

# Catatan Basis Data

## 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.

### Struktur database

NO	NAMA	KELAS	UMUR	KLM
1	Ahmad Anugrah Satya	XI RPL 1	16	L
2	Muh.Daud Reski Jayadi	XI RPL 1	17	L
3	Muh.Agis	XI RPL 1	16	L
4	Muh.Nur Reski Alfatir	XI RPL 1	16	L

### 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,Ahmad Anugrah Satya, XI RPL 1, 16, L, 2, Muh.Daud Reski Jayadi, XI RPL 1, 17, L, Dan seterusnya.
- Kolom merupakan deretan vertikal contoh di atas untuk kolom seperti. 1, 2, 3, 4, Ahmad Anugrah Satya, Muh.Daud Reski Jayadi, 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.

## 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

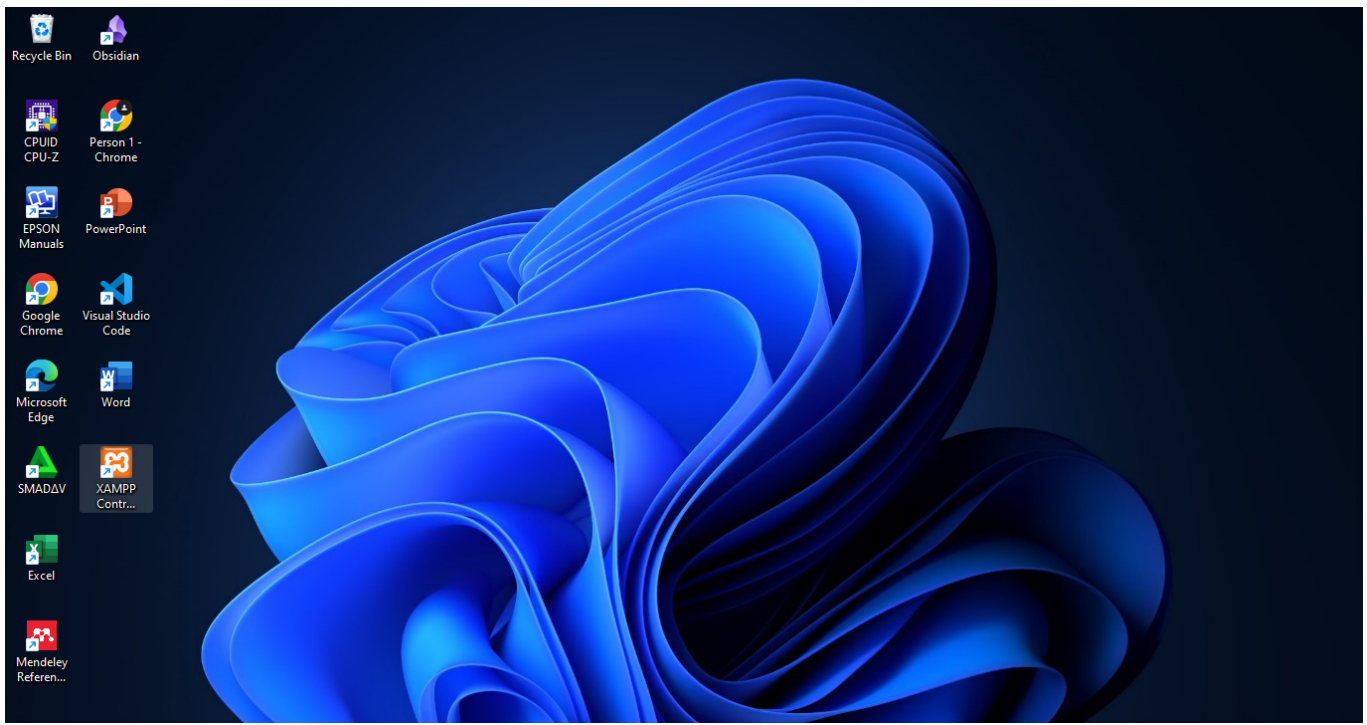
- Baris merupakan deretan horizontal yang terdiri dari kata, angka, data atau objek lainnya, contoh di atas contoh untuk baris seperti. 1,Ahmad Anugrah Satya, XI RPL 1, 16, L, 2, Muh.Daud Reski Jayadi, XI RPL 1, 17, L, Dan seterusnya.
- Kolom merupakan deretan vertikal contoh di atas untuk kolom seperti. 1, 2, 3, 4, Ahmad Anugrah Satya, Muh.Daud Reski Jayadi, Dan seterusnya.
- untuk isinya itu merupakan sebuah item data atau karakter yang di masukkan ke dalam tabel.

## Instalasi & Query Awal Databases

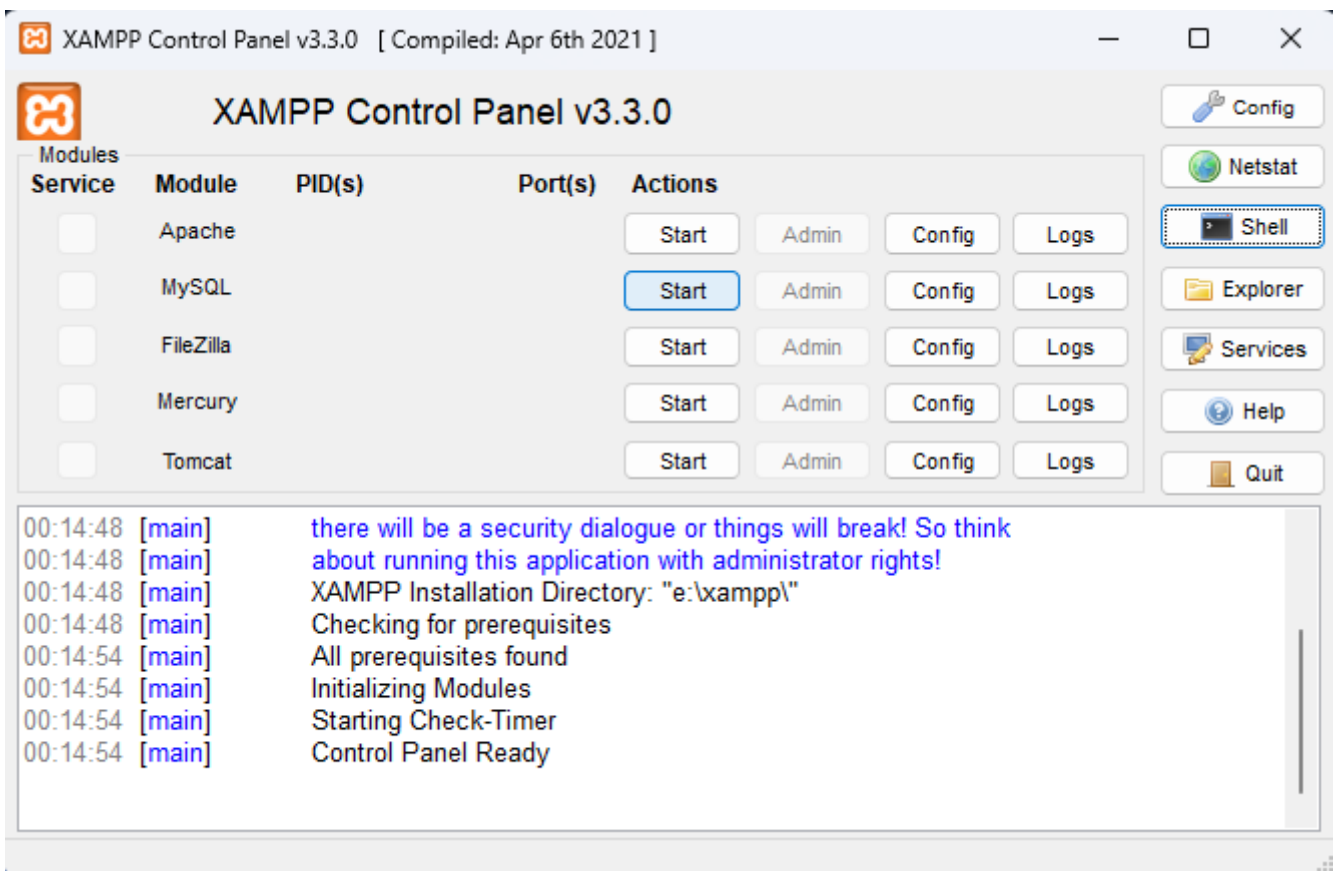
### instalase mysql

### Menggunakan XAMPP

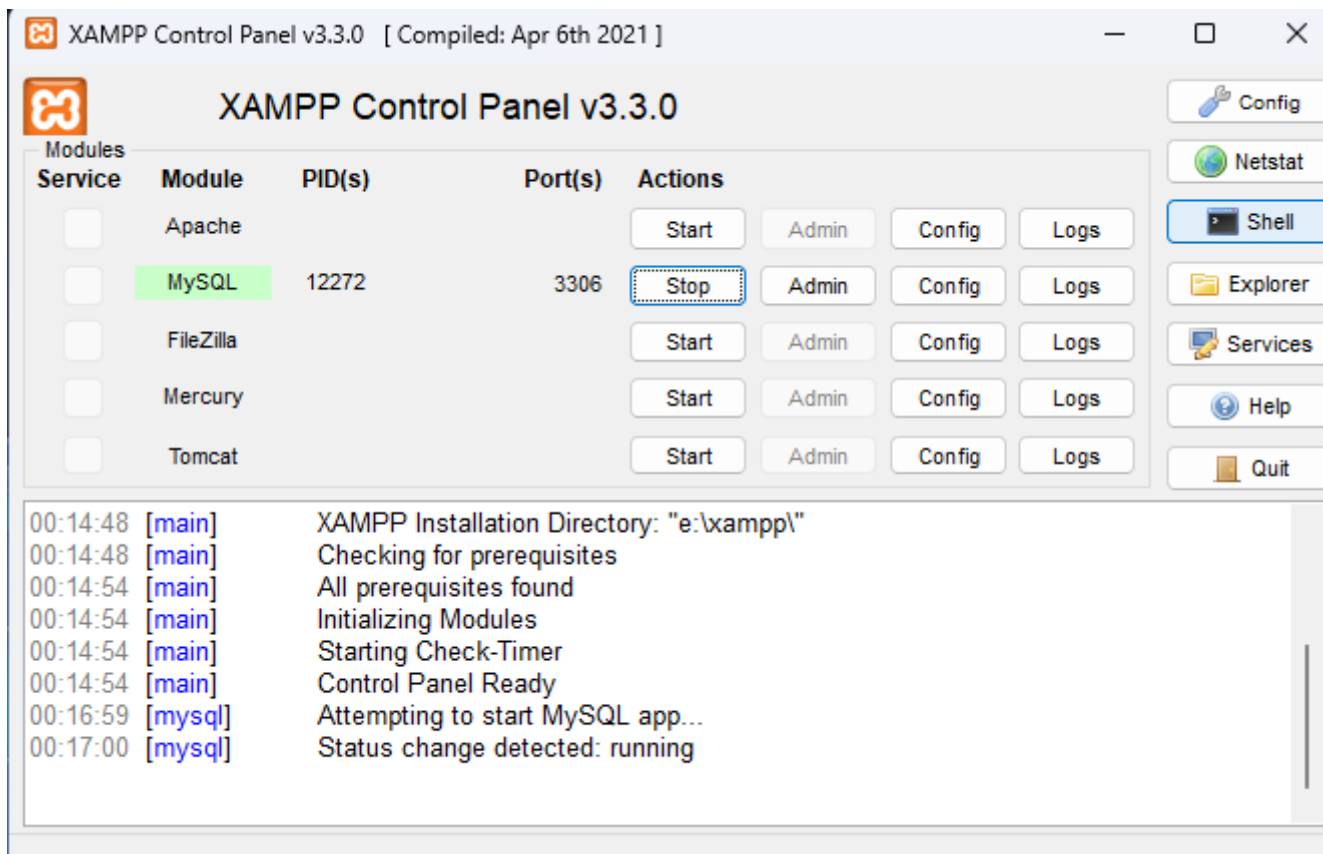
1. buka XAMPP



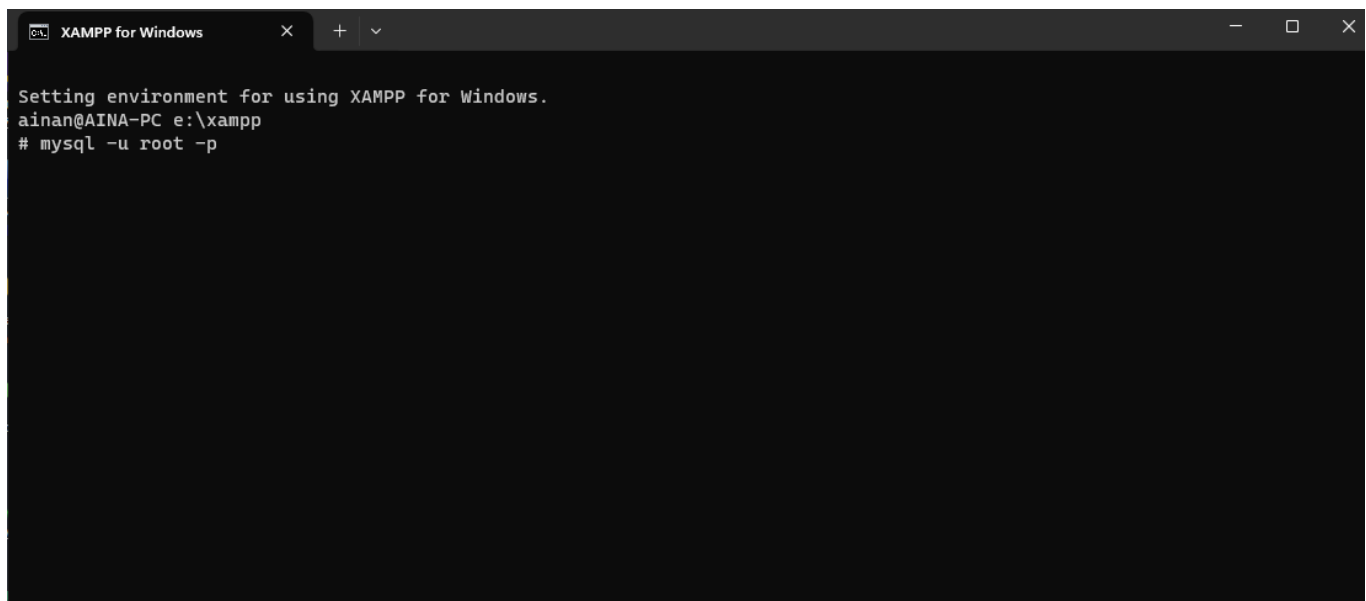
2. klik start di bagian mysql



3. klik shell



4. ketik `mysql -u root -p` lalu enter



5. nanti nya akan muncul seperti ini

```
XAMPP for Windows - mysql X + v
Setting environment for using XAMPP for Windows.
ainan@AINA-PC e:\xampp
# mysql -u root -p
Enter password:
```

6. jika setelah di enter maka dibagian tersebut tidak perlu diisi langsung saja klik enter lagi

```
Setting environment for using XAMPP for Windows.
ainan@AINA-PC e:\xampp
# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 9
Server version: 10.4.32-MariaDB mariadb.org binary distribution

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

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

MariaDB [(none)]>
```

## referensi video youtube

[https://youtu.be/tSGKZ\\_oaJOo?si=XhKflnx-D2Tq28Uy](https://youtu.be/tSGKZ_oaJOo?si=XhKflnx-D2Tq28Uy)

## Penggunaan awal mysql

query

```
mysql -u root -p
```

hasil

```
Setting environment for using XAMPP for Windows.
ainan@AINA-PC e:\xampp
# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 9
Server version: 10.4.32-MariaDB mariadb.org binary distribution

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

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

MariaDB [(none)]>
```

analisis

1. `mysql` digunakan untuk memberitahu sistem bahwa aplikasi yang ingin di gunakan yaitu `mysql`
2. `-u` digunakan untuk membuat username dengan query nya `-u`
3. `root` merupakan sebuah username yang digunakan untuk masuk ke dalam aplikasi xampp
4. `-p` digunakan untuk mengindikasikan bahwa sistem harus meminta kita untuk memasukkan kata sandi

kesimpulan

kesimpulannya adalah query `mysql -u root -p` digunakan untuk masuk ke dalam aplikasi `mysql` melalui aplikasi `xampp`, dan juga untuk `-u` membuat username.

## DataBase

### Membuat database

yaitu dengan cara mengetikkan query `create database (nama database);` contoh seperti ini `create database angga_database;` maka nanti akan muncul seperti ini jika database telah terbuat.

```
MariaDB [(none)]> create database database_1;
Query OK, 1 row affected (0.002 sec)
```

### Menampilkan database

untuk caranya kalian bisa mengetikkan query `show databases;` . penyebab mengapa ada tambahan huruf s di akhir kata database yaitu karena jika kalian mengetikkan `show databases;` maka yang akan di tampilkan ialah semua database yang ada akan tetapi jika kita mengetikkan `show database;` maka yang terjadi ialah error.

```
MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| database_1 |
| information_schema |
| mysql |
| performance_schema |
| phpmyadmin |
| test |
+-----+
6 rows in set (0.003 sec)
```

## Menghapus database

kalian bisa menggunakan cara yaitu mengetikkan query `drop database (nama database);` contoh seperti `drop database angga_database;` maka nanti akan muncul seperti ini jika database berhasil terhapus

```
MariaDB [(none)]> drop database database_1;
Query OK, 0 rows affected (1.675 sec)
```

## Gunakan database

kalian bisa melakukannya dengan cara mengetikkan query `use nama database;` contoh seperti `use angga_database;` maka nanti akan muncul seperti ini jika database telah di buat

```
MariaDB [(none)]> use database_1;
Database changed
MariaDB [database_1]>
```

 `database_1` bisa di akses padahal telah di hapus karena `database_1` telah dibuat ulang

## Tipe data

### angka

Tipe data angka dalam MySQL digunakan untuk menyimpan nilai numerik. Berikut adalah definisi untuk beberapa tipe data angka umum dalam MySQL:

1. **INT atau INTEGER:** Tipe data ini digunakan untuk menyimpan bilangan bulat tanpa koma dengan panjang standar 4 byte. Contohnya, `INT(10)`.
2. **TINYINT:** Tipe data ini digunakan untuk menyimpan bilangan bulat kecil dengan panjang 1 byte. Contohnya, `TINYINT(3)`.
3. **SMALLINT:** Digunakan untuk menyimpan bilangan bulat dengan panjang 2 byte. Contohnya, `SMALLINT(5)`.

4. **MEDIUMINT**: Tipe data ini digunakan untuk menyimpan bilangan bulat dengan panjang 3 byte. Contohnya, `MEDIUMINT(8)`.
5. **BIGINT**: Tipe data ini digunakan untuk menyimpan bilangan bulat yang sangat besar dengan panjang 8 byte. Contohnya, `BIGINT(20)`.
6. **DECIMAL** atau **NUMERIC**: Tipe data desimal yang menyimpan angka berkoma dengan presisi dan skala yang dapat diatur. Contohnya, `DECIMAL(10,2)` untuk menyimpan angka dengan 10 digit, 2 di antaranya di belakang koma.
7. **FLOAT**: Digunakan untuk menyimpan angka berkoma dengan presisi ganda. Contohnya, `FLOAT(7,4)`.
8. **DOUBLE**: Tipe data ini mirip dengan `FLOAT`, namun dengan presisi yang lebih tinggi. Contohnya, `DOUBLE(15,8)`.
9. **BIT**: Tipe data boolean yang dapat menyimpan nilai 0 atau 1, atau NULL. Contohnya, `BIT(1)`.

Definisi tipe data ini mencakup panjang (length), presisi, dan skala yang memberikan kontrol lebih lanjut terhadap cara data angka disimpan.

## Teks

Dalam MySQL, terdapat beberapa tipe data teks yang digunakan untuk menyimpan data karakter atau string. Berikut adalah beberapa tipe data teks yang umum digunakan:

1. **CHAR(n)**: Menyimpan string karakter tetap (fixed-length) dengan panjang n. Jika panjang string kurang dari n, maka akan diisi dengan spasi. Misalnya, `CHAR(10)` akan menyimpan string dengan panjang tepat 10 karakter.
2. **VARCHAR(n)**: Menyimpan string karakter dengan panjang variabel (variable-length) maksimal n. Jika panjang string yang disimpan kurang dari n, maka hanya akan menggunakan ruang yang diperlukan. `VARCHAR` lebih efisien dalam penggunaan ruang dibandingkan `CHAR`.
3. **TEXT**: Menyimpan string karakter dengan panjang variabel yang sangat besar (hingga 65,535 karakter).
4. **TINYTEXT**, **MEDIUMTEXT**, **LONGTEXT**: Variasi dari tipe data `TEXT` dengan batasan panjang yang berbeda. `TINYTEXT` dapat menyimpan hingga 255 karakter, `MEDIUMTEXT` hingga 16,777,215 karakter, dan `LONGTEXT` hingga 4,294,967,295 karakter.

Tipe data teks ini memungkinkan Anda untuk menyimpan dan mengelola data karakter dengan berbagai cara sesuai kebutuhan, baik itu string dengan panjang tetap, variabel, atau dengan batasan panjang tertentu. Pemilihan tipe data tergantung pada karakteristik data yang akan disimpan.

## Tanggal

Dalam MySQL, terdapat beberapa tipe data yang digunakan untuk menangani tanggal dan waktu. Berikut adalah beberapa tipe data tanggal yang umum digunakan:

1. **DATE**: Menyimpan tanggal dengan format "YYYY-MM-DD". Tipe data ini tidak menyimpan informasi waktu, hanya tanggal.

Contoh:

```
'2024-01-30'
```

2. **TIME**: Menyimpan informasi waktu dengan format "HH:MM:SS". Tipe data ini tidak menyimpan tanggal, hanya waktu.



Contoh:

```
'14:30:00'
```

3. **DATETIME**: Kombinasi dari DATE dan TIME. Menyimpan tanggal dan waktu dengan format "YYYY-MM-DD HH:MM:SS".

Contoh:

```
'2024-01-30 14:30:00'
```

4. **TIMESTAMP**: Mirip dengan DATETIME, tetapi dengan perbedaan utama dalam cara nilai-nilai default dan nilai saat direkam diperlakukan. Nilai TIMESTAMP secara otomatis diperbarui setiap kali baris diubah.

Contoh:

```
'2024-01-30 14:30:00'
```

5. **YEAR**: Menyimpan nilai tahun dengan format empat digit "YYYY". Digunakan terutama untuk menyimpan tahun.

Contoh:

```
'2024'
```

Tipe data tanggal ini memungkinkan pengelolaan informasi tanggal dan waktu dengan baik sesuai kebutuhan aplikasi yang menggunakan MySQL. Pemilihan tipe data yang tepat sangat bergantung pada jenis informasi waktu yang ingin Anda simpan dan bagaimana Anda berencana untuk menggunakannya.

## Boolean

Dalam MySQL, tipe data boolean umumnya diwakili oleh TINYINT(1), di mana nilai 0 menunjukkan "false" dan nilai selain 0 (biasanya 1) menunjukkan "true". Meskipun tipe data boolean tidak secara resmi didukung di MySQL sebelum versi 5.0.3, banyak aplikasi dan pengembang MySQL telah mengadopsi penggunaan TINYINT(1) untuk menyimpan nilai boolean.

Berikut adalah beberapa poin yang dapat membantu menjelaskan tipe data boolean dalam MySQL:

1. **TINYINT(1)**: Pada dasarnya, MySQL tidak memiliki tipe data boolean yang jelas. Sebagai gantinya, konvensi umum adalah menggunakan kolom TINYINT(1) untuk menyimpan nilai boolean. Ukuran 1 di sini menunjukkan panjang kolom, dan karena itu, nilai yang diizinkan hanya 0 atau 1.

Contoh:

```
CREATE TABLE contoh_boolean ( status TINYINT(1) );
```

2. **0 dan 1**: Secara tradisional, nilai 0 sering diartikan sebagai "false" dan nilai 1 diartikan sebagai "true". Namun, beberapa aplikasi atau kasus penggunaan mungkin menggunakan nilai lain sebagai representasi boolean.
3. **NULL**: Nilai NULL juga dapat diizinkan untuk kolom TINYINT(1), yang berarti bahwa kolom tersebut tidak memiliki nilai boolean yang ditetapkan.

Contoh:

```
INSERT INTO contoh_boolean (status) VALUES (NULL);
```

4. **Boolean Functions**: MySQL menyediakan beberapa fungsi untuk bekerja dengan nilai boolean atau TINYINT(1), seperti IF(), CASE, dan fungsi-fungsi logika seperti AND, OR, dan NOT.

Contoh:

```
SELECT IF(status = 1, 'Aktif', 'Tidak Aktif') AS status_text FROM contoh_boolean;
```

5. **Penanganan Tipe Data Boolean di Aplikasi:** Saat menggunakan MySQL dengan bahasa pemrograman tertentu, aplikasi dapat menangani nilai boolean ini dengan lebih mudah, memberikan abstraksi tingkat tinggi untuk representasi boolean.

Perlu dicatat bahwa sejak MySQL versi 8.0.1, MySQL memperkenalkan tipe data BOOLEAN yang sebenarnya, tetapi di bawahnya, sebenarnya masih diimplementasikan sebagai TINYINT(1) untuk kompatibilitas mundur. Namun, dengan menggunakan BOOLEAN, kita dapat menambahkan keterangan semantik dan meningkatkan kejelasan dalam skema basis data.

## Tipe data pilihan

ENUM: Menyimpan satu nilai dari daftar yang telah ditentukan sebelumnya. Setiap nilai ENUM memiliki indeks numerik yang terkait, dan nilai yang disimpan dalam tabel adalah indeks tersebut.

SET: Menyimpan satu set nilai dari daftar yang telah ditentukan sebelumnya. Nilai-nilai dalam SET diurutkan sesuai dengan urutan deklarasinya.

## Membuat Database

### Create Database

Cara untuk membuat database yaitu dengan cara

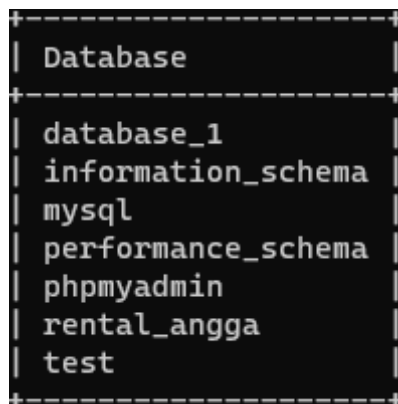
Struktur Query

```
create database nama_database;
```

Contoh Query

```
create database rental_angga
```

hasil



Database
database_1
information_schema
mysql
performance_schema
phpmyadmin
rental_angga
test

Analisis

1. `crate database rental_angga` adalah query yang yang digunakan untuk membuat table di mysql

## Kesimpulan

ketika ingin membuat sebuah database baru kita harus memasukkan sebuah query yaitu `create database nama_database;`

# Tabel

## Buat tabel

Cara membuat tabel di mysql kita bisa menggunakan sebuah query yaitu `create table nama_tabel(isian tabel);`

 sebelum ingin membuat tabel kalian sebaiknya membuat database dan masuk ke dalam database

## 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:

```
create table pelanggan (id_pelanggan int(4) primary key not null, nama_depan varchar(25) not null, nama_belakang varchar(25), no_telpon char(12) unique);
```

hasil:

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_telpon	char(12)	YES	UNI	NULL	

Analisis:

- `create table pelanggan` adalah code yang digunakan untuk membuat sebuah tabel baru dan **pelanggan** adalah nama tabel nya
- `id_pelanggan`, `nama_depan`, `nama_belakang`, `no_telpon` merupakan judul kolom
- `int(4)` merupakan tipe data angka dan 4 adalah jumlah maximal inputan
- `varchar(25)` merupakan tipe data yang Menyimpan string karakter dengan panjang variabel (variable-length) maksimal n dan 25 adalah jumlah maksimal inputan

5. `char(12)` merupakan tipe data yang menyimpan string karakter tetap (fixed-length) dengan panjang n dan 12 adalah jumlah maksimal inputan
6. `primary key` sebagai identitas yang untuk membedakan setiap baris yang ada didalam suatu tabel
7. `unique` untuk memastikan bahwa setiap baris data yang terdapat dalam suatu tabel bersifat unik (tidak sama)

#### Summary

Kesimpulannya adalah ketika kita ingin membuat tabel kalian bisa menggunakan 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] );`

## Tampilkan struktur tabel

Cara untuk menampilkan struktur tabel yaitu dengan cara menggunakan query `desc (nama table);`

contoh:

```
desc pelanggan;
```

hasil:

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_telpon	char(12)	YES	UNI	NULL	

analisis:

1. `desc pelanggan;` merupakan query yang di gunakan untuk menampilkan struktur tabel yang telah di buat dan pelanggan adalah nama dari tabelnya

#### Kesimpulan

ketika kalian ingin membuat kalian bisa menggunakan query `desc nama_table`

## Menampilkan daftar table

Cara untuk menampilkan daftar table yaitu dengan cara

contoh

```
show tables;
```

hasil:

```
+-----+
| Tables_in_rental_angga |
+-----+
| pelanggan              |
| siswa                  |
+-----+
```

analisis:

1. `show tables;` adalah query yang digunakan untuk menampilkan daftar table yang telah dibuat

#### Kesimpulan

ketika ingin melihat daftar table yang telah di buat kalian bisa menggunakan query `show tables;`

## QnA

### Mengapa hanya kolom id\_pelanggan yang menggunakan constraint PRIMARY KEY? >

karena id\_pelanggan karena ini adalah cara untuk mengidentifikasi setiap baris dalam tabel, dan juga primary key digunakan untuk memberikan hak istimewa dari sebuah kolom dan tidak dapat diduplikat oleh setiap baris.

### Mengapa pada kolom no\_telp yang menggunakan tipe data char bukan varchar? >

karena tipe data char menyimpan string karakter tetap dan no\_telp bersifat karakter tetap

### Mengapa hanya kolom no\_telp yang menggunakan constraint UNIQUE? >

karena unique berfungsi untuk memastikan bahwa setiap baris data yang terdapat dalam suatu tabel bersifat unik (tidak sama) dan no\_telp pasti tidak akan sama

### Mengapa kolom no\_telp tidak memakai constraint NOT NULL, sementara kolom lainnya menggunakan constraint tersebut? >

Karena 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.

### ❓ apa perbedaan primary key dan unique? >

kalau primary key untuk membedakan data yang sama dan hanya boleh 1 dan tidak boleh tidak ada. sedangkan 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 [nama_table] values (nilai1, nilai2, nilai3,);
```

#### Contoh

```
insert into pelanggan values(1, 'Ahmad', 'Satya', '0895XXXX');
```

#### Hasil

id_pelanggan	nama_depan	nama_belakang	no_telpon
1	Ahmad	Satya	0895XXXX

#### Analisis

1. `insert into` adalah query yang digunakan untuk menginput isi table
2. `pelanggan` adalah nama table nya
3. `values` adalah query yang digunakan untuk memasukkan nilai ke kolom
4. `(1, 'Ahmad', 'Satya', '0895XXXX')` nilai yang ingin di masukkan ke kolom
5. `;` penutup query

#### Kesimpulan

Jika ingin memasukkan sebuah nilai ke dalam tabel maka kalian bisa menggunakan query `insert into [nama_table] values(nilai1, nilai2, nilai3);`

## insert lebih dari 1 data

### Struktur

```
insert into [nama_table] values (nilai1, nilai2, nilai3,), (nilai4, nilai5, nilai6,);
```

### Contoh

```
insert into pelanggan values(1, 'Ahmad', 'Satya', '0895XXXX'), (2, 'Daud', 'Resky', '0812XXXX'), (3, 'Resky', 'Alfatir', '0834XXXX'), (4, 'Agis', null, '0856XXXX');
```

id_pelanggan	nama_depan	nama_belakang	no_telpon
1	Ahmad	Satya	0895XXXX
2	Daud	Resky	0812XXXX
3	Resky	Alfatir	0834XXXX
4	Agis	NULL	0856XXXX

### Analisis

1. `insert into` query yang di gunakan untuk menginput isi table
2. `pelanggan` adalah nama table yang ingin di isi
3. `values` query yang di gunakan untuk memasukkan nilai ke kolom
4. `(1, 'Ahmad', 'Satya', '0895XXXX'), (2, 'Daud', 'Resky', '0812XXXX'), (3, 'Resky', 'Alfatir', '0834XXXX'), (4, 'Agis', null, '0856XXXX')` merupakan nilai yang akan di masukkan

#### Kesimpulan

jika ingin mengisi atau menginput kolom di mysql kalian bisa menggunakan query dengan struktur `insert into [nama_table] values (nilai1, nilai2, nilai3,), (nilai4, nilai5, nilai6,);`

## Menyebut Kolom

### Struktur

```
insert into [nama_table](kolom1, kolom2, kolom3) values (nilai1, nilai2, nilai3);
```

## Contoh

```
insert into pelanggan (id_pelanggan, nama_depan) values (5, "fahri")
```

## Hasil

id_pelanggan	nama_depan	nama_belakang	no_telpon
1	Ahmad	Satya	0895XXXX
2	Daud	Resky	0812XXXX
3	Resky	Alfatir	0834XXXX
4	Aqis	NULL	0856XXXX
5	fahri	NULL	NULL

## Analisis

1. `insert into` query yang digunakan untuk menginput isi table
2. `pelanggan` adalah nama table yang ingin di isi
3. `(id_pelanggan, nama_depan)` nama kolom yang hanya akan di isi
4. `values` query yang di gunakan untuk memasukkan nilai ke kolom
5. `(5, "fahri")` merupakan nilai yang akan di masukkan

### Kesimpulan

jika ingin memasukkan nilai yang hanya ke kolom tertentu kalian bisa pakai sebuah query yang strukutur nya `insert into [nama_table](kolom1, kolom2, kolom3) values (nilai1, nilai2, nilai3);`

## Select

### Seluruh Data

### Struktur

```
Select * from [nama_table]
```

## Contoh

```
select * from pelanggan;
```

## Hasil



id_pelanggan	nama_depan	nama_belakang	no_telpon
1	Ahmad	Satya	0895XXXX
2	Daud	Resky	0812XXXX
3	Resky	Alfatir	0834XXXX
4	Agis	NULL	0856XXXX
5	fahri	NULL	NULL

## Analisis

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

### Kesimpulan

jika ingin menampilkan hasil dari insert kalian bisa menggunakan query yaitu dengan struktur `Select *`  
`from [nama_table]`

## Data Kolom Tertentu

### Struktur

```
select [nama_kolom1], [nama_kolom2], mysqlolom_n]
```

### Contoh

```
select nama_depan from pelanggan;
```

### Hasil

nama_depan
Ahmad
Daud
Resky
Agis

## Analisis

1. `select` merupakan query yang digunakan untuk menampilkan hasil `insert`
2. `nama_depan` adalah nama kolom yang akan di tampilkan
3. `from` query yang digunakan untuk memberikan penanda bahwa table mana yang akan di tampilkan
4. `pelanggan` merupakan nama table yang isi nya akan di tampilkan

## Klausu WHERE

### Struktur

```
select [nama_kolom] form pelanggan [nama_table] where[kondisi];
```

### Contoh

```
select id_pelanggan, nama_depan from pelanggan where id_pelanggan=2;
```

### Hasil

id_pelanggan	nama_depan
2	Daud

### Analisis

1. `select` merupakan query yang digunakan untuk menampilkan hasil `insert`
2. `id_pelanggan, nama_depan` adalah nama kolom yang akan di tampilkan
3. `from` adalah penanda yang digunakan untuk menandakan table mana yang akan di tampilkan
4. `pelanggan` adalah nama table yang akan di tampilkan
5. `where` adalah query yang digunakan untuk memberikan sebuah kondisi
6. `id_pelanggan=2` adalah sebuah kondisi yang telah di berikan

#### Summary

jika ingin menampilkan baris yaitu dengan query yang dengan struktur `select [nama_kolom] form pelanggan [nama_table] where[kondisi];`

## Update

### Struktur

```
update nama_table set nama_kolom where kondisi;
```

## Contoh

```
update pelanggan set no_telpon="0801XXXX" where id_pelanggan="1";
```

## Hasil

id_pelanggan	nama_depan	nama_belakang	no_telpon
1	Ahmad	Satya	0801XXXX
2	Daud	Resky	0812XXXX
3	Resky	Alfatir	0834XXXX
4	Agis	NULL	0856XXXX
5	fahri	NULL	NULL

## Analisis

1. `update` adalah query yang digunakan untuk memperbarui nilai dari kolom
2. `pelanggan` nama table yang akan di perbarui nilai kolomnya
3. `set` query yang digunakan untuk memberikan penanda bahwa yang nilainya akan di rubah
4. `no_telpon` kolom yang akan di ubah nilai nya
5. `"0801XXXX"` nilai yang akan dimasukkan
6. `where` query yang digunakan untuk memberikan sebuah kondisi

### Kesimpulan

jika ingin mengubah atau mengupdate sebuah table kalian bisa menggunakan query `update` dengan struktur query yaitu `update nama_table set nama_kolom where kondisi;`

## Delete

### Struktur

```
delete from nama_table where kondisi;
```

## Contoh

```
delete from pelanggan where id_pelanggan=5;
```

## Hasil

id_pelanggan	nama_depan	nama_belakang	no_telpon
1	Ahmad	Satya	0801XXXX
2	Daud	Resky	0812XXXX
3	Resky	Alfatir	0834XXXX
4	Agis	NULL	0856XXXX

## Analisis

1. `delete` query yang digunakan untuk menghapus baris kolom
2. `from` query yang digunakan untuk memberikan penanda bahwa table mana yang akan di hapus baris nya
3. `pelanggan` nama table yang akan di hapus baris nya
4. `where` query yang digunakan untuk memberikan sebuah kondisi
5. `id_pelanggan` nama kolom nya
6. `=` operatornya
7. `5` nilai nya

### Kesimpulan

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

## Drop Table

### Struktur

```
Drop table [nama_table];
```

### Contoh

```
drop table pelanggan;
```

### Hasil

```
MariaDB [rental_angga]> drop table pelanggan;
Query OK, 0 rows affected (2.180 sec)
```

## Analisis

1. `drop table` query yang digunakan untuk menghapus sebuah table
2. `pelanggan` merupakan nama table yang akan di hapus

## Kesimpulan

jika ingin menghapus table kalian bisa menggunakan query `drop` dengan struktur query yaitu `drop table [nama_table];`

# Select Lanjutan

## AND

## Struktur

```
select kolom1,kolom2 from nama_table where kolom1='nilai_kolom1' and
kolom2='nilai_kolom2';
```

## Contoh

```
select warna,pemilik from mobil where warna='hitam' and pemilik='ibrahim'
```

## Hasil

warna	pemilik
Hitam	Ibrahim

## Analisis

1. `select` query yang digunakan untuk menampilkan masukan dari `insert`
2. `warna,pemilik` merupakan nama kolom dari mobil
3. `from` query yang digunakan untuk memberi tanda bahwa tabel mana yang akan di tampilkan
4. `where` query yang digunakan untuk memberikan sebuah kondisi
5. `warna='hitam' and pemilik='ibrahim'` merupakan sebuah kondisi untuk query dan `and` digunakan untuk memberikan syarat yang keduanya harus di penuhi

## Kesimpulan

jika ingin menampilkan data yang telah di seleksi dengan cara memberikan syarat yang semuanya harus di penuhi kalian bisa menggunakan query dengan struktur

```
select kolom1,kolom2 from nama_table where kolom1='nilai_kolom1' and
kolom2='nilai_kolom2';
```

# OR

## Struktur

```
select kolom1,kolom2 from nama_table where kolom1='nilai_kolom1' or kolom2='nilai_kolom2';
```

## Contoh

```
select warna,pemilik from mobil where warna='Hitam' or pemilik='Ibrahim';
```

## Hasil

```
MariaDB [rental_angga]> select * from mobil where warna='hitam' or pemilik='ibrahim';
```

id_mobil	no_plat	no_mesin	warna	pemilik	peminjam	harga_rental
1	DD 2650 XY	ACX3560	Hitam	Ibrahim	Afdal	50000
2	DD 2440 AX	BCS1120	Merah	Ibrahim	Elia	100000
4	DD 2981 JK	UQL1029	Hitam	Ibe	NULL	150000
5	DD 2210 LS	CJH1011	Hitam	Ibe	NULL	100000

## Analisis

1. `select` query yang digunakan untuk menampilkan masukan dari `insert`
2. `warna,pemilik` merupakan nama kolom dari mobil
3. `from` query yang digunakan untuk memberi tanda bahwa tabel mana yang akan di tampilkan
4. `where` query yang digunakan untuk memberikan sebuah kondisi
5. `warna='hitam' or pemilik='ibrahim'` merupakan sebuah kondisi untuk query dan `or` digunakan untuk memberikan syarat yang salah satunya harus di penuhi

### Kesimpulan

jika kalian ingin menampilkan data tabel dari kolom yang nilainya telah di seleksi dengan cara memberikan syarat yang salah satunya harus di penuhi kalian bisa menggunakan query dengan struktur `select warna,pemilik from mobil where warna='Hitam' or pemilik='Ibrahim';`

## Between

## Struktur

```
select * from nama_table where nama_kolom between nilai1 and nilai2;
```

## Contoh

```
select * from mobil where harga_rental between 100000 and 150000;
```

## Hasil

```
MariaDB [rental_angga]> select * from mobil where harga_rental between 100000 and 150000;
```

id_mobil	no_plat	no_mesin	warna	pemilik	peminjam	harga_rental
2	DD 2440 AX	BCS1120	Merah	Ibrahim	Elia	100000
4	DD 2981 JK	UQL1029	Hitam	Ibe	NULL	150000
5	DD 2210 LS	CJH1011	Hitam	Ibe	NULL	100000

## Analisis

1. `select` query yang digunakan untuk menampilkan masukan dari `insert`
2. `*` berarti semua kolom akan di tampilkan
3. `from` untuk memberikan tanda bahwa table mana yang akan di tampilkan
4. `mobil` nama table yang akan di tampilkan
5. `where` untuk memberikan sebuah kondisi
6. `harga_rental` nama kolom yang digunakan untuk mengkondisikan sebuah table
7. `between` Ini adalah operator yang digunakan untuk memilih rentang nilai
8. `100000 and 150000` Ini adalah nilai rentang yang digunakan dalam kriteria pemilihan data

### Kesimpulan

Jika ingin menampilkan hasil dari menyeleksi table dengan cara memberikan sebuah rentang nilai kalian bisa menggunakan sebuah query dengan struktur `select * from nama_table where nama_kolom between nilai1 and nilai2;`

## Not Between

### Struktur

```
select * from nama_table where nama_kolom not between nilai1 and nilai2;
```

### Contoh

```
select * from mobil where harga_rental not between 100000 and 150000;
```

## Hasil

```
MariaDB [rental_angga]> select * from mobil where harga_rental not between 100000 and 150000;
```

id_mobil	no_plat	no_mesin	warna	pemilik	peminjam	harga_rental
1	DD 2650 XY	ACX3560	Hitam	Ibrahim	Afdal	50000
3	B 1611 QC	LSQ1112	Silver	Baim	Anty	50000

## Analisis

1. `select` query yang digunakan untuk menampilkan masukan dari `insert`
2. `*` berarti semua kolom akan di tampilkan
3. `from` untuk memberikan tanda bahwa table mana yang akan di tampilkan
4. `mobil` nama table yang akan di tampilkan
5. `where` untuk memberikan sebuah kondisi
6. `harga_rental` nama kolom yang digunakan untuk mengkondisikan sebuah table
7. `not between` Ini adalah operator yang digunakan untuk memilih nilai di luar rentang tertentu.
8. `100000 and 150000` Ini adalah nilai rentang yang digunakan dalam kriteria pemilihan data

### Kesimpulan

Jika ingin menampilkan hasil dari menyeleksi table dengan cara memberikan sebuah rentang nilai yang beda nya sebelumnya itu jika nilai tersebut masih berada di dalam rentang nilai yang diberikan maka akan di tampilkan sedangkan kali ini di luar dari rentang nilai yang akan di tampilkan untuk itu kalian bisa menggunakan sebuah query dengan struktur `select * from nama_table where nama_kolom not between nilai1 and nilai2;`

<=

## Struktur

```
select * from nama_table where nama_kolom<=nilai;
```

## Contoh

```
select * from mobil where harga_rental<=50000;
```

## Hasil

```
MariaDB [rental_angga]> select * from mobil where harga_rental <= 50000;
```

id_mobil	no_plat	no_mesin	warna	pemilik	peminjam	harga_rental
1	DD 2650 XY	ACX3560	Hitam	Ibrahim	Afdal	50000
3	B 1611 QC	LSQ1112	Silver	Baim	Anty	50000



## Analisis

1. `select` query yang digunakan untuk menampilkan hasil dari `insert`
2. `*` arti nya semua kolom akan ditampilkan
3. `from` query yang digunakan untuk memberikan penanda bahwa table mana yang akan di tampilkan
4. `mobil` nama table yang akan ditampilkan
5. `where` query yang digunakan untuk memberikan sebuah kondisi
6. `harga_rental<=50000` sebuah kondisi yang telah di berikan dan `harga_rental` itu nama kolom, `<=` merupakan operator, dan `50000` merupakan sebuah nilai

### Kesimpulan

jika ingin menampilkan table dengan menggunakan hasil seleksi yang dimana jika dia lebih kecil dari nilai yang di tentukan maka dia akan tampil, yaitu dengan cara menggunakan query dengan struktur

```
select * from nama_table where nama_kolom<=nilai;
```

**>=**

## Struktur

```
select * from nama_table where nama_kolom>=nilai;
```

## Contoh

```
select * from mobil where harga_rental>=50000
```

## Hasil

```
MariaDB [rental_angga]> select * from mobil where harga_rental >= 50000;
```

id_mobil	no_plat	no_mesin	warna	pemilik	peminjam	harga_rental
1	DD 2650 XY	ACX3560	Hitam	Ibrahim	Afdal	50000
2	DD 2440 AX	BCS1120	Merah	Ibrahim	Elia	100000
3	B 1611 QC	LSQ1112	Silver	Baim	Anty	50000
4	DD 2981 JK	UQL1029	Hitam	Ibe	NULL	150000
5	DD 2210 LS	CJH1011	Hitam	Ibe	NULL	100000

## Analisis

1. `select` query yang digunakan untuk menampilkan hasil dari `insert`
2. `*` arti nya semua kolom akan ditampilkan
3. `from` query yang digunakan untuk memberikan penanda bahwa table mana yang akan di tampilkan

4. `mobil` nama table yang akan ditampilkan
5. `where` query yang digunakan untuk memberikan sebuah kondisi
6. `harga_rental >= 500000` sebuah kondisi yang telah di berikan dan `harga_rental` itu nama kolom, `>=` merupakan operator, dan `500000` merupakan sebuah nilai

#### Kesimpulan

jika ingin menampilkan table dengan menggunakan hasil seleksi yang dimana jika dia lebih besar dari nilai yang di tentukan maka dia akan tampil, yaitu dengan cara menggunakan query dengan struktur

```
select * from nama_table where nama_kolom <= nilai;
```

## <> atau !=

### Struktur1

```
select * from nama_table where nama_kolom <> nilai;
```

### Struktur2

```
select * from nama_table where nama_kolom != nilai;
```

### Contoh1

```
select * from mobil where harga_rental <> 500000;
```

### Contoh2

```
select * from mobil where harga_rental != 500000;
```

### Hasil1

```
MariaDB [rental_angga]> select * from mobil where harga_rental <> 500000;
```

id_mobil	no_plat	no_mesin	warna	pemilik	peminjam	harga_rental
2	DD 2440 AX	BCS1120	Merah	Ibrahim	Elia	100000
4	DD 2981 JK	UQL1029	Hitam	Ibe	NULL	150000
5	DD 2210 LS	CJH1011	Hitam	Ibe	NULL	100000

### Hasil2

```
MariaDB [rental_angga]> select * from mobil where harga_rental != 50000;
```

id_mobil	no_plat	no_mesin	warna	pemilik	peminjam	harga_rental
2	DD 2440 AX	BCS1120	Merah	Ibrahim	Elia	100000
4	DD 2981 JK	UQL1029	Hitam	Ibe	NULL	150000
5	DD 2210 LS	CJH1011	Hitam	Ibe	NULL	100000

## Analisis1

1. `select` query yang digunakan untuk menampilkan hasil dari `insert`
2. `*` arti nya semua kolom akan ditampilkan
3. `from` query yang digunakan untuk memberikan penanda bahwa table mana yang akan di tampilkan
4. `mobil` nama table yang akan ditampilkan
5. `where` query yang digunakan untuk memberikan sebuah kondisi
6. `harga_rental<>50000` sebuah kondisi yang telah di berikan dan `harga_rental` itu nama kolom, `<>` merupakan operator, dan `50000` merupakan sebuah nilai

## Analisis2

1. `select` query yang digunakan untuk menampilkan hasil dari `insert`
2. `*` arti nya semua kolom akan ditampilkan
3. `from` query yang digunakan untuk memberikan penanda bahwa table mana yang akan di tampilkan
4. `mobil` nama table yang akan ditampilkan
5. `where` query yang digunakan untuk memberikan sebuah kondisi
6. `harga_rental!=50000` sebuah kondisi yang telah di berikan dan `harga_rental` itu nama kolom, `!=` merupakan operator, dan `50000` merupakan sebuah nilai

### Summary

dari kedua contoh operator kita bisa menyimpulkan bahwa operator `!=` dengan `<>` memiliki arti yang sama yang dimana jika ingin menampilkan table dengan menggunakan sebuah nilai maka nilai yang ingin di tampilkan tidak boleh sama dengan nilai yang telah di tentukan

## Tantangan Login

### Struktur

```
select nama_kolom1 from nama_table where nama_kolom2=nilai;
```

### Contoh

```
select pemilik from mobil where no_plat="DD 2560 XY";
```

## Hasil

```
MariaDB [rental_angga]> select pemilik from mobil where no_plat="DD 2650 XY";
+-----+
| pemilik |
+-----+
| Ibrahim |
+-----+
```

## Analisis

1. `select` query yang digunakan untuk menampilkan sebuah table
2. `pemilik` nama kolom yang dimana hanya isi dari kolom ini yang akan di tampilkan
3. `from` query yang digunakan untuk memberikan sebuah tanda ke table yang akan di tampilkan
4. `mobil` nama table yang akan di tampilkan
5. `where` query yang digunakan untuk memberikan sebuah kondisi
6. `no_plat="DD 2560 XY"` sebuah kondisi yang telah diberikan. `no_plat` adalah nama kolom, `=` adalah operator, dan `"DD 2560 XY"` adalah nilai.

### Summary

jika ingin menampilkan dari hasil seleksi yang dimana hanya ada satu nilai dari satu kolom aka di tampilkan, yaitu dengan cara menggunakan query dengan struktur `select nama_kolom1 from nama_table where nama_kolom2=nilai;`



`nama_kolom2` harus kolom yang menggunakan constraint unique

```
create table pelanggan
(id_pelanggan int(4) primary key not null,
nama_depan varchar(25) not null,
nama_belakang varchar(25),
no_telpon char(12) unique);
```

```
create table table_guru
(id_guru int(2) PRIMARY KEY not null,
nama_depan varchar(30) not null,
nama_belakang varchar(25),
mapel varchar(30) not null,
jabatan varchar(30) unique not null,
usia int(10) not null,
tanggal_lahir date(12) not null);
```