

Pengenalan Basis Data

Definisi Basis Data

Basis : Adalah tempat berkumpul, markas, gudang, wadah suatu data

Data : Adalah sekumpulan fakta sebuah objek

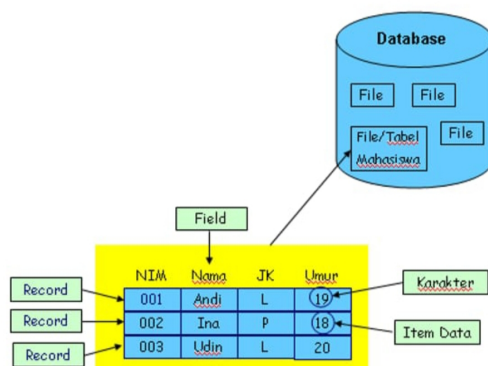
Kesimpulan : *Basis data* adalah kumpulan informasi yg disimpan di dalam komputer secara sistematis

Peranan Basis Data

Bank: Bank merupakan salah satu organisasi yang sangat tergantung kepada sistem basis data. Data nasabah, transaksi yang dilakukan, dan data keuangan lainnya disimpan di sistem basis data. Sistem ini memungkinkan layanan kepada nasabah bank dapat dilakukan dengan baik.

Data base

Contoh tabel database



Struktur Tabel Dalam Basis Data

Struktur/Hirarki DataBase adalah struktur organisasi data dalam database yang mengatur hubungan antara entitas atau tabel data. Di dalam hirarki database, data diorganisir dalam bentuk pohon dengan satu entitas induk atau tabel utama yang memiliki beberapa entitas tabel yang terkait.

Database saya anggap seperti Liga Inggris yang dimana Dia Menampung Club" Seperti MU,MC, yg di dalamnya Mempunyai Pemain Yang berbeda beda Sama Seperti Database dimana Dia Menampung File" Yg berbeda beda di dalamnya Contohnya: Ada 4 Club yg saya Ambil 1 saya Uraikan DiMana 4 Club itu ada Manchester United, Manchester city, Tottenham, Aston Villa. Yang saya Uraikan Manchester City dalam bentuk Tabel.

Struktur database

No	Nama	Alamat	Hobi
1	Fatir	Jl. Muh jufri4	Baca komik
2	Daud	Jl. Kalumpang	Main minecraft
3	Agis	Jl. Galangan kapal	Baca komik
4	Angga	Jl. Traktor4	Nonton anim

Tabel

Tabel adalah sebuah struktur dasar yang menyimpan data dalam format terstruktur. Setiap tabel memiliki kolom yang mewakili atribut dan baris yang mewakili catatan. Contoh seperti di bawah berikut

- Baris merupakan deretan horizontal yang terdiri dari kata, angka, data atau objek lainnya, contoh di atas contoh untuk baris seperti. 1, Fatir, jl. Muh jufri4, baca komik, , 2, Daud, jl. Kalumpang, main minecraft, , Dan seterusnya.
- Kolom merupakan deretan vertikal contoh di atas untuk kolom seperti. 1, 2, 3, 4, Ahmad Fatir, daud, Dan seterusnya.
- untuk isinya itu merupakan sebuah item data atau karakter yang di masukkan ke dalam tabel.

Database

Database (basis data) adalah kumpulan data yang terorganisir dengan cara tertentu untuk memudahkan pengelolaan, penyimpanan, dan pengambilan informasi. Dalam sebuah database, data disimpan dalam tabel yang terdiri dari baris dan kolom. Setiap baris dalam tabel mewakili sebuah catatan atau entitas, sedangkan kolom menyimpan. Di database juga memiliki komponen utama seperti.

1. **Tabel:** Struktur dasar yang menyimpan data dalam format terstruktur. Setiap tabel memiliki kolom yang mewakili atribut dan baris yang mewakili catatan.
2. **Baris atau Record:** Masing-masing baris dalam tabel berisi data untuk satu entitas atau catatan tertentu.

3. **Kolom atau Field:** Masing-masing kolom dalam tabel menyimpan informasi tentang atribut tertentu, seperti nama, alamat, atau nomor telepon.
4. **Item Data atau Karakter:** merupakan isian dari baris dan kolom.

instalasi mySQL

menggunakan termux

1. Buka termux
2. Ketik `termux-setup-storage`
3. Klik izinkan/allow access
4. Lakukan update dan upgrade paket.ketik `pkg update && upgrade`
5. Jika ada konfirmasi untuk melanjutkan instalasi ketik aja `Y`
6. Install aplikasi mariadb dengan mengetik `pkg install mariadb`
7. Ketika proses nya berhenti dan ada pilihan ketik saja `Y` untuk melanjutkan proses penginstalannya.
8. Ketik `mysqld_safe` untuk memberi keamanan
9. Untuk menghentikan proses `ctrl+z`
10. Masuk ke akun admin `mysql -u root`
11. Buatlah data base dengan ketik `create database xi_rpl_1;`
12. Kalo mau menampilkan data basenya ketik `show databases;`
13. Kalo mau menghapus data basenya ketik ```drop database [nama database];`
14. Kalo mau menggunakan data basenya ketik `use [nama database];`

referensi youtube

<https://youtu.be/JoJQd-l7fEE?si=OPIB01q45A2FmY1x>

penggunaan awal MySQL

Query

```
<mysql -u root -p>
```

hasil

```

sr/var/lib/mysql'

You can test the MariaDB daemon with mariadb-test-run.pl
cd '/data/data/com.termux/files/usr/share/mariadb/mariadb-test'
; perl mariadb-test-run.pl

Please report any problems at https://mariadb.org/jira

The latest information about MariaDB is available at https://mar
iadb.org/.

Consider joining MariaDB's strong and vibrant community:
https://mariadb.org/get-involved/

- $ mysqld_safe
/data/data/com.termux/files/usr/bin/mysqld_safe: Deprecated prog
ram name. It will be removed in a future release, use 'mariadb-s
afe' instead
240130 12:13:57 mysqld_safe Logging to '/data/data/com.termux/fi
les/usr/var/lib/mysql/localhost.err'.
240130 12:13:58 mysqld_safe Starting mariadbd daemon with databa
se
termux/files/usr/var/lib/mysql

Copy Paste More...      mysqld_safe

- $ mysql -u root
mysql: Deprecated program name. It will be removed in a future r
elease, use '/data/data/com.termux/files/usr/bin/mariadb' instea
d
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 3
Server version: 11.1.2-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and oth
ers.

Type 'help;' or '\h' for help. Type '\c' to clear the current in
put statement.

```

Analisis kesimpulan

- `<mySQL>` Salah satu aplikasi database server dengan bahasa pemrograman structured query language (`SQL`) yang berfungsi untuk mengelola data secara terstruktur dan sistematis.
- `←u root>` Bagian ini mengeset pengguna (user) yang akan digunakan saat terhubung ke server MYSQL. Dalam contoh ini, pengguna yang digunakan adalah "root". Pengguna "root" biasanya memiliki hak akses penuh ke server MYSQL dan dapat melakukan tindakan administratif.
- `←p>` Opsi ini digunakan untuk meminta kata sandi (password) setelah perintah dijalankan ini adalah langkah keamanan yang umum digunakan untuk memastikan hanya pengguna yang sah yang dapat mengakses server MYSQL. Setelah kita menekan Enter setelah perintah ini, kita akan diminta memasukkan kata sandi untuk pengguna "root".

data base

Database (basis data) adalah kumpulan terstruktur dari informasi yang disimpan secara elektronik dalam sistem komputer. Database dirancang untuk menyimpan, mengatur, dan mengelola data dengan cara yang efisien dan dapat diakses.

buat data base

- `CREATE DATABASE` adalah perintah untuk membuat database baru.
- `[XI_RPL_1]` adalah nama yang Anda pilih untuk database baru Anda. Tanda kurung siku `<("[]")>` digunakan di sini untuk menghindari kesalahan jika nama database mengandung karakter spesial atau spasi. Namun, perlu dicatat bahwa tidak semua DBMS

mengizinkan penggunaan tanda kurung siku dalam nama database, jadi pastikan untuk menyesuaikan sintaks dengan DBMS yang Anda gunakan.

Query

```
create database xi_rpl_1;
```

Hasil:

```
MariaDB [(none)]> create database xi_rpl_1
-> ;
Query OK, 1 row affected (0.002 sec)
```

Analisis kesimpulan :

- CREATE DATABASE adalah perintah untuk membuat database baru.
- XRPL 1 adalah nama yang Anda pilih untuk database baru Anda. Tanda kurung siku '<("{})>' digunakan di sini untuk menghindari kesalahan jika nama database mengandung karakter spesial atau spasi. Namun, perlu dicatat bahwa tidak semua DBMS mengizinkan penggunaan tanda kurung siku dalam nama database, jadi pastikan untuk menyesuaikan sintaks dengan DBMS yang Anda gunakan.

Tampilkan data base

SHOW DATABASE digunakan untuk menampilkan daftar database yang ada dalam sistem manajemen basis data (DBMS). Perintah ini dapat digunakan di beberapa DBMS seperti MYSQL, PostgreSQL, dan beberapa DBMS lainnya. Namun, perintahnya dapat sedikit berbeda tergantung

Query

```
show databases;
```

Hasil:

```
MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| phpmyadmin |
| test |
| xi_rpl_1 |
+-----+
6 rows in set (0.058 sec)
```

Analisis kesimpulan :

`SHOW DATABASE` digunakan untuk menampilkan daftar database yang ada dalam sistem manajemen basis data (DBMS). Perintah ini dapat digunakan di beberapa DBMS seperti MySQL, PostgreSQL, dan beberapa DBMS lainnya. Namun, perintahnya dapat sedikit berbeda tergantung

hapus database

`<DROP DATABASE [nama_database]>` digunakan dalam sistem manajemen basis data (DBMS) untuk menghapus sebuah database beserta semua objek yang terkait dengan database tersebut, seperti tabel, indeks, tampilan, prosedur tersimpan, dan lain-lain.

Query

```
drop database xi_rpl_1
```

Hasil:

```
MariaDB [(none)]> drop database xi_rpl_1;  
Query OK, 0 rows affected (0.002 sec)
```

Analisis kesimpulan

`DROP DATABASE` nama database digunakan dalam sistem manajemen basis data (DBMS) untuk menghapus sebuah database beserta semua objek yang terkait dengan database tersebut, seperti tabel, indeks, tampilan, prosedur tersimpan, dan lain-lain.

gunakan data base

`USE [nama_database]` digunakan dalam sistem manajemen basis data (DBMS) untuk beralih atau memilih database yang akan digunakan. Ketika Anda menggunakan perintah `<USE>` diikuti dengan nama database, DBMS akan mengarahkan semua perintah dan operasi selanjutnya pada database yang ditentukan.

Query

```
use xi_rpl_1;
```

Hasil:

```
MariaDB [(none)]> use xi_rpl_1;  
Database changed  
MariaDB [xi_rpl_1]>
```

Analisis kesimpulan :

Use [nama database] digunakan dalam sistem manajemen basis data (DBMS) untuk beralih atau memilih database yang akan digunakan. Ketika Anda menggunakan perintah `use` diikuti dengan nama database, DBMS akan mengarahkan semua perintah dan operasi selanjutnya pada database yang ditentukan.

tipe data

angka

- **Int (integer)** tipe data ini digunakan untuk merepresentasikan bilangan bulat tanpa bagian desimal. Contoh: 0, 42, -10
- **DECIMAL**: Digunakan untuk menyimpan nilai desimal presisi tinggi, cocok untuk perhitungan finansial atau keuangan.
- **FLOAT dan DOUBLE**: Digunakan untuk menyimpan nilai desimal dengan presisi floating-point. DOUBLE memiliki presisi lebih tinggi dibandingkan FLOAT.
- **TINYINT , SMALLINT , MEDIUMINT , dan BIGINT**: Tipe data ini menyimpan bilangan bulat dengan ukuran yang berbeda-beda.

teks

- **CHAR(N)**: Menyimpan string karakter tetap dengan panjang N. Contoh: CHAR(10) akan menyimpan string dengan panjang tepat 10 karakter.
- **VARCHAR(N)**: Menyimpan string karakter dengan panjang variabel maksimal N. Misalnya, VARCHAR(255) dapat menyimpan string hingga 255 karakter, tetapi sebenarnya hanya menyimpan panjang yang diperlukan plus beberapa overhead.
- **TEXT**: Digunakan untuk menyimpan teks dengan panjang variabel, tanpa batasan panjang tertentu. Cocok untuk data teks yang panjangnya tidak terduga.
- **ENUM**: Memungkinkan Anda mendefinisikan set nilai yang mungkin dan membatasi kolom hanya dapat mengambil salah satu dari nilai tersebut.
- **SET**: Mirip dengan ENUM, namun dapat menyimpan satu atau lebih nilai dari himpunan yang telah ditentukan.

tanggal

- **Date** digunakan untuk menyimpan informasi tentang tanggal, biasanya terdiri dari hari, bulan, dan tahun seperti 30 Januari 2024
- **Time** digunakan untuk menyimpan informasi tentang waktu dalam sehari, biasanya terdiri dari jam, menit, detik, dan milidetik seperti 14:30:45.500

- **DateTime** menggabungkan informasi tanggal dan waktu dalam satu objek, biasanya terdiri dari hari, bulan, tahun, jam, menit, detik, dan milidetik seperti 30 Januari 2024 14:30:45.500
- **TimeStamp**: Sama seperti DATETIME, tetapi dengan kelebihan diatur secara otomatis saat data dimasukkan atau diubah.

boolean

- **BOOL / BOOLEAN / TINYINT**: Digunakan untuk menyimpan nilai boolean, yang dapat mewakili kebenaran (true) atau kesalahan (false).

TIPE DATA PILIHAN

ENUM

Menyimpan satu nilai dari daftar nilai yang ditentukan.

SET

Menyimpan beberapa nilai dari daftar nilai yang ditentukan

Table

Buat Table

Analisis Kesimpulan

Perintah `"CREATE TABLE"` digunakan dalam SQL untuk membuat sebuah tabel baru dalam basis data. Berikut adalah penjelasan mengenai sintaks dan bagaimana menggunakan perintah `"CREATE TABLE"`.

Struktur Query:

```
CREATE TABLE [nama_table] (
  nama_kolom1 tipe_data(ukuran) [tipe_constraint]
  nama_kolom2 tipe_data(ukuran) [tipe_constraint]
  nama_kolom3 tipe_data(ukuran) [tipe_constraint] );
```

Contoh Query

```
CREATE TABLE Pelanggan (
  id_pelanggan int(4) PRIMARY KEY NOT NULL,
  nama_depan varchar(25) NOT NULL,
```



```
nama_belakang varchar(25) NOT NULL,  
no_telp char(12) UNIQUE );
```

Hasil

```
MariaDB [rental_fatir]> desc PELANGGAN  
-> ;  
+-----+-----+-----+-----+-----+-----+  
| Field      | Type      | Null | Key | Default | Extra |  
+-----+-----+-----+-----+-----+-----+  
| id_pelanggan | int(4)     | NO   | PRI | NULL    |       |  
| nama_depan   | varchar(25) | NO   |     | NULL    |       |  
| nama_belakang | varchar(25) | YES  |     | NULL    |       |  
| no_telp      | char(12)   | YES  | UNI | NULL    |       |  
+-----+-----+-----+-----+-----+-----+  
4 rows in set (0.085 sec)
```

Analisis:

1. CREATE TABLE Pelanggan:

- Ini adalah perintah SQL untuk membuat sebuah tabel baru dengan nama "Pelanggan".

2. id_pelanggan int(4) PRIMARY KEY NOT NULL:

- Kolom "id_pelanggan" dibuat sebagai tipe data integer dengan panjang maksimal 4 digit.
- PRIMARY KEY menandakan bahwa kolom ini adalah kunci utama (primary key) yang unik dan tidak boleh kosong (NOT NULL).
- Ini menunjukkan bahwa setiap baris dalam tabel akan diidentifikasi oleh nilai unik dalam kolom "id_pelanggan".

3. nama_depan varchar(25) NOT NULL:

- Kolom "nama_depan" dibuat sebagai tipe data string (varchar) dengan panjang maksimal 25 karakter.
- NOT NULL menunjukkan bahwa kolom ini tidak boleh kosong.

4. nama_belakang varchar(25) NOT NULL:

- Kolom "nama_belakang" dibuat sebagai tipe data string (varchar) dengan panjang maksimal 25 karakter.
- NOT NULL menunjukkan bahwa kolom ini tidak boleh kosong.

5. no_telp char(12) UNIQUE:

- Kolom "no_telp" dibuat sebagai tipe data karakter (char) dengan panjang tepat 12 karakter.
- UNIQUE menandakan bahwa setiap nilai dalam kolom ini harus unik, tidak boleh ada duplikat.

Kesimpulan :

tabel "Pelanggan" memiliki struktur yang terorganisir dengan baik. Setiap entri dalam tabel akan memiliki id_pelanggan sebagai identitas unik, serta informasi nama depan, nama belakang, dan nomor telepon pelanggan. Nomor telepon harus unik dalam tabel ini. Dengan struktur yang jelas seperti ini, database akan dapat menyimpan informasi pelanggan secara efisien dan memastikan integritas data yang baik.

Tampilkan Struktur Tabel

Struktur Query:

```
desc [nama_table];`
```

Contoh Query:

```
desc Pelanggan;
```

Hasil

```
MariaDB [rental_fatir]> desc PELANGGAN
-> ;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id_pelanggan | int(4)    | NO   | PRI | NULL    |       |
| nama_depan   | varchar(25) | NO   |     | NULL    |       |
| nama_belakang | varchar(25) | YES  |     | NULL    |       |
| no_telp      | char(12)   | YES  | UNI | NULL    |       |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.085 sec)
```

Analisis

`desc pelanggan;` :dapat melihat secara detail karakteristik dari setiap kolom dalam tabel tersebut, termasuk nama kolom, tipe data, panjang maksimum (jika berlaku), dan konstrain khusus seperti PRIMARY KEY, UNIQUE, atau NOT NULL yang diterapkan pada setiap kolom.

Kesimpulan

perintah tersebut memberikan gambaran tentang bagaimana tabel "Pelanggan" telah didefinisikan dalam basis data.

Menampilkan Daftar Tabel

Struktur Query:

```
show tables;
```

Contoh Query:

```
show tables;
```

Hasil

```
Database changed
MariaDB [rental_fatir]> show tables;
+-----+
| Tables_in_rental_fatir |
+-----+
| PELANGGAN               |
+-----+
1 row in set (0.001 sec)
```

Analisis:

`show tables;` : untuk menampilkan semua tabel yang ada dalam database yang sedang aktif.

Kesimpulan:

memiliki peran penting dalam memberikan visibilitas awal terhadap struktur database, yang menjadi dasar untuk pekerjaan lebih lanjut dalam pengelolaan dan penggunaan data.

QnA

❓ Mengapa hanya kolom `id_pelanggan` yang menggunakan constraint **PRIMARY KEY**?

Untuk membedakan id Pelanggan yang sama, mencegah duplikasi, dan mempermudah pencarian data.

❓ Mengapa pada kolom `no_telp` yang menggunakan tipe data `char` bukan `varchar`?

Tipe data `char` menyimpan data dalam karakter panjang lebih efisien. pencarian pada kolom tipe data `CHAR` dapat lebih cepat

❓ Mengapa hanya kolom `no_telp` yang menggunakan constraint **UNIQUE**?

Karna no_telp tidak ada yang sama semua pasti berbeda dan nilainya unik maka menggunakan constraints unique artinya data dalam tabel id_telpon berbeda tidak ada yang sama.

? Mengapa kolom no_telp tidak memakai constraint NOT NULL, sementara kolom lainnya menggunakan constraint tersebut?

Nomor telpon dianggap opsional. nomor telepon hanya menjadi wajib saat pengguna melakukan langkah-langkah tertentu, Anda mungkin tidak ingin mengharuskan pengguna mengisinya pada tahap awal.

? Perbedaan PK & UNIQUE

PRIMARY KEY untuk membedakan data yang sama dan hanya boleh 1 dan tidak boleh tidak ada.

Kalau UNIQUE sebuah kolom yang memiliki data yang berbeda atau tidak sama unique boleh 1,2,3 Dan seterusnya dan boleh tidak ada.

insert

insert 1 data

struktur

```
Insert into pelanggan  
Values(1,fatir,tir,0821772842)
```

Contoh

```
MariaDB [rental_fatir]> insert into PELANGGAN  
→ VALUES(813,"FATIR","REZKI","082163626");
```

Hasil

```
MariaDB [rental_fatir]> select * from PELANGGAN;  
+-----+-----+-----+-----+  
| id_pelanggan | nama_depan | nama_belakang | no_telp |  
+-----+-----+-----+-----+  
| 813 | FATIR | REZKI | 082163626 |  
+-----+-----+-----+-----+  
1 row in set (0.023 sec)
```

analisis

- `insert into Pelanggan`: Ini adalah perintah untuk memasukkan baris data ke dalam tabel "Pelanggan".
- `values (1,"muhammad","agis","08500000")`: Ini adalah nilai yang akan dimasukkan ke dalam tabel "Pelanggan". Urutannya sesuai dengan kolom-kolom pada tabel tersebut.

Kesimpulan

mencoba untuk memasukkan data baru ke dalam tabel "Pelanggan".

Insert >1 Data

Struktur

```
Insert into [nama_table]
Values (nilai1, nilai2, nilai3, nilai4)
       (nilai1, nilai2, nilai3, nilai4)
       (nilai1, nilai2, nilai3, nilai4)
```

Contoh

```
Insert into pelanggan
Values (2, "muh", "agis", "08900000"), (3, "muh", "daud", "08700000" ), (4,
"ahmad", "anugrah", "08100000");
```

Hasil

```
MariaDB [rental_fatir]> select * from PELANGGAN;
+-----+-----+-----+-----+
| id_pelanggan | nama_depan | nama_belakang | no_telp |
+-----+-----+-----+-----+
| 2 | muh | agis | 08900000 |
| 3 | muh | daud | 08700000 |
| 4 | ahmad | anugrah | 08100000 |
| 813 | FATIR | REZKI | 082163626 |
+-----+-----+-----+-----+
4 rows in set (0.006 sec)
```

Analisis

- `insert into Pelanggan`: Ini adalah perintah untuk memasukkan baris data ke dalam tabel "Pelanggan".
- `Values (2, "muh", "agis", "08900000"), (3, "muh", "daud", "08700000"), (4, "ahmad", "anugrah", "08100000");`: Ini adalah nilai yang akan dimasukkan ke dalam tabel "Pelanggan". Urutannya sesuai dengan kolom-kolom pada tabel tersebut.

Kesimpulan

data telah dimasukkan ke dalam tabel Pelanggan dengan masing-masing nilai kolom sesuai dengan urutan yang diberikan

Menyebut kolom

struktur

```
Insert into pelanggan  
(Nama_depan, id)value  
("Stuwja",6);
```

Contoh

```
insert into Pelanggan  
(id_pelanggan,nama_depan,nama_belakang)  
values (813,"FATIR","REZKI");
```

Hasil

```
MariaDB [rental_fatir]> select * from PELANGGAN;  
+-----+-----+-----+-----+  
| id_pelanggan | nama_depan | nama_belakang | no_telp |  
+-----+-----+-----+-----+  
| 2 | muh | agis | 0890000 |  
| 3 | muh | daud | 0870000 |  
| 4 | ahmad | anugrah | 08100000 |  
| 813 | FATIR | REZKI | 085326690 |  
+-----+-----+-----+-----+  
4 rows in set (0.001 sec)
```

Analisis

- `insert into Pelanggan` : Ini adalah perintah untuk memasukkan baris data ke dalam tabel "Pelanggan".
- `(id_pelanggan,nama_depan,nama_belakang)` : yang diberikan nilai, sedangkan kolom lainnya akan diisi dengan nilai default jika diperbolehkan atau NULL jika tidak diizinkan.
- `values (813,"FATIR","REZKI")` : Ini adalah nilai yang akan dimasukkan ke dalam tabel "Pelanggan". Urutannya sesuai dengan kolom-kolom pada tabel tersebut.

Kesimpulan

sebuah entri baru telah dimasukkan ke dalam tabel Pelanggan, Tidak ada kolom lain yang diberikan nilai dalam perintah INSERT, sehingga kolom-kolom yang tidak disebutkan akan menggunakan nilai default atau NULL.

select

seluruh data

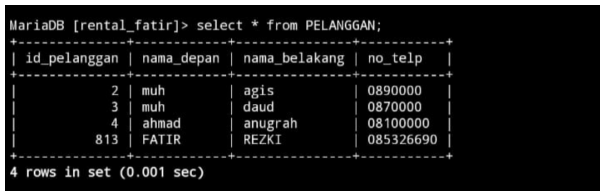
Struktur

```
select * from [nama_table];
```

Contoh

```
Select * from pelanggan;
```

Hasil



The screenshot shows a terminal window with the following content:

```
MariaDB [rental_fatir]> select * from PELANGGAN;
```

id_pelanggan	nama_depan	nama_belakang	no_telp
2	muh	agis	0890000
3	muh	daud	0870000
4	ahmad	anugrah	08100000
813	FATIR	REZKI	085326690

4 rows in set (0.001 sec)

Analisis

- `Select` merupakan query yang digunakan untuk menampilkan hasil `insert`
- `*` artinya semua kolom yang ada di table akan di tampilkan
- `from` query yang digunakan untuk memberikan penanda bahwa table mana yang akan di tampilkan
- `pelanggan` merupakan nama table yang isi nya akan di tampilkan

Kesimpulan

program akan mengambil dan menampilkan semua data yang tersimpan dalam tabel Pelanggan, termasuk setiap kolom dan setiap baris yang ada dalam tabel tersebut.

Data kolom tertentu

Struktur

```
Select [nama_kolom1], [nama_kolom2], [nama_kolom_n]  
From[nama_tabel];
```

contoh

```
Select nama_depan From pelanggan;
```

Hasil

```
MariaDB [rental_fatir]> Select nama_depan From PELANGGAN;
+-----+
| nama_depan |
+-----+
| muh        |
| muh        |
| ahmad      |
| Fadil      |
| FATIR      |
+-----+
5 rows in set (0.001 sec)
```

Analisis

- `Select` merupakan query yang digunakan untuk menampilkan hasil `insert`
- `nama_depan` nama kolom dalam tabel database yang mungkin menyimpan informasi tentang nama depan dari pelanggan.
- `from` query yang digunakan untuk memberikan penanda bahwa table mana yang akan di tampilkan
- `pelanggan` merupakan nama table yang isi nya akan di tampilkan

Kesimpulan

Hasilnya akan berupa daftar `nama_depan` dari semua pelanggan yang terdaftar dalam tabel tersebut.

kalusa WHERE

Struktur

```
Select [nama_kolom] From [nama_tabel] Where[kondisi];
```

Contoh

```
Select nama_depan From pelanggan
Where id=813;
```

Hasil

```
MariaDB [rental_fatir]> Select id_pelanggan,nama_depan From PELANGGAN
-> Where id_pelanggan=813;
+-----+-----+
| id_pelanggan | nama_depan |
+-----+-----+
| 813         | FATIR      |
+-----+-----+
1 row in set (0.006 sec)
```


Analisis

- `Select` merupakan query yang digunakan untuk menampilkan hasil `insert`
- `id_pelanggan, nama_depan` nama kolom dalam tabel database yang mungkin menyimpan informasi tentang nama depan dari pelanggan.
- `from` query yang digunakan untuk memberikan penanda bahwa table mana yang akan di tampilkan
- `pelanggan` merupakan nama table yang isi nya akan di tampilkan
- `where` untuk menyaring baris data berdasarkan kondisi tertentu.
- `id_pelanggan=1` hanya baris-baris data di mana nilai kolom

Kesimpulan

hasilnya akan berisi ID dan nama depan pelanggan yang memiliki ID tertentu

update

struktur

```
Update nama_table set nama_kolom where kondisi;
```

contoh

```
Update Pelanggan set no_telp="085326690" where id_pelanggan="813";
```

hasil

id_pelanggan	nama_depan	nama_belakang	no_telp
2	muh	agis	0890000
3	muh	daud	0870000
4	ahmad	anugrah	08100000
6	Fadil	NULL	NULL
813	FATIR	REZKI	085326690

5 rows in set (0.001 sec)

Analisis

- `UPDATE pelanggan` : Ini adalah klausa yang menentukan tabel mana yang akan diperbarui. Dalam kasus ini, tabel yang diperbarui adalah "pelanggan".
- `SET no_telp="085358639358"` : Ini adalah klausa yang menentukan kolom mana yang akan diperbarui dan nilai baru yang akan diberikan. Dalam hal ini, kolom yang diperbarui adalah "no_telp" dan nilainya diubah menjadi "085358639358".

- `WHERE id_pelanggan="1"` : Ini adalah klausa opsional yang digunakan untuk membatasi baris mana yang akan diperbarui. Dalam hal ini, perubahan hanya akan diterapkan pada baris dengan nilai "id_pelanggan" yang sama dengan "1".

Kesimpulan

nomor telepon (no_telp) dari pelanggan dengan ID "1" akan diubah menjadi "085358639358". Perintah tersebut mengupdate data pada tabel "pelanggan" dan mengaplikasikan perubahan hanya pada baris dengan nilai "id_pelanggan" yang sama dengan "1".

Delete

struktur

```
Delete from nama_table where kondisi;
```

contoh

```
delete from Pelanggan where id_pelanggan="6";
```

hasil

id_pelanggan	nama_depan	nama_belakang	no_telp
2	muh	agis	0890000
3	muh	daud	0870000
4	ahmad	anugrah	08100000
813	FATIR	REZKI	085326690

Analisis

- `DELETE FROM pelanggan` : Ini adalah klausa yang menentukan tabel mana yang akan dihapus datanya. Dalam kasus ini, data akan dihapus dari tabel "pelanggan".
- `WHERE id_pelanggan="2"` : Ini adalah klausa opsional yang digunakan untuk membatasi baris mana yang akan dihapus. Dalam hal ini, baris dengan nilai "id_pelanggan" yang sama dengan "2" akan dihapus.

kesimpulan

menghapus baris table kalian bisa menggunakan query delete dengan struktur yaitu delete from nama_table where kondisi;