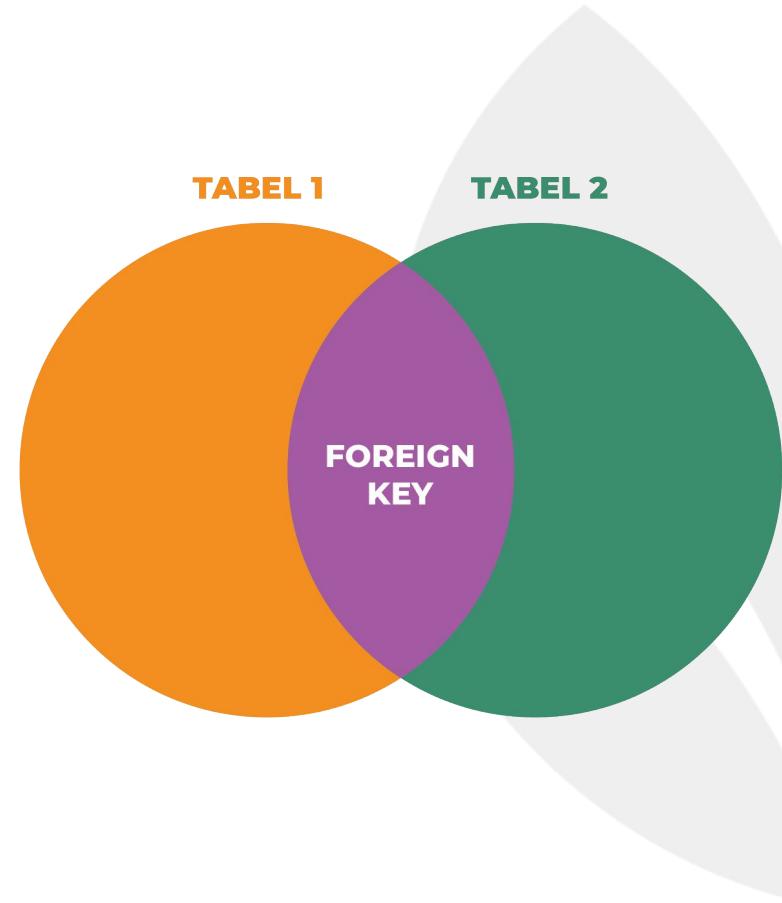


Foreign Key

Adalah suatu kolom (bisa satu atau lebih) yang dipakai sebagai penghubung antara tabel satu dengan tabel yang lainnya pada database.

Tujuannya untuk menjaga hubungan antar data dan memungkinkan navigasi antara dua tabel yang berbeda.

Foreign key ini bertindak sebagai referensi silang antara dua tabel yang punya Primary Key di tabel lainnya.



Misalnya, Tabel Stok Produk dan Tabel Penjualan adalah dua tabel yang saling berhubungan.

Buat bikin relasinya, kita bisa pakai kolom Kode Produk buat jadi Foreign Key karena kolom ini unik dan bisa memberikan informasi yang mewakili record di Tabel Stok Produk.

Cuma dengan memasukkan Kode Produk ke Tabel Penjualan, kita bisa tahu nih kalau yang dibeli adalah Selai L' zat dan Susu Cimora~

Tabel Stok Produk

Kode Produk	Merek Produk	Stok
B01	Selai L' zat	10
B02	Susu Cimora	12

Di tabel Stok Produk, Kode Produk adalah Primary Key (tidak boleh ada duplikasi)

Tabel Penjualan

Tanggal Transaksi	Kode Produk	Jumlah
14 Juni 2021	B01	2 pcs
14 Juni 2021	B02	3 pcs
15 Juni 2021	B01	1 pcs

Di tabel Penjualan, Kode Produk adalah Foreign Key (boleh ada duplikasi)

Ini dia perbedaan antara Primary Key dan Foreign Key~

Primary Key	Foreign Key
Nilai kolom bersifat unik	Boleh punya nilai kolom yang sama lebih dari satu dalam satu tabel
Nggak boleh Null	Bisa punya beberapa nilai null
Cuma ada satu Primary Key dalam satu tabel	Boleh punya lebih dari satu Foreign Key dalam satu tabel

Ini dia perbedaan antara Primary Key dan Foreign Key~

Primary Key	Foreign Key
Nilai di Primary Key nggak bisa dihapus dari tabel induk	Boleh punya lebih dari satu Foreign Key dalam satu tabel
Nilai di Primary Key nggak bisa dihapus dari tabel induk	Nilai pada Foreign Key bisa dihapus dari tabel
Nggak ada batasan dalam memasukkan nilai pada Primary Key asalkan nilainya berbeda setiap row nya	Nggak ada batasan dalam memasukkan nilai pada Foreign Key

Alternate Key

Dari contoh tadi, kita punya Candidate Key yang terdiri dari kolom Kode Produk dan Merek Produk. Terus, kita pilih satu Primary Key, yaitu Kode Produk.

Karena kolom **Merek Produk** ini nggak terpilih jadi Primary Key, berarti ia disebut sebagai **Alternate Key**!

Tabel Stok Produk

Kode Produk	Merek Produk	Stok
B01	Selai L' zat	10
B02	Susu Cimora	12
B03	Tepung Segitiga Hijau	7

Primary Key Alternate Key

Gabungan keduanya adalah Candidate Key

Kembali ke perkucinan duniawi tadi, dari **Candidate Key** yang terdiri dari warna bulu dan bentuk telinga, kita pilih warna bulu buat jadi **Primary Key**.

Bentuk telinga yang nggak kita pilih tadi, bisa disebut sebagai **Alternate Key**.



Jadi, kalau di rumah mati lampu dan kita nggak bisa bedain kucing dari warnanya, kita punya alternatif buat bedain dari bayangan bentuk telinga mereka di tembok~

Yaa, walaupun lebih gampang ambil lampu senter sih ☺



Relational Database Model punya beberapa kelebihan dan kekurangan kayak gini, nih!

Kelebihan	Kekurangan
<ul style="list-style-type: none">• Data bisa cepet banget diakses• Struktur database mudah dilakukan perubahan.• Data direpresentasikan secara logik, jadi user nggak butuh cara menyimpan data• Gampang buat membentuk query yang komplek pas melakukan pengambilan data• Gampang buat membangun dan memodifikasi program aplikasi	<ul style="list-style-type: none">• Kelompok informasi/tables yang berbeda harus dilakukan relasi (join) untuk melakukan pengambilan data• User harus familiar dengan relasi antar tabel

LANJUTTT

Udah ada gambaran belum kaitannya SQL sama Relational Database Model?

Selanjutnya kita bakal bahas materi yang serupa tapi tak sama, yaitu **DDL** dan **DML**~

Apa si kembar ini bagian dari database? Kita coba cari tau ☺♀



Sebelum bahas DDL dan DML, kamu tahu nggak kalau Database ternyata bisa diajak ngobrol lho??!

Ibaratnya nih kalau kita mau fansign sama Taehyung dan Jungkook di Korea, kita perlu tuh belajar bahasa Korea.

Buat berkomunikasi sama database, user juga perlu menggunakan bahasa pemrograman tertentu yang sesuai dengan sistem yang digunakan.

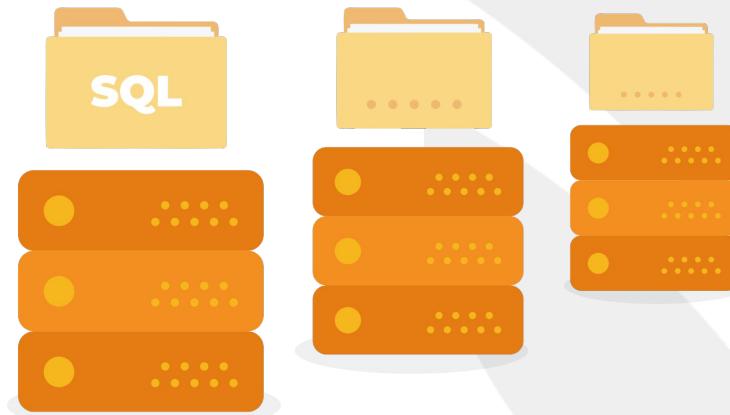
Biar pas lagi ngobrol, bisa nyambung gitu~



Salah satu bahasa pada database adalah SQL!

SQL adalah akronim dari **Structured Query Language** yang secara sederhana menggunakan SQL sebagai bahasanya.

Jadi, pas kita mau memanipulasi database, kita harus pakai bahasa SQL dan semua data kita harus mengikuti struktur yang sama.



Biar makin paham, kita pakai contoh cerita ya~

Misalnya nih, Sabrina dari bayi udah tinggal di Cappadocia.

Terus, pas udah dewasa, dia pindah ke kampung halamannya di Bekasi. Karena nggak bisa bahasa Indonesia sama sekali, dia tetap berkomunikasi sama tetangganya menggunakan bahasa Turki.



Tapi, karena tetangga Sabrina nggak ada yang ngerti sama sekali sama bahasa Turki, otomatis tetangganya malah pusing waktu berkomunikasi sama Sabrina. Alhasil Sabrina malah mengganggu semua orang.

Begitu juga sama SQL. Kalau ada perubahan dari struktur database kita, otomatis bakal bikin sulit dan mengganggu keseluruhan sistem.

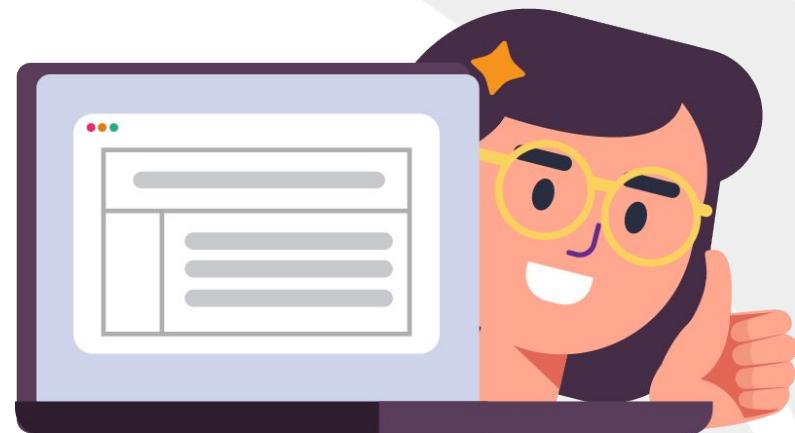
Jadi, harus hati-hati ya, bestie~



Di SQL ternyata bisa menyimpan tabel lho~

Di dalam SQL, kita bisa menyimpan data dalam bentuk tabel yang di dalamnya ada row dan column.

Buat setiap datanya kita sebut dengan **row**, sedangkan informasi-informasi yang ada di suatu data kita sebut dengan **column**.



Berikut adalah contoh software database yang berbasis SQL:

- MySQL
- PostgreSQL
- Microsoft SQL Server
- SQLite
- Oracle RDBMS
- MariaDB

Software PostgreSQL yang udah kamu install merupakan salah contohnya!



Ciri khas dari SQL tuh query-nya terstruktur dan strict banget, sob!

Iya! Bener!

Karena terlalu terstruktur dan strict-nya, query dalam SQL ini bisa kita golongkan jadi empat jenis.

Apa aja, yaaa?

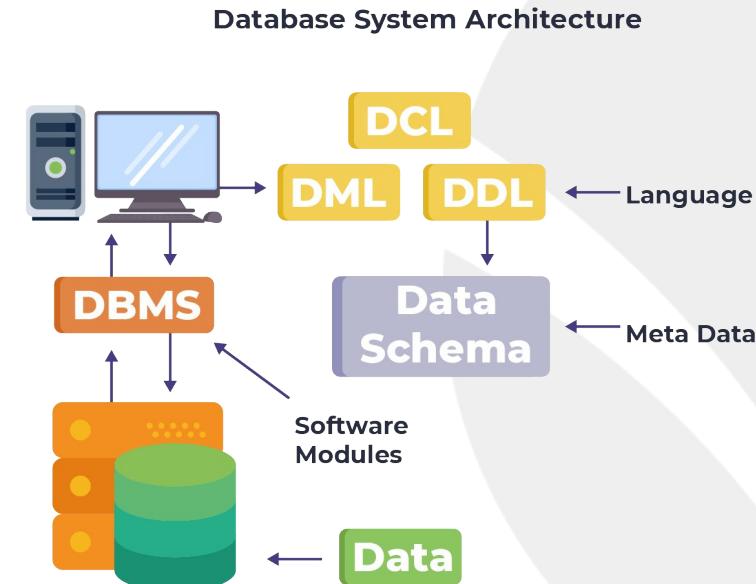


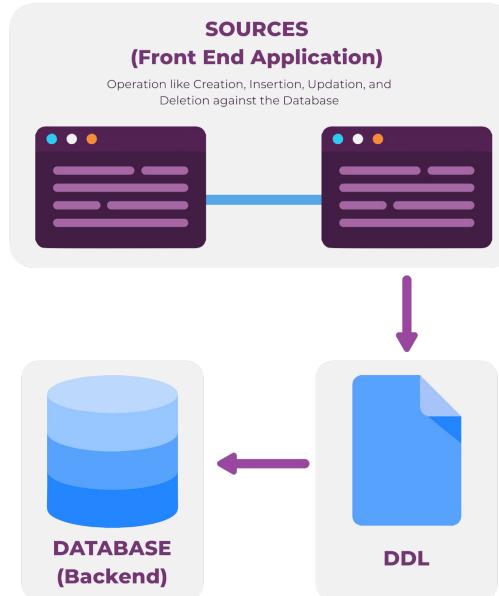
Golongan query ini di antaranya:

- Data Definition Language (DDL)
- Data Manipulation Language (DML)
- Data Control Language (DCL)
- Transaction Control Language (TCL)

Tapiii, kita cuma bakal bahas DDL dan DML aja, kok.

Kita bahas satu-satu yuk!





Query SQL yang pertama adalah DDL~

Data Definition Language adalah perintah yang kita pakai untuk mendefinisikan suatu struktur, baik itu table, database, dan lain-lain.

Berikut ini contoh perintah DDL:

- CREATE
- ALTER
- DROP

1. Create

perintah create ini dipakai untuk **membuat sesuatu di dalam SQL**, baik itu table, database, view, procedure, dan lain-lain.

Tapi, lagi-lagi di materi ini kita cuma bakal bahas beberapa jenis aja, yaitu **create database** dan **create table**.



Create database

Perintah ini cukup simpel kok. Simak caranya di bawah ini, ya. Lalu untuk membuat code-nya, kamu bisa lihat gambar di samping~

1. Create user database baru.
2. Login pakai user tersebut.
3. Create database baru.

Selesai, deh~

1

```
create user someuser password 'somepassword';
GRANT ALL PRIVILEGES ON DATABASE mydb TO someuser;
```

2



3

```
CREATE DATABASE db_name;
```

Create table

Buat bikin table, kita harus berada di dalam database yang udah kita bikin sebelumnya.

Tapi, sebelum bikin, kita perlu tahu dulu beberapa tipe data yang biasa dipakai di PostgreSQL.

Kamu bisa cek tipe data dari PostgreSQL secara lengkap pada link di bawah ini yaaa~

[PostgreSQL Data Types](#)



Berikut adalah contoh format default buat bikin table:



```
CREATE TABLE nama_table (
    nama_kolom1 TIPEDATA ATURAN,
    nama_kolom2 TIPEDATA ATURAN,
    nama_kolom3 TIPEDATA ATURAN
);
```

Setelah kita tahu formatnya, kita bisa coba bikin table untuk artikel yang di dalamnya terdapat kolom title, body, dan approved.

Oh iya, ada satu kolom yang wajib kita tambahkan, yaitu **id**.

Kolom id ini biasanya kita pakai sebagai **PRIMARY KEY**, yang berarti formatnya harus unik dan bisa jadi informasi pas kita mau cari data tersebut secara spesifik. Keunikan kolom id ini bisa kita ibaratkan seperti sidik jari suatu data.



```
CREATE TABLE users(  
    user_id INT AUTO_INCREMENT PRIMARY KEY,  
    username VARCHAR(40),  
    password VARCHAR(255),  
    email VARCHAR(255)  
);
```

Kita coba langsung bikin query-nya aja ya!

PRIMARY KEY, NOT NULL, dan DEFAULT FALSE adalah sebuah constraint yang mengatur kaidah pembuatan data dalam table tersebut.

Buat tahu lebih lanjut tentang constraint bisa dibaca di link [ini](#) ya~

2. Alter

Perintah alter ini dipakai untuk **mengubah struktur** di dalam suatu database dan table.

Sama kayak perintah DDL create, kita juga bakal bahas dua jenis dari alter, yaitu **alter database** dan **alter table**.



Alter database

Perintah ini untuk memodifikasi database. Misalnya:

- Kalau mau melakukan penamaan ulang pada suatu database.
- Kalau mau menambah kolom baru pada suatu table.

```
ALTER DATABASE nama_database
RENAME TO nama_database_baru;
```

```
ALTER TABLE contacts
ADD nama_tabel varchar(40) NOT NULL
```

Alter table

Perintah ini untuk mengganti nama table. Query-nya simpel banget, kita cuma perlu menuliskan perintah kayak berikut ini:

- Perintah default Alter
- Kalo mau menambah kolom baru pada suatu table



```
ALTER TABLE nama_tabel_lama
RENAME TO nama_tabel_baru;
```

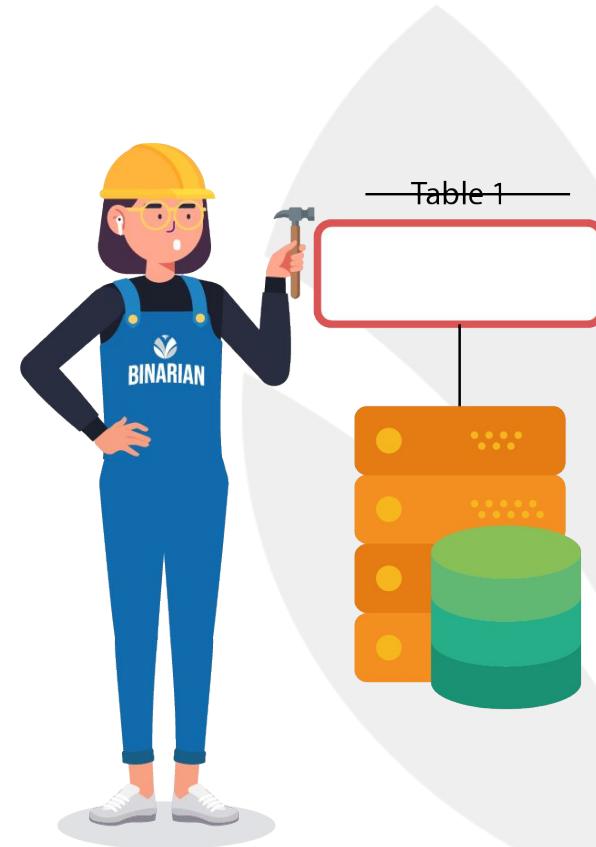


```
ALTER TABLE articles
ADD COLUMN created_at TIMESTAMP NOT
NULL;
```

3. Drop

Perintah drop ini dipakai untuk **menghapus sebuah struktur**, baik itu database maupun table.

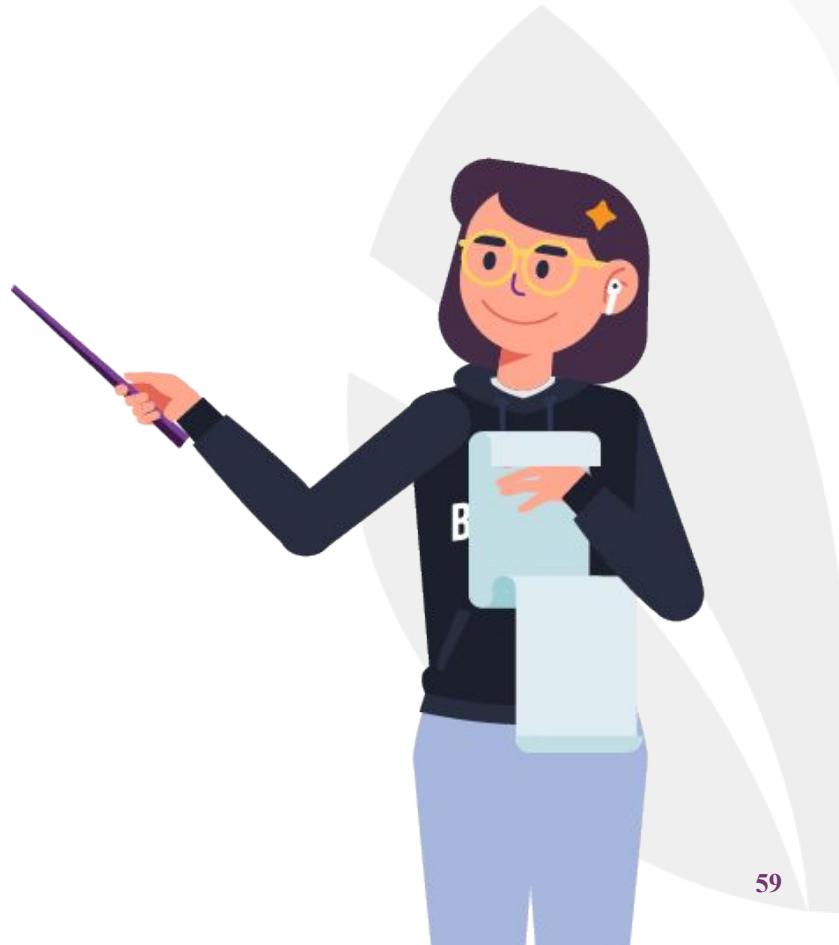
Tapi kalau untuk menghapus kenangan kamu sama si dia masih belum bisa nih ☐



Catatan penting nih, sob!

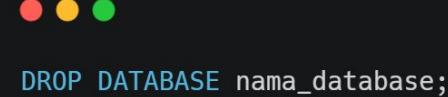
Berhubung perintah drop termasuk dalam **kategori perintah yang berbahaya karena bisa menghapus keseluruhan data**, jadi kita perlu ekstra hati-hati dalam menggunakannya, ya!

Daripada kenapa-napa kan malah repot nantinya~



Sekarang, kita coba perhatikan perintah-perintah dari Drop berikut,
yaaaa~

- Perintah drop untuk menghapus database

A dark-themed terminal window icon with three colored circular icons (red, yellow, green) at the top. Inside the window, the text "DROP DATABASE nama_database;" is displayed in white.

- Perintah drop untuk menghapus table

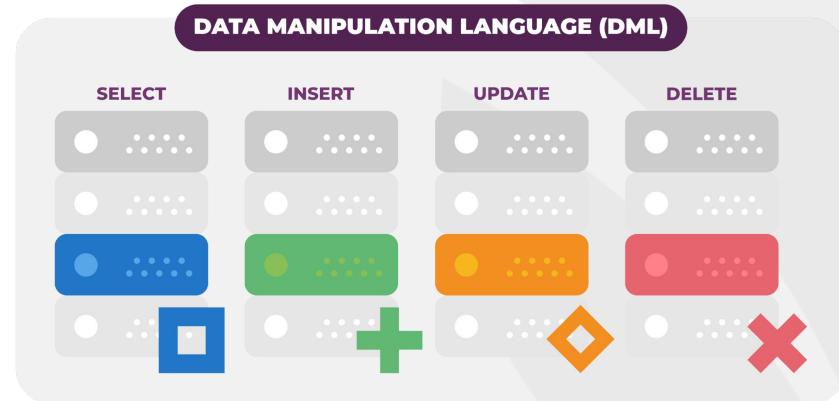
A dark-themed terminal window icon with three colored circular icons (red, yellow, green) at the top. Inside the window, the text "DROP TABLE nama_table;" is displayed in white.

Query SQL yang kedua adalah DML~

Data Manipulation Language adalah perintah yang kita pakai untuk melakukan **manipulasi data dalam suatu database**. Manipulasi itu bisa berupa mencari data, membuat data, mengubah data, dan menghapus data.

Berikut ini contoh perintah DML:

- **SELECT**
- **INSERT**
- **UPDATE**
- **DELETE**



Selanjutnya, kita bakal menggunakan perintah query untuk **Data Manipulation Language (DML)** dalam Operasi CRUD (Create - Read - Update - Delete).

Langsung aja deh kita cari tahu~



Insert

Perintah insert ini dipakai untuk **memasukkan baris data baru (create) di dalam sebuah table**.

Perhatikan perintah default insert pada gambar di samping, yaaaa~

```
● ● ●  
INSERT INTO (  
    nama_kolom1,  
    nama_kolom2  
) VALUES (nilai1, nilai2);
```

Kita bisa validasi atau mengecek datanya udah masuk atau belum pakai perintah **SELECT**



```
SELECT * FROM articles;
```

Select

Perintah select ini dipakai untuk **mencari data (read) di dalam database**. Perintah select sebenarnya bisa kompleks banget, tergantung kita mau cari data yang kayak apa.

Sebelum yang kompleks, sekarang kita coba bahas perintah yang simpel dulu aja, ya. Misalnya, mencari data dari table artikel.

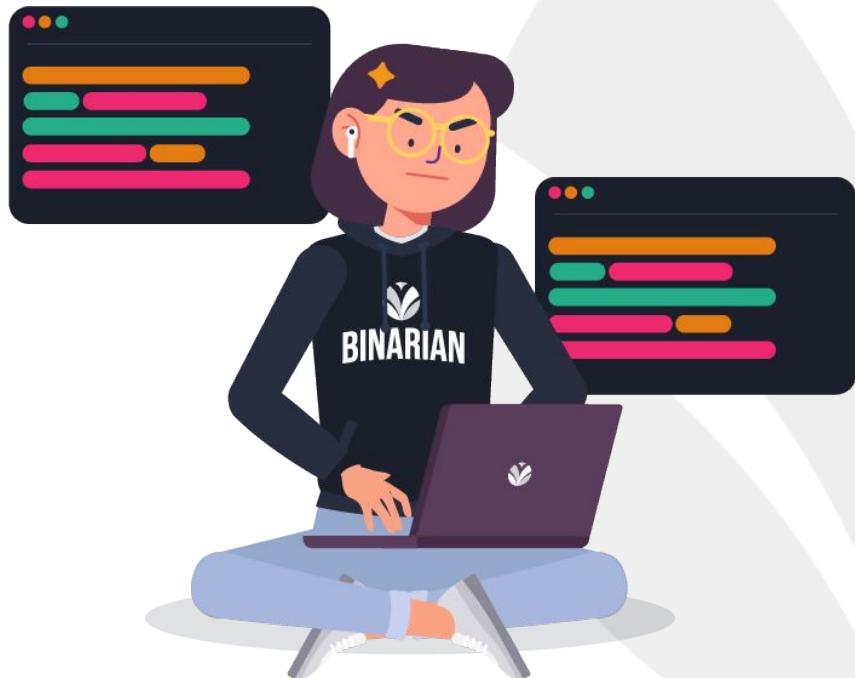
Cek gambar di samping untuk perintah default select-nya.



```
SELECT * FROM nama_tabel;
```

Response dari perintah tersebut bakal memunculkan data-data dari table articles yang berupa informasi di setiap kolomnya.

Tapi, kalau ternyata nggak ada data di dalam table, maka output-nya bakal menampilkan table kosong aja dengan judul kolomnya aja.



Kalau tabel tersebut bernama articles, maka perintahnya kayak gini,
gengs~



```
SELECT * FROM articles;
```

Btw! Kita juga bisa **cari data berdasarkan suatu kondisi dengan perintah select**, lho!

Misalnya mau cari artikel yang sudah di-approve aja, maka query-nya kayak di samping ini!

Outputnya nanti bakal menampilkan data dari table articles yang udah di-approve aja, gengs~



```
SELECT * FROM articles
WHERE articles.approved =
TRUE;
```

Selain itu, kita juga bisa milih column mana aja, lho!

Output-nya nanti bakal menampilkan data dari table articles dengan column id dan title aja~



```
SELECT id, title FROM  
articles;
```

Update

Perintah update ini dipakai untuk **memperbarui atau meng-update suatu data di dalam sebuah table**.

Berikut adalah perintah default dari UPDATE, ya.

```
● ● ●  
UPDATE nama_tabel  
SET  
    nama_kolom1 = nilai1  
    nama_kolom2 = nilai2  
WHERE kondisi;
```

Kalau yang di samping ini adalah perintah UPDATE untuk mengubah data di table articles.

Outputnya nanti bakal berupa data di table articles yang berjudul "Hello World" yang punya properti approved dan nilainya berubah jadi FALSE.

```
● ● ●  
UPDATE articles  
SET  
    approved = FALSE  
WHERE articles.title =  
    'Hello World';
```

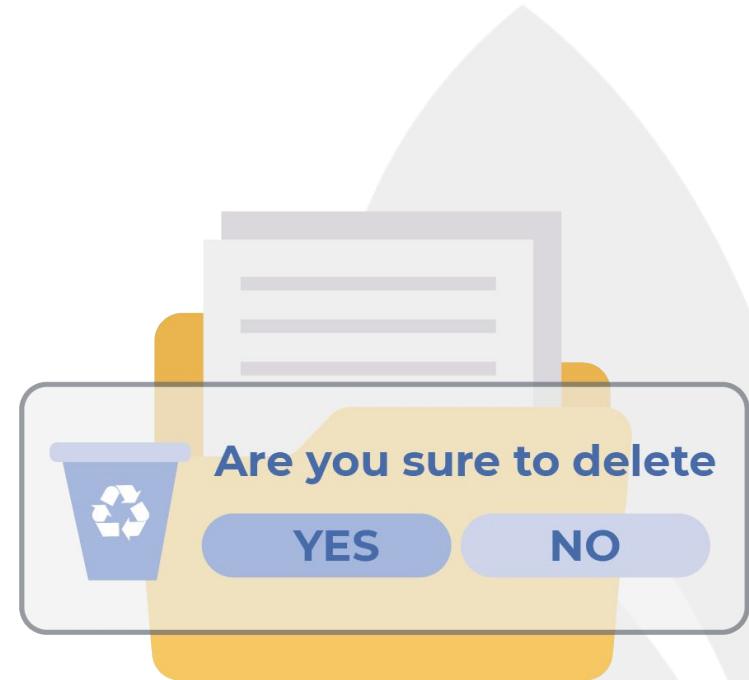
Delete

Perintah delete ini dipakai untuk **menghapus suatu data di dalam sebuah table**.

Tapi, kita harus selalu ingat untuk hati-hati ketika mau pakai perintah ini, ya.

Karena perintah delete bisa menghapus data secara permanen. Jadi, sebaiknya kita selalu pakai operator **WHERE** pas mau menggunakan perintah ini.

Kalau lupa menggunakannya, hmm.. perintah delete ini bakal menghapus semua data di dalam suatu table.



Coba perhatikan perintah-perintah delete berikut, yaaa~

- Perintah default delete.
- Perintah delete untuk menghapus data di table articles yang tidak di-approve.



```
DELETE FROM nama_tabel WHERE kondisi;
```



```
DELETE FROM articles WHERE articles.approved = FALSE;
```

Aku siap, aku siap!

Terakhir nih, sebelum kita move on ke topic selanjutnya, yuk **latihan** melakukan **CRUD operation menggunakan PostgreSQL**.

Latihan ini dilakukan di kelas dan jangan lupa untuk mendiskusikan hasil jawabannya bersama teman sekelas dan facilitator, ya.

Selamat mencoba~



Mantap, udah berhasil ya melakukan CRUD operationnya? ☐

Dari 4 query untuk melakukan data manipulation, mana sih yang paling challenging? Boleh dong cerita sama Sabrina~



Nah, selesai sudah pembahasan kita di Chapter 4 Topic 3 (part 1) ini.

Selanjutnya, yuk kita lanjut ke part 2.

Penasaran kayak gimana? Cus langsung ke topik selanjutnya~

