BUKU
PROJECT
BASIS DATA II

OLEH: ARIF HASUDUNGAN SILABAN 24S04

KATA PENGANTAR

Puji syukur saya panjatkan ke hadirat Tuhan Yang Maha Esa, karena atas limpahan rahmat dan karunia-Nya, Project Praktikum ini dapat saya selesaikan dengan baik. Project Praktikum ini disusun dengan tujuan untuk memberikan pemahaman yang lebih mendalam tentang materi-materi penting dalam Basis Data II.

Saya menyadari bahwa Project Basis Data II ini merupakan salah satu pilar utama dalam pengelolaan informasi yang sangat penting di era digital saat ini. Oleh karena itu, buku ini dirancang agar dapat menjadi panduan praktis bagi mahasiswa, dosen, dan pembaca lainnya yang ingin mengembangkan kompetensinya dalam memahami konsep-konsep tersebut secara komprehensif. Dengan penjelasan yang terstruktur, diharapkan pembaca dapat lebih mudah mengikuti setiap langkah dalam praktikum dan mampu mengimplementasikan pengetahuan yang diperoleh ke dalam dunia nyata.

Saya menyadari bahwa dalam proses penyusunan buku ini masih terdapat kekurangan dan keterbatasan, baik dari segi materi maupun penyajiannya. Oleh karena itu, saya sangat mengharapkan kritik dan saran yang konstruktif dari para pembaca. Masukan tersebut akan menjadi bekal berharga bagi saya untuk terus memperbaiki dan menyempurnakan buku ini dimasa mendatang.

Akhir kata, saya mengucapkan terima kasih kepada Bapak Dr. Dedy Hartama, S.T, M.Kom dan semua pihak yang telah membantu dalam penyusunan buku ini, baik secara langsung maupun tidak langsung. Semoga buku ini dapat memberikan manfaat yang sebesar-besarnya bagi pembaca, khususnya meningkatkan wawasan dan keterampilan dalam bidang Basis Data.

Pematangsiantar, 06 Desember 2025

Arif Hasudungan Silaban

DAFTAR ISI

Kata Pengantar ..... **i**

Daftar Isi ..... **ii**

BAB I PENDAHULUAN **1**

1.1 Latar Belakang 1

1.2 Tujuan Praktikum 2

1.3 Pemahaman Dasar Basis Data 2

1.4 Struktur Buku 3

BAB II PENGELOLAAN DATABASE **5**

2.1 Memulai XAMPP Control Panel 5

2.2 Membuat Database 6

2.3 Membuat, Mengisi, dan Menampilkan Table 7

2.4 DDL (Data Definition Language) 9

A. Membuat Database 10

B. Membuat Tabel 10

C. Alter Table 13

D. Describe Table 17

Gambar Relasi Antar Tabel 20

Latihan Soal 21

2.5 DML (Data Manipulation Language) 22

A. INSERT 23

B. UPDATE 23

C. DELETE 23

D. SELECT 23

2.6 DQL (Data Query Language) 29

Latihan Soal SQL 34

BAB III EXPORT & IMPORT **36**

3.1 Export 36

3.2 Import 37
3.3 Konversi file Excel ke MySQL 37
BAB IV JOIN 46
Latihan Soal Join 50
BAB V VIEW 52
Latihan Soal View 58
BAB VI STORED PROCEDURE 59
Latihan Soal Stored Procedure 63
BAB VII TRIGGER 64
Latihan Soal Trigger 69
BAB VII GRANT & REVOKE 70
Latihan Soal Grant & Revoke 75
BAB IX ENTITY RELATIONSHIP DIAGRAM (ERD) 76
9.1 Simbol-Simbol pada ERD 77
9.2 Contoh Kardinalitas 78
9.3 Manfaat ERD dalam Perancangan Database 78
9.4 Contoh Kasus Penggunaan ERD 79
Contoh ERD Crow's Foot 80
Contoh ERD Classic Chen 81
9.5 Tabel ERD pada Sistem dbGajiGuru 82
1. Entitas dan Atribut 82
2. Relasi Antar Entitas 84
3. Implementasi Kunci Utama dan Kunci Tamu 84
4. Keuntungan Struktur ERD dan Tabel 85
BAB X KESIMPULAN DAN SARAN 86
10.1 Kesimpulan 86
10.2 Saran 86
DAFTAR PUSTAKA 88

BAB I

PENDAHULUAN

1.1 Latar Belakang

Di era modern saat ini, pengolahan data yang cepat, tepat, dan akurat menjadi kebutuhan utama dalam berbagai kegiatan, termasuk di lingkungan pendidikan. Sekolah bukan hanya sebagai tempat kegiatan belajar mengajar, tetapi juga sebagai organisasi yang memiliki banyak data yang harus diolah, seperti data guru, data mata pelajaran, data kelas, absensi, hingga perhitungan gaji. Seluruh data tersebut perlu disusun dengan baik agar mudah diakses dan diolah menjadi informasi yang bermanfaat. Pengolahan secara manual seringkali menimbulkan masalah, seperti kesalahan pencatatan, data yang tidak konsisten, serta kesulitan dalam menampilkan laporan. Oleh karena itu, diperlukan sebuah sistem basis data yang dapat membantu mengelola seluruh data tersebut secara terstruktur.

Basis data merupakan kumpulan data yang terorganisir dan saling terhubung. Dengan menggunakan sistem basis data, informasi dapat disimpan, diperbarui, dan diambil kembali secara efisien. Salah satu komponen penting dalam perancangan basis data adalah desain struktur tabel, relasi antar tabel, serta implementasinya menggunakan bahasa SQL. Mata kuliah Praktikum Basis Data II memberikan kesempatan kepada mahasiswa untuk memahami dan menerapkan konsep tersebut melalui pembuatan sistem berbasis database yang nyata. Melalui kegiatan ini, mahasiswa dapat mempelajari berbagai perintah SQL seperti DDL, DML, JOIN, VIEW, TRIGGER, PROCEDURE, dan GRANT REVOKE, yang merupakan dasar dalam pengelolaan database modern.

Proyek database pada laporan ini dirancang sebagai bentuk penerapan dari teori yang telah dipelajari, dengan studi kasus pada pengolahan data guru, absensi mengajar, dan perhitungan gaji. Seluruh data disusun ke dalam tabel yang saling terhubung menggunakan primary key dan foreign key, kemudian divisualisasikan menggunakan Entity Relationship Diagram (ERD). Hasilnya menunjukkan bahwa penggunaan sistem basis data mampu mengoptimalkan proses pengolahan data, mempermudah pembuatan laporan, dan menghindari duplikasi informasi. Dengan

adanya sistem ini, kebutuhan akan data yang akurat dan cepat dapat terpenuhi sehingga kegiatan pengelolaan sekolah dapat berjalan lebih efektif.

Melalui penyusunan proyek ini, diharapkan pembaca dapat memahami proses perancangan basis data secara menyeluruh, mulai dari analisis kebutuhan, pembuatan tabel, implementasi query, hingga pengujian hasil. Selain itu, laporan ini dapat menjadi referensi bagi mahasiswa lain yang akan melaksanakan praktikum serupa di masa mendatang. Sistem basis data bukan hanya konsep teori, tetapi juga solusi praktis yang mampu meningkatkan kinerja organisasi dalam mengelola informasi. Oleh karena itu, pembelajaran mengenai basis data sangat penting untuk mendukung kompetensi mahasiswa di bidang teknologi informasi.

1.2 Tujuan Praktikum

Tujuan dari praktikum ini adalah:

1. Membantu mahasiswa memahami konsep lanjutan basis data yang sering digunakan dalam pengembangan aplikasi nyata.
2. Memberikan pengalaman praktis dalam menerapkan operasi basis data yang melibatkan banyak tabel, seperti join dan view.
3. Mengajarkan cara membuat dan menggunakan prosedur tersimpan, trigger, serta function untuk meningkatkan efisiensi pengolahan data.
4. Menjelaskan penggunaan perintah grant untuk mengelola hak akses pengguna dalam basis data.
5. Mengembangkan kemampuan mahasiswa dalam merancang basis data yang efisien dan aman melalui praktik langsung.

1.3 Pemahaman dasar Basis Data

Sebelum masuk ke pembahasan umum, seharusnya kita sudah tau apa itu SQL. SQL adalah bahasa pemrograman yang efektif untuk mengelola database. Fungsinya meliputi eksekusi quey, pengaturan hak akses pengguna, serta manpulasi dan pengaksesan database. Penggunaan SQL mempermudah pekerjaan dengan basis data secara efisien dan terstruktur yang meliputi; membuat, mengambil, menambah, merubah, menghapus.

1.4 Struktur Buku

Buku ini disusun dalam beberapa bab yang saling berkaitan, dengan urutan sebagai berikut:

- **BAB II: Pengololaan Database** - Pengetahuan dasar yang harus dikuasai, seperti SQL dasar, pembuatan tabel, manipulasi data, dan penggunaan query sederhana.
- **BAB III: EXPORT & IMPORT –** Membahas cara praktis untuk membackup database untuk menghindari kerusakan data/file, serta membahas cara mengkoversi file excel ke MySQL.
- **BAB IV: JOIN** - Membahas jenis-jenis operasi join, seperti inner join, left join, right join, dan full join. Bab ini dilengkapi contoh kasus serta latihan untuk mempraktikkan penggunaan join dalam berbagai skenario.
- **BAB V: VIEW** - Mengulas konsep view dan cara pembuatan serta keuntungannya dalam basis data, terutama untuk menyederhanakan akses data yang kompleks.
- **BAB VI: PROCEDURE** - Menjelaskan tentang prosedur tersimpan (stored procedure), termasuk cara membuat dan menggunakan prosedur dengan parameter untuk menjalankan tugas-tugas yang berulang dalam basis data.
- **BAB VII: TRIGGER** - Mempelajari trigger, yaitu mekanisme untuk mengeksekusi otomatis perintah SQL sebagai respons terhadap perubahan dalam tabel, seperti penambahan atau penghapusan data.
- **BAB VIII: GRANT & REVOKE** - Membahas tentang manajemen hak akses dan keamanan dalam basis data, dengan menggunakan perintah grant dan revoke untuk mengontrol izin pengguna.
- **BAB IX**: **ENTITY RELATIONSHIP DIAGRAM (ERD)** - Membahas tentang pemodelan struktur data dan hubungan antar entitas dalam sebuah sistem basis data. Diagram ini digunakan untuk menggambarkan objek data, atribut yang dimiliki, serta relasi dan kardinalitas antar entitas, sehingga desain database menjadi terarah, logis, dan mudah dipahami sebelum diimplementasikan.

- **BAB X: PENUTUP** - Berisi kesimpulan dari materi yang telah dibahas dalam praktikum serta saran pengembangan bagi mahasiswa yang ingin mendalami topik lebih lanjut**.**

BAB II
PENGELOLAAN DATABASE

2.1 Memulai XAMPP Control Panel

XAMPP Control Panel adalah antarmuka utama yang digunakan untuk mengelola layanan (server) yang ada di paket XAMPP. XAMPP sendiri adalah paket software yang berisi beberapa aplikasi server penting, seperti:

Apache → web server untuk menjalankan aplikasi berbasis web.

MySQL/MariaDB → database server untuk menyimpan data.

PHP → bahasa pemrograman server-side.

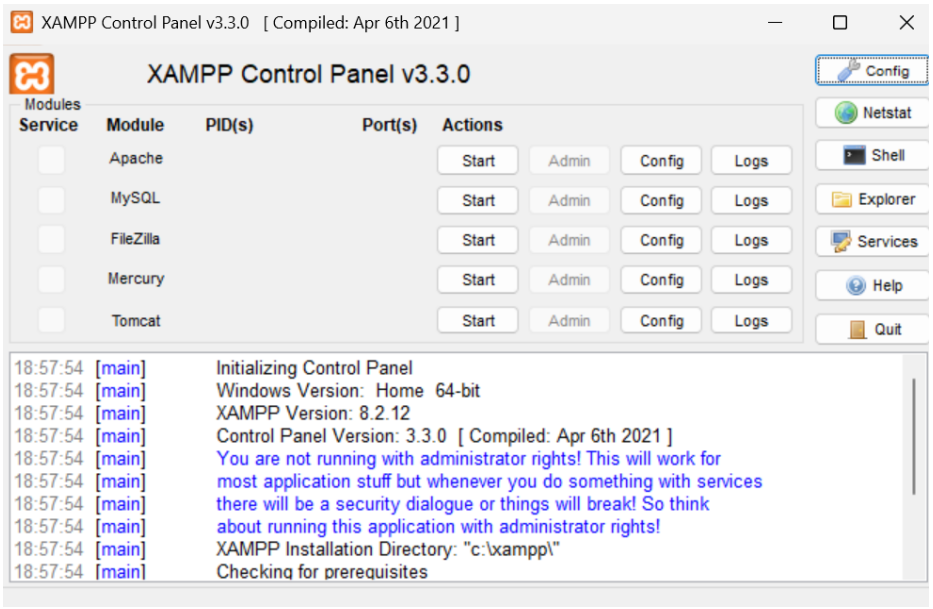
Perl → bahasa pemrograman tambahan. (Gunawan et al., 2023)

Control Panel berfungsi sebagai pusat kendali untuk:

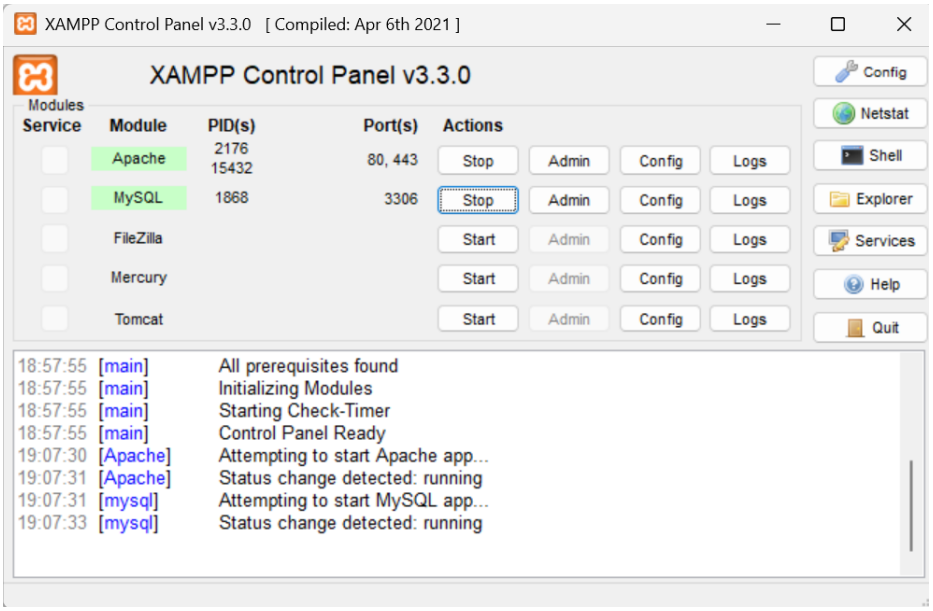
- Menyalakan / mematikan layanan (Apache, MySQL, FileZilla, Mercury, Tomcat).
- Melihat status layanan apakah sedang berjalan atau berhenti.
- Mengakses konfigurasi setiap layanan dengan mudah.
- Melihat log/error jika ada masalah saat menjalankan server.

(Ismai, 2021)

Contoh tampilannya biasanya ada tombol Start / Stop untuk masing-masing layanan, juga tombol Config, Logs, dan Admin.



Apabila ingin memulai Xampp Control Panel, klik Start pada Apache dan MySQL



Apabila module pada Apache dan MySQL sudah hijau, klik Shell untuk memulai XAMPP

Tampilan Awal:

```
Setting environment for using XAMPP for Windows.
MSI MODERN 14@MSI c:\xampp
#
```

Ketik mysql -u root untuk masuk ke MariaDB

```
Setting environment for using XAMPP for Windows.
MSI MODERN 14@MSI c:\xampp
# mysql -u root
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 8
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)]>
```

2.2 Membuat Database

Untuk memulai proses database hal pertama yang dilakukan adalah membuat database dengan syntax:

```
create database nama_db;
```

```
Query OK, 1 row affected (0.004 sec)
```

Setelah database telah dibuat selanjutkan gunakan database yang telah dibuat, selanjutnya kita harus menggunakan database yang telah dibuat dengan syntax:

```
use nama_db;
Database changed
```

Setelah melakukan hal ini maka database siap digunakan dan diisi dengan table table yang dibutuhkan.

2.3 Membuat, Mengisi dan Menampilkan Table

Untuk Membuat Table menggunakan syntax:

```
create table tblkelas(
    -> id_nama int not null PRIMARY KEY,
    -> nama Varchar (50) NOT NULL);
Query OK, 0 rows affected (0.015 sec)
```

Tampak Querry Ok, setelah membuat table berhasil selanjutnya syntax untuk menampilkan table yang ada pada database syntax:

```
show tables;
+---------------------+
| Tables_in_nama_db   |
+---------------------+
| nama_table          |
+---------------------+
1 rows in set (0.005 sec)
```

Dan jika ingin mengetahui isi field didalam table anda bisa memanggilnya dengan syntax:

```
desc nama_table;
```

```
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| id_nama  | int(11)     | NO   | PRI | NULL    |       |
| nama     | varchar(50) | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
2 rows in set (0.002 sec)
```

Setelah table dibuat dan kolom field telah ditentukan maka selanjutnya adalah memasukkan record data pada table dengan syntax:

```
insert into nama_table values
    -> (1,'Arif'),
    -> (2,'Despi'),
    -> (3,'Kessha');
Query OK, 3 rows affected (0.045 sec)
Records: 3  Duplicates: 0  Warnings: 0
```

Untuk Menmapilkan table menggunakan syntax:

```
select * from nama_table;
+-----------+---------+
|  id_nama  |  nama   |
+-----------+---------+
|         1 |  Arif   |
|         2 |  Despi  |
|         3 |  Kessha |
+-----------+---------+
3 rows in set (0.001 sec)
```

Jika ingin melakukan perubahan pada isi record, maka syntax yang digunakan adalah:

```
Update nama_table
   -> set nama ='Ananda'
   -> where id_nama ='2';
      Query OK, 1 row affected (0.066 sec)
      Rows matched: 1  Changed: 1  Warnings: 0
```

Lihat kembali yang telah berubah dengan syntax yang sama seperti sebelumnya yaitu:

```
select * from nama_table;
+----------+---------+
|  id_nama  |  nama   |
+----------+---------+
|         1 |  Arif   |
|         2 |  Ananda |
|         3 |  Kessha |
+----------+---------+
3 rows in set (0.001 sec)
```

Itu tadi adalah cara membuat table mengisi dan menampilkan record pada database yang ingin dibuat.

2.4 DDL (Data Definition Language)

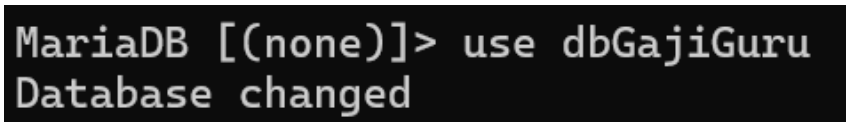
DDL (Data Definition Language) adalah bagian dari SQL yang digunakan untuk membuat, mengubah, dan menghapus struktur database dan tabel. DDL berfokus pada *definisi* atau *desain* struktur, bukan pada pengolahan datanya. Perintah-perintah utamanya meliputi CREATE, ALTER, DROP, RENAME, dan TRUNCATE (Budayawan et al., 2023). Perintah CREATE digunakan untuk membuat database, tabel, atau objek baru, misalnya CREATE TABLE tblguru (...). Perintah ALTER berfungsi untuk mengubah struktur tabel seperti menambah kolom, menghapus kolom, atau mengubah tipe data. DROP digunakan untuk menghapus tabel atau database beserta seluruh isinya secara permanen. RENAME dipakai untuk mengganti nama tabel atau kolom, sedangkan TRUNCATE digunakan untuk menghapus seluruh isi tabel tanpa menghapus struktur tabelnya. DDL diproses langsung oleh sistem database sehingga setiap perubahan bersifat permanen dan biasanya tidak dapat dibatalkan (Server & Hartama, 2000). Karena itu, DDL merupakan dasar penting dalam membangun database sebelum data dimasukkan menggunakan DML atau diambil menggunakan DQL.

Dalam sistem basis data, database adalah wadah utama yang menyimpan kumpulan tabel. Sebuah tabel berisi data yang sudah terstruktur dalam baris dan kolom. Database dapat dianalogikan sebagai lemari besar, sementara tabel adalah laci di dalam lemari tersebut. Satu database dapat memiliki banyak tabel yang saling berhubungan melalui primary key dan foreign key. Hubungan antar tabel ini yang memungkinkan data dapat dianalisa dan ditampilkan dalam bentuk laporan yang lebih kompleks. Tanpa struktur yang benar, database tidak akan optimal dan dapat menyebabkan duplikasi ataupun inkonsistensi data.

A. Membuat Database

Untuk memulai proses database hal pertama yang dilakukan adalah membuat database dengan syntax:

Soal :	Soal DDL – 01
Syntax :	Create database dbGajiGuru;
Maksud :	Membuat database dengan nama dbGajiGuru
Hasil	

Soal :	Soal DDL – 02
Syntax :	use dbGajiGuru;
Maksud :	Masuk ke database dbGajiGuru untuk digunakan dan diisi
Hasil	

B. Membuat Tabel

Soal :	Soal DDL – 03.B
Syntax :	create table tbllogin(-> Username Varchar(20) PRIMARY KEY, -> Password Varchar(20) NULL);
Maksud :	Membuat Tabel tbllogin

Hasil	MariaDB [dbGajiGuru]> create table tbllogin(-> Username Varchar(20) PRIMARY KEY, -> Password Varchar(20) NULL); Query OK, 0 rows affected (0.017 sec)

Soal :	Soal DDL – 03.A
Syntax :	create table tblguru(-> Kd_Guru int(10) PRIMARY KEY, -> Nama Varchar(20) NULL, -> Tmp_Lahir Varchar(50) NULL, -> Tgl_Lahir DATE NULL, -> Jenkel Enum('Pria','Wanita') NULL, -> Agama Varchar(20) NULL, -> Status_Serti Varchar(20) NULL, -> Ijazah_Terakhir Varchar(10) NULL, -> Tmt Varchar(10) NULL, -> Alamat Varchar(50) NULL, -> No_Telp Varchar(12) NULL);
Maksud :	Membuat Tabel tblguru
Hasil	MariaDB [dbGajiGuru]> create table tblguru(-> Kd_Guru int(10) PRIMARY KEY, -> Nama Varchar(20) NULL, -> Tmp_Lahir Varchar(50) NULL, -> Tgl_Lahir DATE NULL, -> Jenkel Enum('Pria','Wanita') NULL, -> Agama Varchar(20) NULL, -> Status_Serti Varchar(20) NULL, -> Ijazah_Terakhir Varchar(10) NULL, -> Tmt Varchar(10) NULL, -> Alamat Varchar(50) NULL, -> No_Telp Varchar(12) NULL); Query OK, 0 rows affected (0.019 sec)

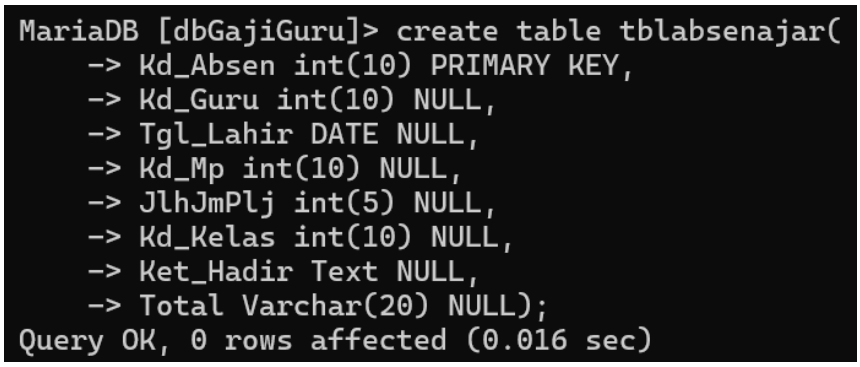
Soal :	Soal DDL – 03.B
Syntax :	create table tblmatapelajaran(-> Kd_Mp int(10) PRIMARY KEY, -> Nama_Mp Varchar(50) NULL, -> Kd_Guru int(10) NULL);
Maksud :	Membuat Tabel tblmatapelajaran
Hasil	

```
MariaDB [dbGajiGuru]> create table tblmatapelajaran(
    -> Kd_Mp int(10) PRIMARY KEY,
    -> Nama_Mp Varchar(50) NULL,
    -> Kd_Guru int(10) NULL);
Query OK, 0 rows affected (0.017 sec)
```

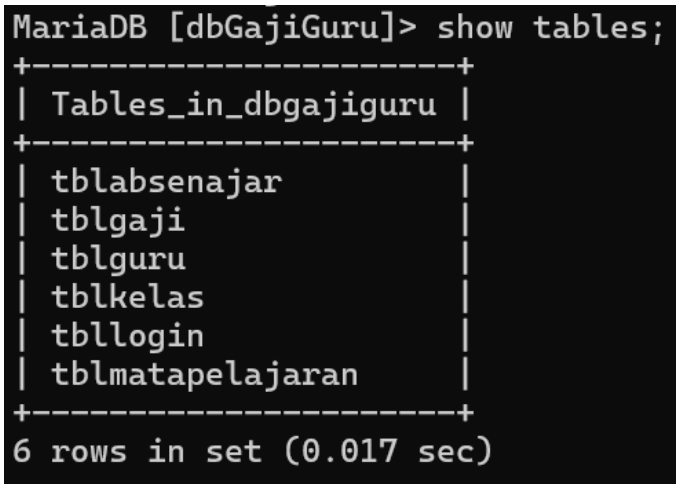
Soal :	Soal DDL – 03.C
Syntax :	create table tblkelas(-> Kd_Kelas int(10) PRIMARY KEY, -> Nm_Kelas Varchar(20) NULL);
Maksud :	Membuat Tabel tblkelas
Hasil	MariaDB [dbGajiGuru]> create table tblkelas(-> Kd_Kelas int(10) PRIMARY KEY, -> Nm_Kelas Varchar(20) NULL); Query OK, 0 rows affected (0.015 sec)

Soal :	Soal DDL – 03.D
Syntax :	create table tblgaji(-> Kd_Gaji int(10) PRIMARY KEY, -> Kd_Guru int(10) NULL, -> Kd_Absen int(10) NULL, -> TotJlhJamPlj int(10) NULL, -> GajiPrJam int(10) NULL, -> KetHonorTambahan TEXT NULL, -> JlhTambahan int(20) NULL, -> Pot int(10) NULL, -> Tot_Gaji int(20) NULL);
Maksud :	Membuat Tabel tblgaji
Hasil	MariaDB [dbGajiGuru]> create table tblkelas(-> Kd_Kelas int(10) PRIMARY KEY, -> Nm_Kelas Varchar(20) NULL); Query OK, 0 rows affected (0.015 sec)

Soal :	Soal DDL – 03.E
Syntax :	create table tblabsenajar(-> Kd_Absen int(10) PRIMARY KEY, -> Kd_Guru int(10) NULL, -> Tgl_Lahir DATE NULL, -> Kd_Mp int(10) NULL, -> JlhJmPlj int(5) NULL, -> Kd_Kelas int(10) NULL, -> Ket_Hadir Text NULL, -> Total Varchar(20) NULL);

Maksud :	Membuat Tabel tblabsenajar
Hasil	

```
MariaDB [dbGajiGuru]> create table tblabsenajar(
    -> Kd_Absen int(10) PRIMARY KEY,
    -> Kd_Guru int(10) NULL,
    -> Tgl_Lahir DATE NULL,
    -> Kd_Mp int(10) NULL,
    -> JlhJmPlj int(5) NULL,
    -> Kd_Kelas int(10) NULL,
    -> Ket_Hadir Text NULL,
    -> Total Varchar(20) NULL);
Query OK, 0 rows affected (0.016 sec)
```

Soal :	Soal DDL – 04
Syntax :	Show tables;
Maksud :	Menampilkan semua tabel yang telah dibuat dalam database
Hasil	

```
MariaDB [dbGajiGuru]> show tables;
+---------------------+
| Tables_in_dbgajiguru |
+---------------------+
| tblabsenajar        |
| tblgaji             |
| tblguru             |
| tblkelas            |
| tbllogin            |
| tblmatapelajaran    |
+---------------------+
6 rows in set (0.017 sec)
```

C. Alter Table

ALTER TABLE adalah perintah dalam SQL yang digunakan untuk mengubah struktur tabel setelah tabel tersebut dibuat. Pengubahan ini dapat meliputi penambahan kolom baru, mengubah tipe data kolom, mengganti nama kolom, menambah atau menghapus constraint seperti PRIMARY KEY dan FOREIGN KEY, hingga menghapus kolom dari tabel.

Constraint adalah aturan yang diterapkan pada kolom tabel untuk menjaga integritas data. Jenis constraint yang sering digunakan antara lain PRIMARY KEY, FOREIGN KEY, NOT NULL, UNIQUE, dan CHECK. PRIMARY KEY

berfungsi mengidentifikasi baris secara unik, sedangkan FOREIGN KEY memastikan bahwa hubungan antar tabel valid. Dengan adanya constraint, kesalahan input dapat dicegah sejak awal sehingga data dalam tabel tetap konsisten, akurat, dan bebas dari duplikasi. Ketika constraint tidak dipasang, data dapat saling bertabrakan dan menimbulkan anomali yang menyulitkan proses analisis. Misalnya, ketika membutuhkan data tambahan pada tabel, kita dapat menggunakan perintah:

```
`ALTER TABLE nama_tabel ADD nama_kolom tipe_data`
```

Untuk menambahkan kolom baru. Jika ingin mengubah tipe data atau panjang karakter, digunakan perintah:

```
`ALTER TABLE nama_tabel MODIFY nama_kolom tipe_data_baru`.
```

Untuk mengganti nama kolom, beberapa database menggunakan:

```
`ALTER TABLE nama_tabel RENAME COLUMN kolom_lama TO kolom_baru`.
```

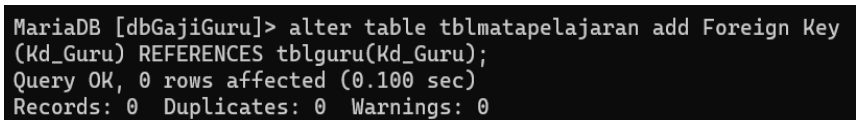
Selain itu, jika perlu menghapus kolom yang tidak digunakan lagi, dapat dipakai perintah:

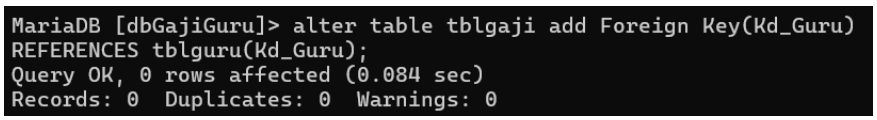
```
`ALTER TABLE nama_tabel DROP COLUMN nama_kolom`.
```

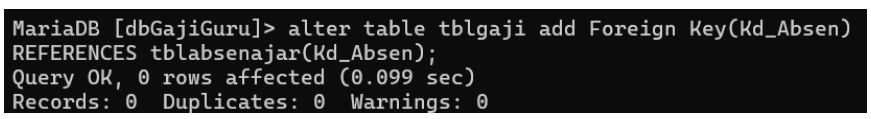
Penggunaan alter:

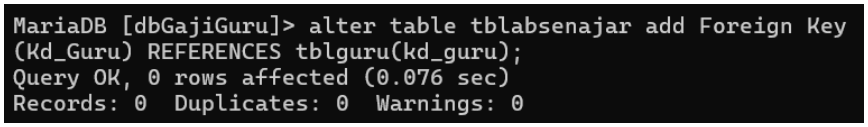
1. Alter Table Add Column → Menambahkan kolom baru pada tabel.
2. Alter Table Add Foreign Key → Menambahkan Foreign key sebagai penghubung Primary Key dengan syarat tipe dan ukuran data yang sama.
3. Alter Table Add Primary Key → Menambahkan Primary key pada kolom.
4. Alter Table Add…..First → Menambahkan kolom baru pada tabel yang posisinya berada di paling atas.
5. Alter Table Add…..After → Menambahkan kolom baru pada tabel yang posisinya berada setelah kolom yang ada.
6. Alter Table Add…..Before → Menambahkan kolom baru pada tabel yang posisinya berada sebelum di bawah kolom yang ada.
7. Alter Table Modify → Mengubah size dari tipe data pada tabel tertentu.

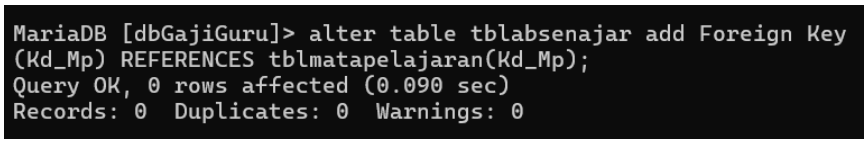
8. Alter Table Rename → Mengubah nama tabel tertentu.
9. Alter Table Change → Mengubah kolom tertentu menjadi kolom yang berbeda.
10. Alter Table Drop → Menghapus kolom tertentu di dalam tabel.(Samsoni et al., 2021)

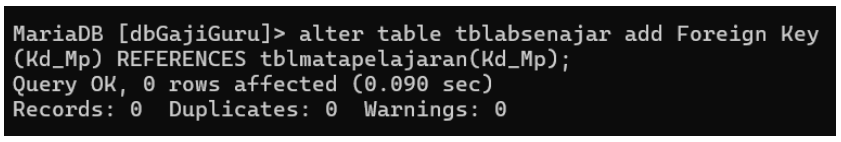
Soal :	Soal DDL – 05
Syntax :	Alter table tblmatapelajaran add Foreign Key(Kd_Guru) REFERENCES tblguru(Kd_Guru);
Maksud :	Membuat Foreign Key pada Kolom Kd_Guru yang ada pada tabel tblmatapelajaran dan dihubungkan pada kolom Kd_Guru yang ada pada tabel tblguru sebagai sumbernya.
Hasil	

Soal :	Soal DDL – 05.A
Syntax :	alter table tblgaji add Foreign Key(Kd_Guru) REFERENCES tblguru(Kd_guru);
Maksud :	Membuat Foreign Key pada Kolom Kd_Guru yang ada pada tabel tblgaji dan dihubungkan pada kolom Kd_Guru yang ada pada tabel tblgaji sebagai sumbernya.
Hasil	

Soal :	Soal DDL – 05.B
Syntax :	alter table tblgaji add Foreign Key(Kd_Absen) REFERENCES tblabsenajar(Kd_Absen);
Maksud :	Membuat Foreign Key pada Kolom Kd_Absen yang ada pada tabel tblgaji dan dihubungkan pada kolom Kd_Absen yang ada pada tabel tblabsenajar sebagai sumbernya.
Hasil	

Soal :	Soal DDL – 05.C
Syntax :	alter table tblabsenajar add Foreign Key(Kd_Guru) REFERENCES tblguru(kd_guru);
Maksud :	Membuat Foreign Key pada Kolom Kd_Guru yang ada pada tabel tblabsenajar dan dihubungkan pada kolom Kd_Guru yang ada pada tabel tblguru sebagai sumbernya.
Hasil	

Soal :	Soal DDL – 05.D
Syntax :	alter table tblabsenajar add Foreign Key(Kd_Kelas) REFERENCES tblkelas(Kd_Kelas);
Maksud :	Membuat Foreign Key pada Kolom Kd_Mp yang ada pada tabel tblabsenajar dan dihubungkan pada kolom Kd_Mp yang ada pada tabel tblmatapelajaran sebagai sumbernya.
Hasil	

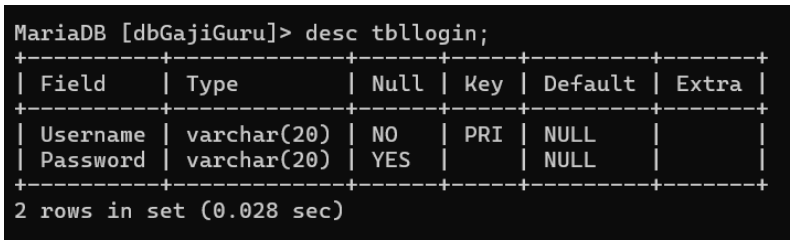
Soal :	Soal DDL – 05.E
Syntax :	alter table tblabsenajar add Foreign Key(Kd_Mp) REFERENCES tblmatapelajaran(Kd_Mp);
Maksud :	Membuat Foreign Key pada Kolom Kd_Mp yang ada pada tabel tblabsenajar dan dihubungkan pada kolom Kd_Mp yang ada pada tabel tblmatapelajaran sebagai sumbernya.
Hasil	

D. Describe Table

DESCRIBE (atau DESC) adalah perintah dalam SQL yang digunakan untuk menampilkan informasi struktur sebuah tabel tanpa harus membuka atau mengedit data di dalamnya. Perintah ini sering dipakai untuk melihat daftar kolom, tipe data setiap kolom, apakah kolom tersebut mengizinkan nilai NULL, apakah memiliki key seperti PRIMARY KEY atau FOREIGN KEY, serta informasi tambahan seperti default value. Contohnya, dengan mengetik

```
DESCRIBE nama_tabel;
```

Sistem database akan menampilkan detail lengkap mengenai kolom yang ada pada tabel tersebut. Perintah ini sangat membantu ketika seorang developer atau administrator ingin memahami struktur tabel sebelum menjalankan query lain, terutama pada database yang memiliki banyak kolom atau ketika bekerja dengan tabel yang bukan dibuat sendiri.

Soal :	Soal DDL – 06
Syntax :	Desc tbllogin;
Maksud :	Menampilkan deskripsi kolom dari tabel tbllogin beserta atribut atribut lain yang dimiliki setiap kolom
Hasil	

```
MariaDB [dbGajiGuru]> desc tbllogin;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| Username | varchar(20) | NO   | PRI | NULL    |       |
| Password | varchar(20) | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
2 rows in set (0.028 sec)
```

Soal :	Soal DDL – 06.A
Syntax :	Desc tblguru;
Maksud :	Menampilkan deskripsi kolom dari tabel tblguru beserta atribut atribut lain yang dimiliki setiap kolom
Hasil	

```
MariaDB [dbGajiGuru]> desc tblguru;
+-----------------+----------------------+------+-----+---------+-------+
| Field           | Type                 | Null | Key | Default | Extra |
+-----------------+----------------------+------+-----+---------+-------+
| Kd_Guru         | int(10)              | NO   | PRI | NULL    |       |
| Nama            | varchar(20)          | YES  |     | NULL    |       |
| Tmp_Lahir       | varchar(50)          | YES  |     | NULL    |       |
| Tgl_Lahir       | date                 | YES  |     | NULL    |       |
| Jenkel          | enum('Pria','Wanita') | YES  |     | NULL    |       |
| Agama           | varchar(20)          | YES  |     | NULL    |       |
| Status_Serti    | varchar(20)          | YES  |     | NULL    |       |
| Ijazah_Terakhir | varchar(10)          | YES  |     | NULL    |       |
| Tmt             | varchar(10)          | YES  |     | NULL    |       |
| Alamat          | varchar(50)          | YES  |     | NULL    |       |
| No_Telp         | varchar(12)          | YES  |     | NULL    |       |
+-----------------+----------------------+------+-----+---------+-------+
11 rows in set (0.021 sec)
```

Soal :	Soal DDL – 06.B
Syntax :	Desc tblmatapelajaran;
Maksud :	Menampilkan deskripsi kolom dari tabel tblmatapelajaran beserta atribut atribut lain yang dimiliki setiap kolom
Hasil	

```
MariaDB [dbGajiGuru]> desc tblmatapelajaran;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| Kd_Mp    | int(10)     | NO   | PRI | NULL    |       |
| Nama_Mp  | varchar(50) | YES  |     | NULL    |       |
| Kd_Guru  | int(10)     | YES  | MUL | NULL    |       |
+----------+-------------+------+-----+---------+-------+
3 rows in set (0.023 sec)
```

Soal :	Soal DDL – 06.C
Syntax :	Desc tblkelas;
Maksud :	Menampilkan deskripsi kolom dari tabel tblkelas beserta atribut atribut lain yang dimiliki setiap kolom
Hasil	

```
MariaDB [dbGajiGuru]> desc tblkelas;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| Kd_Kelas  | int(10)     | NO   | PRI | NULL    |       |
| Nm_Kelas  | varchar(20) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
2 rows in set (0.033 sec)
```

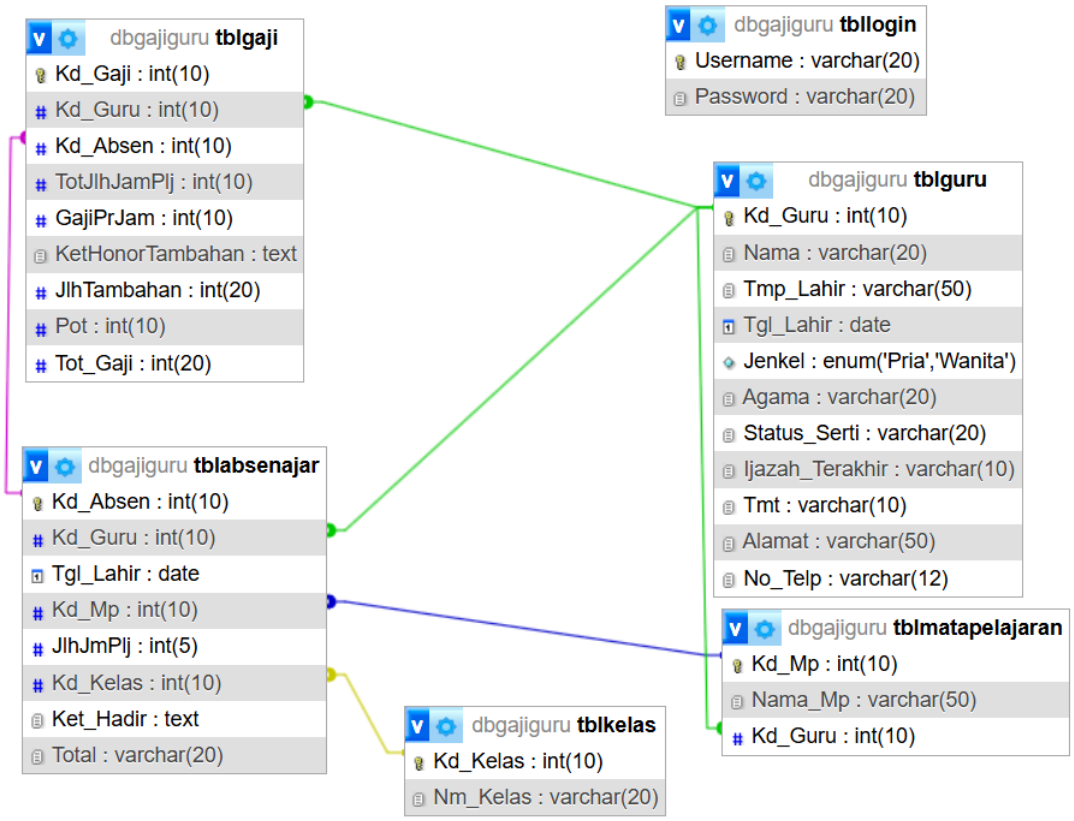
Soal :	Soal DDL – 06.D
Syntax :	Desc tblgaji;
Maksud :	Menampilkan deskripsi kolom dari tabel tblgaji beserta atribut atribut lain yang dimiliki setiap kolom
Hasil	

```
MariaDB [dbGajiGuru]> desc tblgaji;
+------------------+---------+------+-----+---------+-------+
| Field            | Type    | Null | Key | Default | Extra |
+------------------+---------+------+-----+---------+-------+
| Kd_Gaji          | int(10) | NO   | PRI | NULL    |       |
| Kd_Guru          | int(10) | YES  | MUL | NULL    |       |
| Kd_Absen         | int(10) | YES  | MUL | NULL    |       |
| TotJlhJamPlj     | int(10) | YES  |     | NULL    |       |
| GajiPrJam        | int(10) | YES  |     | NULL    |       |
| KetHonorTambahan | text    | YES  |     | NULL    |       |
| JlhTambahan      | int(20) | YES  |     | NULL    |       |
| Pot              | int(10) | YES  |     | NULL    |       |
| Tot_Gaji         | int(20) | YES  |     | NULL    |       |
+------------------+---------+------+-----+---------+-------+
9 rows in set (0.024 sec)
```

Soal :	Soal DDL – 06.E
Syntax :	Desc tblabsenajar;
Maksud :	Menampilkan deskripsi kolom dari tabel tblabsenajar beserta atribut atribut lain yang dimiliki setiap kolom
Hasil	

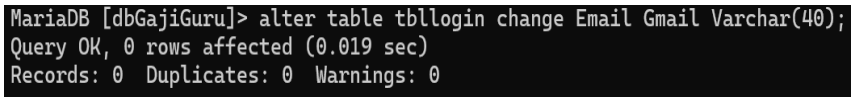
```
MariaDB [dbGajiGuru]> desc tblabsenajar;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| Kd_Absen  | int(10)     | NO   | PRI | NULL    |       |
| Kd_Guru   | int(10)     | YES  | MUL | NULL    |       |
| Tgl_Lahir | date        | YES  |     | NULL    |       |
| Kd_Mp     | int(10)     | YES  | MUL | NULL    |       |
| JlhJmPlj  | int(5)      | YES  |     | NULL    |       |
| Kd_Kelas  | int(10)     | YES  | MUL | NULL    |       |
| Ket_Hadir | text        | YES  |     | NULL    |       |
| Total     | varchar(20) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
8 rows in set (0.022 sec)
```

Gambar Relasi Antar Tabel DBGAJIGURU:

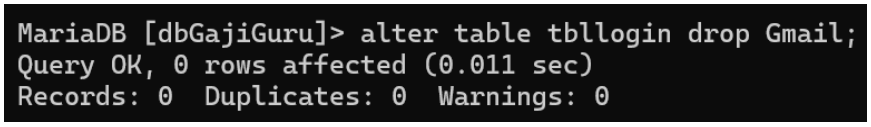


Soal :	Soal DDL – 07
Syntax :	Alter table tbllogin add Email Varchar(50) NULL;
Maksud :	Menambahkan kolom Email pada tabel tbllogin
Hasil	MariaDB [dbGajiGuru]> alter table tbllogin add Email Varchar(50) NULL; Query OK, 0 rows affected (0.042 sec) Records: 0 Duplicates: 0 Warnings: 0

Soal :	Soal DDL – 08
Syntax :	Alter table tbllogin Modify Email Varchar(40);
Maksud :	Mengubah Size tipe data dari kolom Email menjadi 40
Hasil	MariaDB [dbGajiGuru]> alter table tbllogin modify Email Varchar(40); Query OK, 0 rows affected (0.103 sec) Records: 0 Duplicates: 0 Warnings: 0

Soal :	Soal DDL – 09
Syntax :	Alter table tbllogin Change Email Gmail Varchar(40);
Maksud :	Mengubah kolom Email menjadi Gmail
Hasil	

```
MariaDB [dbGajiGuru]> alter table tbllogin change Email Gmail Varchar(40);
Query OK, 0 rows affected (0.019 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Soal :	Soal DDL – 10
Syntax :	Alter table tbllogin drop Gmail;
Maksud :	Menghapus kolom Gmail dari tabel tbllogin
Hasil	

```
MariaDB [dbGajiGuru]> alter table tbllogin drop Gmail;
Query OK, 0 rows affected (0.011 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Latihan Soal:

Setelah kamu mempraktikkan Soal-soal, boleh dong latihan sedikit, berikut soal-soal latihan:

1. Buat Database dengan nama dbBengkelUmum

2. Buat tabel tblpelanggan dengan struktur berikut:
   Kd_Pelanggan Varchar(15) PK AUTO_INCREMENT, Nama_Pelanggan VARCHAR(50), No_Telp VARCHAR(15), Alamat VARCHAR(100).

3. Buat tabel tblmekanik dengan struktur berikut:
   Kd_Mekanik VARCHAR(15) PK, Nama_Mekanik VARCHAR(50), Keahlian VARCHAR(50), No_Telp VARCHAR(15)

4. Buat tabel tblsukucadang dengan struktur berikut:

Kd_Sparepart Varchar(15) PK, Nama_Sparepart VARCHAR(50), Harga BIGINT(20), Stok INT(10)

5. Buat tabel tblservis dengan struktur berikut:
   Kd_Servis Varchar(15) PK, Kd_Pelanggan INT(15) FK, Kd_Mekanik Varchar(15) FK, Keluhan TEXT, Tgl_Servis DATE

6. Buat tabel tbldetailservis dengan struktur berikut:
   Kd_Detail VARCHAR(15), PRIMARY KEY, Kd_Servis Varchar(15) FK, Kd_Sparepart → VARCHAR(15) FK, Jumlah INT(10), Total_Harga INT(20)

7. Lakukan perintah ALTER TABLE berikut:

   a. Tambahkan kolom Biaya_Jasa (INT(20)) pada tabel tblservis.

   b. Ubah tipe data kolom No_Telp di tblpelanggan menjadi VARCHAR(20).

   c. Hapus kolom Keahlian pada tabel tblmekanik.

   d. Tambahkan relasi kunci asing:

   tblservis.Kd_Pelanggan → tblpelanggan(Kd_Pelanggan)

   tblservis.Kd_Mekanik → tblmekanik(Kd_Mekanik)

   tbldetailservis.Kd_Servis → tblservis(Kd_Servis)

   tbldetailservis.Kd_Sparepart → tblsukucadang(Kd_Sparepart)

8. Setelah direlasikan, tampilkan gambar RAT pada Domain PhpMyAdmin.

2.5 DML (Data Manipulation Language)

DML (Data Manipulation Language) adalah bagian dari SQL yang digunakan untuk mengolah atau memanipulasi data di dalam tabel. DML berfungsi untuk menambah, mengubah, dan menghapus data tanpa mengubah struktur tabel. Perintah-perintah utama dalam DML meliputi INSERT, UPDATE, dan DELETE (Silalahi, 2022).

A. INSERT

INSERT digunakan untuk menambahkan data kedalam table. Ada 5 jenis penggunaan INSERT yaitu:

1. INSERT INTO WITH NO COLUMN → Memasukkan data ke tabel tanpa menyebut nama kolom.
2. INSERT INTO WITH COLUMN → Memasukkan data dengan menyebut nama kolom yang akan diisi.
3. INSERT INTO SET → Memasukkan data dengan Set Kolom = Nilai.
4. INSERT INTO VALUES → Cara memasukkan data dengan VALUES, bisa untuk satu atau banyak baris.
5. INSERT INTO SELECT → Memasukkan data ke tabel dengan cara mengambil data dari tabel lain. (Anggoro Dimas Aryo et al., 2021)

B. UPDATE

UPDATE digunakan untuk mengubah record dalam table.

Syntax:

```
UPDATE (nama_table) SET (nama_kolom) = (nilai_baru) WHERE (nama_kolom_unik) = (Nilai_unik);
```
(Amin, 2022)

C. DELETE

DELETE digunakan untuk menghapus record dari table.

Syntax:

```
DELETE FROM (nama_table) WHERE (nama_kolom_unik) = (nilai_unik);
```

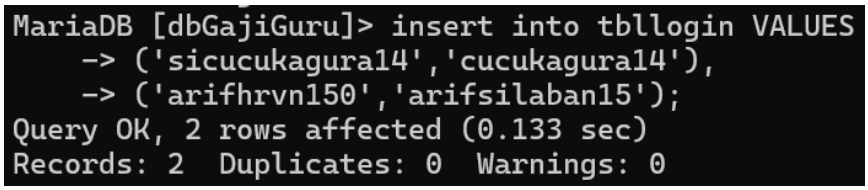
D. SELECT

SELECT digunakan untuk menampilkan data atau record pada table.

Ada beberapa jenis select yaitu:

1. SELECT … AS → Menampilkan data sambil mengganti nama kolom (alias).
2. SELECT …ORDER BY → Menampilkan data dengan urutan asc atau desc.

3. SELECT …LIMIT → Menampilkan data dengan batas jumlah tertentu.
4. SELECT …WHERE → Menampilkan data dengan syarat tertentu.
5. SELECT …WHERE…IN → Menampilkan data yang kolomnya cocok dengan beberapa nilai dalam daftar.
6. SELECT…WHERE…BETWEEN → Menampilkan data dengan rentang nilai.
7. SELECT…WHERE…LIKE → Menampilkan data dengan pola teks tertentu.
8. SELECT DISTINCT → Menampilkan data dengan menghilangkan duplikasi.
9. SELECT…GROUP BY → Mengelompokkan data berdasarkan kolom tertentu.
10. SELECT AND,OR,NOT → Menampilkan data dengan kondisi logika.
11. SELECT AGREGATE → Menampilkan data dengan fungsi agregat seperti SUM, AVG, MAX, MIN, COUNT. (Garcia et al., n.d.)

Soal :	Soal DML – 01
Syntax :	insert into tbllogin VALUES -> ('sicucukagura14','cucukagura14'), -> ('arifhrvn150','arifsilaban15');
Maksud :	Menambahkan beberapa record ke dalam tabel tbllogin
Hasil	

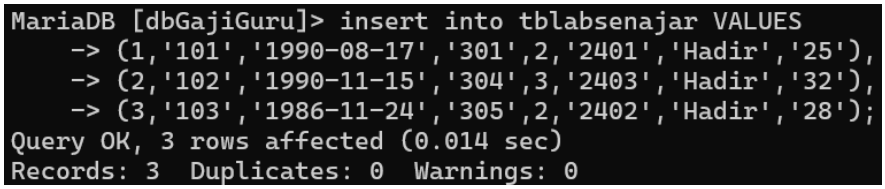
Soal :	Soal DML – 01.A
Syntax :	insert into tblguru VALUES -> (101,'Harly Okprana','Paris','1990-08 17','Pria', 'Islam','Sudah Sertifikasi','Magister','UPP', 'Pematangsiantar','089636808128'), -> (102,'Indra Gunawan','Jerman','1990-11-15', 'Pria', 'Islam','Sudah Sertifikasi','Magister','USU', 'Pematangsiantar','081278498721'), -> (103,'Surya Darma','Indonesia','1986-11-24', 'Pria', 'Islam','Sudah Sertifikasi','Magister','UPP', 'Pematangsiantar','087983213451');
Maksud :	Menambahkan beberapa record ke dalam tabel tblguru
Hasil	

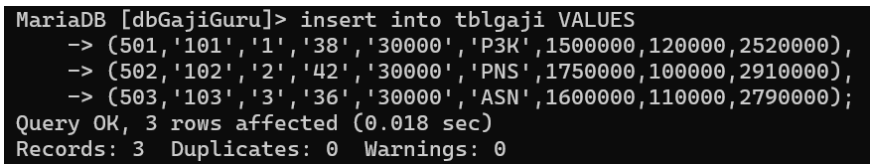
```
MariaDB [dbGajiGuru]> insert into tblguru VALUES
    -> (101,'Harly Okprana','Paris','1990-08-17','Pria','Islam','Sudah
Sertifikasi','Magister','UPP','Pematangsiantar','089636808128'),
    -> (102,'Indra Gunawan','Jerman','1990-11-15','Pria','Islam','Sudah
 Sertifikasi','Magister','USU','Pematangsiantar','081278498721'),
    -> (103,'Surya Darma','Indonesia','1986-11-24','Pria','Islam','Suda
h Sertifikasi','Magister','UPP','Pematangsiantar','087983213451');
Query OK, 3 rows affected (0.020 sec)
Records: 3  Duplicates: 0  Warnings: 0
```

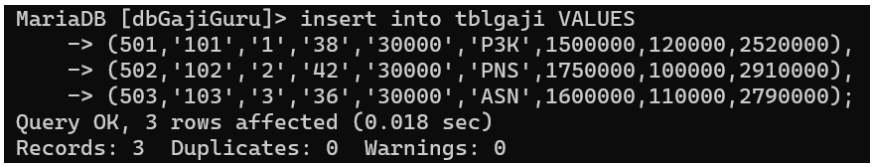
Soal :	Soal DML – 01.B
Syntax :	insert into tblkelas VALUES -> (2401,'24S04 SIKOPAT'), -> (2402,'24S05 SIKOMA'), -> (2403,'24S06 SIKONAM');
Maksud :	Menambahkan beberapa record ke dalam tabel tblkelas
Hasil	MariaDB [dbGajiGuru]> insert into tblkelas VALUES -> (2401,'24S04 SIKOPAT'), -> (2402,'24S05 SIKOMA'), -> (2403,'24S06 SIKONAM'); Query OK, 3 rows affected (0.007 sec) Records: 3 Duplicates: 0 Warnings: 0

Soal :	Soal DML – 01.C
Syntax :	insert into tblmatapelajaran VALUES -> (301,'Informatika','101'), -> (302,'Manajemen','103'), -> (303,'Programming','101'), -> (304,'Desain Grafis','102'), -> (305,'TKJ','103'), -> (306,'UI/UX','102');
Maksud :	Menambahkan beberapa record ke dalam tabel tblmatapelajaran
Hasil	MariaDB [dbGajiGuru]> insert into tblmatapelajaran VALUES -> (301,'Informatika','101'), -> (302,'Manajemen','103'), -> (303,'Programming','101'), -> (304,'Desain Grafis','102'), -> (305,'TKJ','103'), -> (306,'UI/UX','102'); Query OK, 6 rows affected (0.015 sec) Records: 6 Duplicates: 0 Warnings: 0

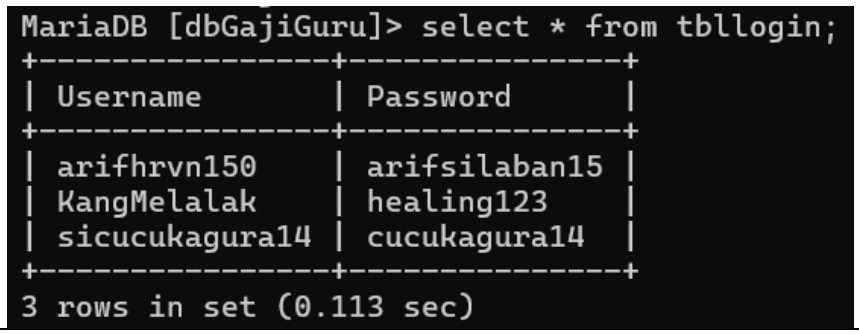
Soal :	Soal DML – 01.D
Syntax :	insert into tblabsenajar VALUES -> (1,'101','1990-08-17','301',2,'2401', 'Hadir','25'), -> (2,'102','1990-11-15','304',3,'2403', 'Hadir','32'),

	-> (3,'103','1986-11-24','305',2,'2402', 'Hadir','28');
Maksud :	Menambahkan beberapa record ke dalam tabel tblabsenajar
Hasil	

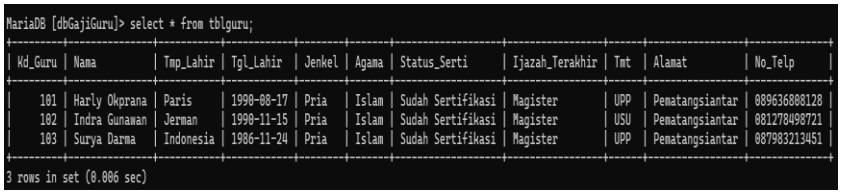
Soal :	Soal DML – 01.E
Syntax :	insert into tblgaji VALUES -> (501,'101','1','38','30000','P3K', 1500000,120000,2520000), -> (502,'102','2','42','30000','PNS', 1750000,100000,2910000), -> (503,'103','3','36','30000','ASN', 1600000,110000,2790000);
Maksud :	Menambahkan beberapa record ke dalam tabel tblgaji
Hasil	

Soal :	Soal DML – 01.F
Syntax :	insert into tbllogin -> SET Username = 'sicucukagura14', -> Password = 'cucukagura14';
Maksud :	Menambahkan beberapa record ke dalam tabel tbllogin menggunakan Insert SET setiap kolom
Hasil	

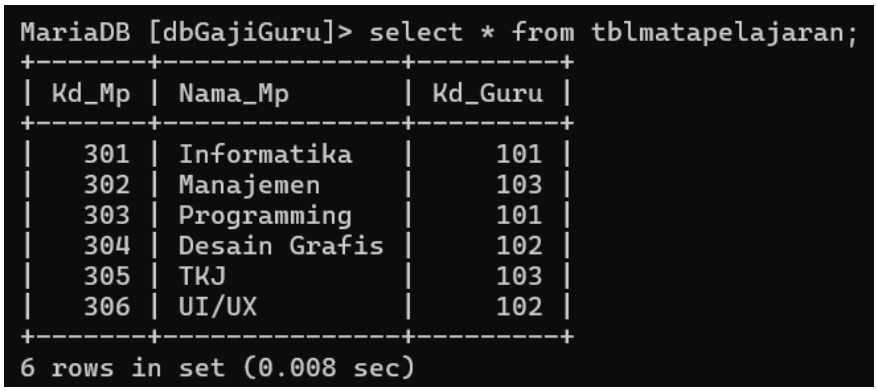
Soal :	Soal DML – 02
Syntax :	Select * from tbllogin;
Maksud :	Menampilkan Semua record pada tabel tbllogin

Hasil	

```
MariaDB [dbGajiGuru]> select * from tbllogin;
+----------------+---------------+
| Username       | Password      |
+----------------+---------------+
| arifhrvn150    | arifsilaban15 |
| KangMelalak    | healing123    |
| sicucukagura14 | cucukagura14  |
+----------------+---------------+
3 rows in set (0.113 sec)
```

Soal :	Soal DML – 02.A
Syntax :	Select * from tblguru;
Maksud :	Menampilkan Semua record pada tabel tblguru
Hasil	

```
MariaDB [dbGajiGuru]> select * from tblguru;
+--------+--------------+-----------+------------+-------+-------+-----------------+----------------+-----+----------------+-------------+
| Kd_Guru | Nama         | Tmp_Lahir | Tgl_Lahir  | Jenkel | Agama | Status_Serti    | Ijazah_Terakhir | Tmt | Alamat         | No_Telp     |
+--------+--------------+-----------+------------+-------+-------+-----------------+----------------+-----+----------------+-------------+
|    101 | Harly Okprana | Paris     | 1990-08-17 | Pria   | Islam | Sudah Sertifikasi | Magister       | UPP | Pematangsiantar | 089636808128 |
|    102 | Indra Gunawan | Jerman    | 1990-11-15 | Pria   | Islam | Sudah Sertifikasi | Magister       | USU | Pematangsiantar | 081278498721 |
|    103 | Surya Darma   | Indonesia | 1986-11-24 | Pria   | Islam | Sudah Sertifikasi | Magister       | UPP | Pematangsiantar | 087983213451 |
+--------+--------------+-----------+------------+-------+-------+-----------------+----------------+-----+----------------+-------------+
3 rows in set (0.006 sec)
```

Soal :	Soal DML – 02.B
Syntax :	Select * from tblmatapelajaran;
Maksud :	Menampilkan Semua record pada tabel tblmatapelajaran
Hasil	

```
MariaDB [dbGajiGuru]> select * from tblmatapelajaran;
+--------+---------------+----------+
| Kd_Mp  | Nama_Mp       | Kd_Guru  |
+--------+---------------+----------+
|    301 | Informatika   |      101 |
|    302 | Manajemen     |      103 |
|    303 | Programming   |      101 |
|    304 | Desain Grafis |      102 |
|    305 | TKJ           |      103 |
|    306 | UI/UX         |      102 |
+--------+---------------+----------+
6 rows in set (0.008 sec)
```

Soal :	Soal DML – 02.C
Syntax :	Select * from tblkelas;
Maksud :	Menampilkan Semua record pada tabel tblkelas
Hasil	

```
MariaDB [dbGajiGuru]> select * from tblkelas;
+----------+---------------+
| Kd_Kelas | Nm_Kelas      |
+----------+---------------+
|     2401 | 24S04 SIKOPAT |
|     2402 | 24S05 SIKOMA  |
|     2403 | 24S06 SIKONAM |
+----------+---------------+
3 rows in set (0.009 sec)
```

Soal :	Soal DML – 02.D
Syntax :	Select * from tblabsenajar;
Maksud :	Menampilkan Semua record pada tabel tblabsenajar
Hasil	

```
MariaDB [dbGajiGuru]> select * from tblabsenajar;
+----------+---------+------------+-------+----------+----------+-----------+-------+
| Kd_Absen | Kd_Guru | Tgl_Lahir  | Kd_Mp | JlhJmPlj | Kd_Kelas | Ket_Hadir | Total |
+----------+---------+------------+-------+----------+----------+-----------+-------+
|        1 |     101 | 1990-08-17 |   301 |        2 |     2401 | Hadir     | 25    |
|        2 |     102 | 1990-11-15 |   304 |        3 |     2403 | Hadir     | 32    |
|        3 |     103 | 1986-11-24 |   305 |        2 |     2402 | Hadir     | 28    |
+----------+---------+------------+-------+----------+----------+-----------+-------+
3 rows in set (0.009 sec)
```

Soal :	Soal DML – 02.E
Syntax :	Select * from tblgaji;
Maksud :	Menampilkan Semua record pada tabel tblgaji
Hasil	

```
MariaDB [dbGajiGuru]> select * from tblgaji;
+---------+---------+----------+-------------+----------+-----------------+------------+-------+----------+
| Kd_Gaji | Kd_Guru | Kd_Absen | TotJlhJamPlj | GajiPrJam | KetHonorTambahan | JlhTambahan | Pot   | Tot_Gaji |
+---------+---------+----------+-------------+----------+-----------------+------------+-------+----------+
|     501 |     101 |        1 |          38 |    30000 | P3K             |    1500000 | 120000 |  2520000 |
|     502 |     102 |        2 |          42 |    30000 | PNS             |    1750000 | 100000 |  2910000 |
|     503 |     103 |        3 |          36 |    30000 | ASN             |    1600000 | 110000 |  2790000 |
+---------+---------+----------+-------------+----------+-----------------+------------+-------+----------+
3 rows in set (0.009 sec)
```

Soal :	Soal DML – 03
Syntax :	Update tblabsenajar Set Tgl_Lahir = '1990-11-24' Where Kd_Absen = '3';
Maksud :	Mengubah Tanggal Lahir Menjadi 1990-11-24 pada record Kd_Absen = 3 (Sebelumnya Tgl Lahirnya adalah 1986-11-24)
Hasil	

```
MariaDB [dbGajiGuru]> Update tblabsenajar Set Tgl_Lahir = '1990-11-24' Where Kd_Absen = '3';
Query OK, 1 row affected (0.019 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Setelah Diubah:

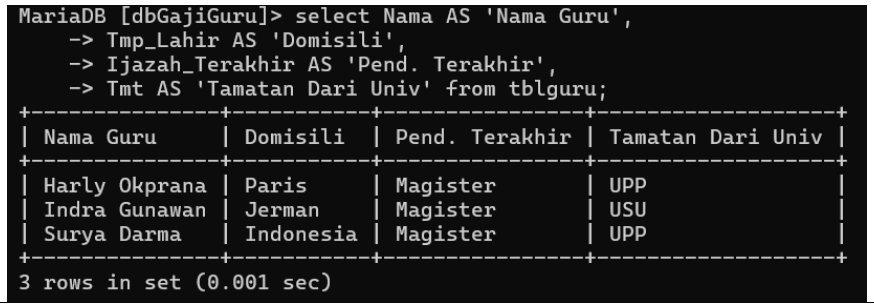
```
MariaDB [dbGajiGuru]> select * from tblabsenajar;
+----------+---------+------------+-------+----------+----------+-----------+-------+
| Kd_Absen | Kd_Guru | Tgl_Lahir  | Kd_Mp | JlhJmPlj | Kd_Kelas | Ket_Hadir | Total |
+----------+---------+------------+-------+----------+----------+-----------+-------+
|        1 |     101 | 1990-08-17 |   301 |        2 |     2401 | Hadir     | 25    |
|        2 |     102 | 1990-11-15 |   304 |        3 |     2403 | Hadir     | 32    |
|        3 |     103 | 1990-11-24 |   305 |        2 |     2402 | Hadir     | 28    |
+----------+---------+------------+-------+----------+----------+-----------+-------+
3 rows in set (0.001 sec)
```

Soal :	Soal DML – 04
Syntax :	delete from tblmatapelajaran -> WHERE Kd_Mp = '303';
Maksud :	Menghapus 1 record pada tblmatapelajaran yang memiliki Kd_Mp = '303'
Hasil	MariaDB [dbGajiGuru]> delete from tblmatapelajaran -> WHERE Kd_Mp = '303'; Query OK, 1 row affected (0.013 sec) Setelah Dihapus: MariaDB [dbGajiGuru]> select * from tblmatapelajaran; +-------+---------------+---------+ | Kd_Mp | Nama_Mp | Kd_Guru | +-------+---------------+---------+ | 301 | Informatika | 101 | | 302 | Manajemen | 103 | | 304 | Desain Grafis | 102 | | 305 | TKJ | 103 | | 306 | UI/UX | 102 | +-------+---------------+---------+ 5 rows in set (0.001 sec)

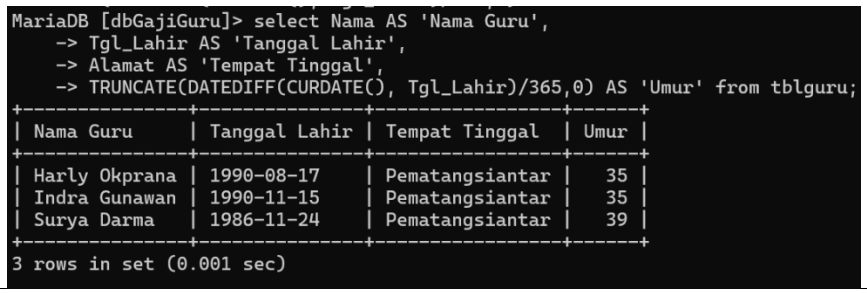
2.6 DQL (Data Query Language)

DQL (Data Query Language) adalah bagian dari SQL yang berfungsi untuk mengambil, menampilkan, atau melakukan pencarian data dari sebuah tabel dalam database. DQL tidak digunakan untuk mengubah struktur tabel maupun menambah data, tetapi hanya untuk melakukan query atau permintaan data (Adflin, 2020). Perintah utama dalam DQL adalah SELECT, yang memungkinkan pengguna menampilkan seluruh data, memilih kolom tertentu, melakukan penyaringan data dengan WHERE, mengurutkan data menggunakan ORDER BY, serta melakukan pengelompokan data dengan GROUP BY dan menyaring hasil grup menggunakan HAVING (Canggih Ajika Pamungkas, 2017).

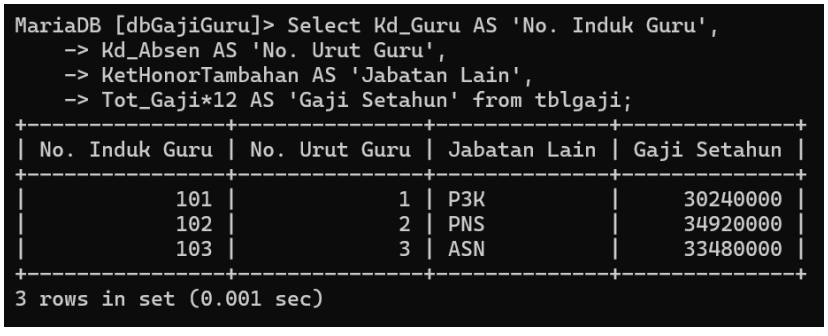
Soal :	Soal DQL – 01
Syntax :	select Nama AS 'Nama Guru', -> Tmp_Lahir AS 'Domisili', -> Ijazah_Terakhir AS 'Pend. Terakhir', -> Tmt AS 'Tamatan Dari Univ' from tblguru;

Maksud :	Menampilkan Record dengan Nama Alias pada nama Field yang ditampilkan menggunakan Fungsi AS STRING
Hasil	

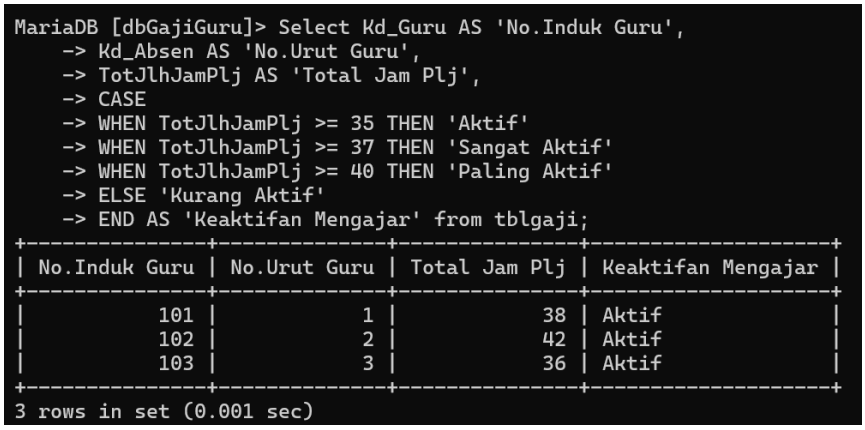
```
MariaDB [dbGajiGuru]> select Nama AS 'Nama Guru',
    -> Tmp_Lahir AS 'Domisili',
    -> Ijazah_Terakhir AS 'Pend. Terakhir',
    -> Tmt AS 'Tamatan Dari Univ' from tblguru;
+--------------+-----------+----------------+-------------------+
| Nama Guru    | Domisili  | Pend. Terakhir | Tamatan Dari Univ |
+--------------+-----------+----------------+-------------------+
| Harly Okprana | Paris     | Magister       | UPP               |
| Indra Gunawan | Jerman    | Magister       | USU               |
| Surya Darma   | Indonesia | Magister       | UPP               |
+--------------+-----------+----------------+-------------------+
3 rows in set (0.001 sec)
```

Soal :	Soal DQL – 02
Syntax :	select Nama AS 'Nama Guru', -> Tgl_Lahir AS 'Tanggal Lahir', -> Alamat AS 'Tempat Tinggal', -> TRUNCATE(DATEDIFF(CURDATE(), Tgl_Lahir)/365,0) AS 'Umur' from tblguru;
Maksud :	Menampilkan Record dengan Nama Alias pada nama Field yang ditampilkan menggunakan Fungsi AS STRING dan Kalkulasi Umur Berdasarkan Tanggal Lahir.
Hasil	

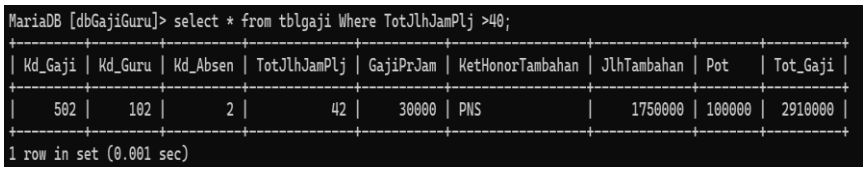
```
MariaDB [dbGajiGuru]> select Nama AS 'Nama Guru',
    -> Tgl_Lahir AS 'Tanggal Lahir',
    -> Alamat AS 'Tempat Tinggal',
    -> TRUNCATE(DATEDIFF(CURDATE(), Tgl_Lahir)/365,0) AS 'Umur' from tblguru;
+--------------+--------------+-----------------+------+
| Nama Guru    | Tanggal Lahir | Tempat Tinggal  | Umur |
+--------------+--------------+-----------------+------+
| Harly Okprana | 1990-08-17   | Pematangsiantar |   35 |
| Indra Gunawan | 1990-11-15   | Pematangsiantar |   35 |
| Surya Darma   | 1986-11-24   | Pematangsiantar |   39 |
+--------------+--------------+-----------------+------+
3 rows in set (0.001 sec)
```

Soal :	Soal DQL – 03
Syntax :	Select Kd_Guru AS 'No. Induk Guru', -> Kd_Absen AS 'No. Urut Guru', -> KetHonorTambahan AS 'Jabatan Lain', -> Tot_Gaji 12 AS 'Gaji Setahun' from tblgaji;
Maksud :	Menampilkan Record dengan Nama Alias pada nama Field yang ditampilkan menggunakan Fungsi AS STRING dan Kalkulasi Total Gaji Guru Selama Setahun (12 Bulan).
Hasil	

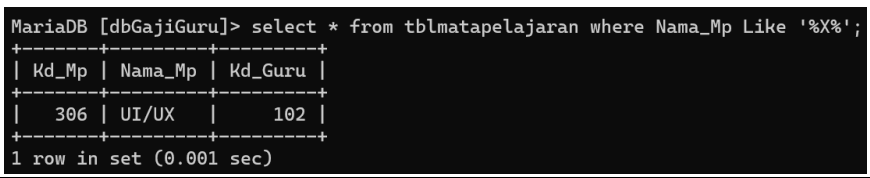
```
MariaDB [dbGajiGuru]> Select Kd_Guru AS 'No. Induk Guru',
    -> Kd_Absen AS 'No. Urut Guru',
    -> KetHonorTambahan AS 'Jabatan Lain',
    -> Tot_Gaji*12 AS 'Gaji Setahun' from tblgaji;
+---------------+--------------+--------------+--------------+
| No. Induk Guru | No. Urut Guru | Jabatan Lain | Gaji Setahun |
+---------------+--------------+--------------+--------------+
|           101 |            1 | P3K          |     30240000 |
|           102 |            2 | PNS          |     34920000 |
|           103 |            3 | ASN          |     33480000 |
+---------------+--------------+--------------+--------------+
3 rows in set (0.001 sec)
```

Soal :	Soal DQL – 04
Syntax :	Select Kd_Guru AS 'No. Induk Guru', -> Kd_Absen AS 'No. Urut Guru', -> KetHonorTambahan AS 'Jabatan Lain', -> Tot_Gaji 12 AS 'Gaji Setahun' from tblgaji;
Maksud :	Menampilkan Record dengan Nama Alias pada nama Field yang ditampilkan menggunakan Fungsi AS STRING dan Fungsi CASE sebagai Logika Keaktifan Guru berdasarkan Total Jumlah Mata Pelajaran.
Hasil	

```
MariaDB [dbGajiGuru]> Select Kd_Guru AS 'No.Induk Guru',
    -> Kd_Absen AS 'No.Urut Guru',
    -> TotJlhJamPlj AS 'Total Jam Plj',
    -> CASE
    -> WHEN TotJlhJamPlj >= 35 THEN 'Aktif'
    -> WHEN TotJlhJamPlj >= 37 THEN 'Sangat Aktif'
    -> WHEN TotJlhJamPlj >= 40 THEN 'Paling Aktif'
    -> ELSE 'Kurang Aktif'
    -> END AS 'Keaktifan Mengajar' from tblgaji;
+---------------+--------------+---------------+--------------------+
| No.Induk Guru | No.Urut Guru | Total Jam Plj | Keaktifan Mengajar |
+---------------+--------------+---------------+--------------------+
|           101 |            1 |            38 | Aktif              |
|           102 |            2 |            42 | Aktif              |
|           103 |            3 |            36 | Aktif              |
+---------------+--------------+---------------+--------------------+
3 rows in set (0.001 sec)
```

Soal :	Soal DQL – 05
Syntax :	select * from tblgaji Where TotJlhJamPlj >40;
Maksud :	Menampilkan Record yang Total Jam Pelajarannya Lebih dari 40
Hasil	

```
MariaDB [dbGajiGuru]> select * from tblgaji Where TotJlhJamPlj >40;
+---------+---------+----------+-------------+-----------+-----------------+------------+--------+----------+
| Kd_Gaji | Kd_Guru | Kd_Absen | TotJlhJamPlj | GajiPrJam | KetHonorTambahan | JlhTambahan | Pot    | Tot_Gaji |
+---------+---------+----------+-------------+-----------+-----------------+------------+--------+----------+
|     502 |     102 |        2 |          42 |     30000 | PNS              |    1750000 | 100000 |  2910000 |
+---------+---------+----------+-------------+-----------+-----------------+------------+--------+----------+
1 row in set (0.001 sec)
```

Soal :	Soal DQL – 06
Syntax :	select * from tblmatapelajaran where Nama_Mp Like '%X%';
Maksud :	Menampilkan Record pada table tblmatapelajaran yang dimana record tersebut ada mengandung huruf X di datanya
Hasil	

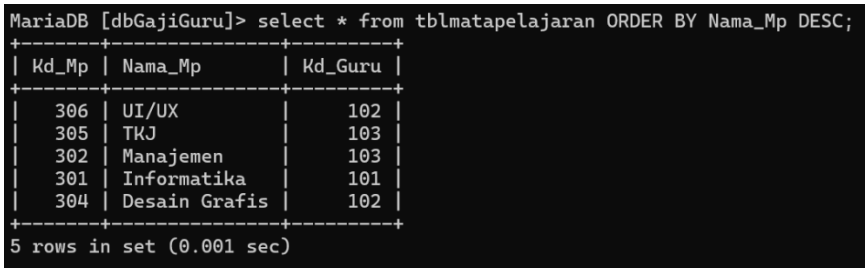
```
MariaDB [dbGajiGuru]> select * from tblmatapelajaran where Nama_Mp Like '%X%';
+--------+---------+----------+
| Kd_Mp  | Nama_Mp | Kd_Guru  |
+--------+---------+----------+
|   306  | UI/UX   |      102 |
+--------+---------+----------+
1 row in set (0.001 sec)
```


Soal :	Soal DQL – 7
Syntax :	select * from tblmatapelajaran where Nama_Mp Like 'I%';
Maksud :	Menampilkan Record pada table tblmatapelajaran yang dimana record tersebut memiliki huruf awalan “I” di datanya
Hasil	

```
MariaDB [dbGajiGuru]> select * from tblmatapelajaran where Nama_Mp Like 'I%';
+-------+-------------+---------+
| Kd_Mp | Nama_Mp     | Kd_Guru |
+-------+-------------+---------+
|   301 | Informatika |     101 |
+-------+-------------+---------+
1 row in set (0.001 sec)
```

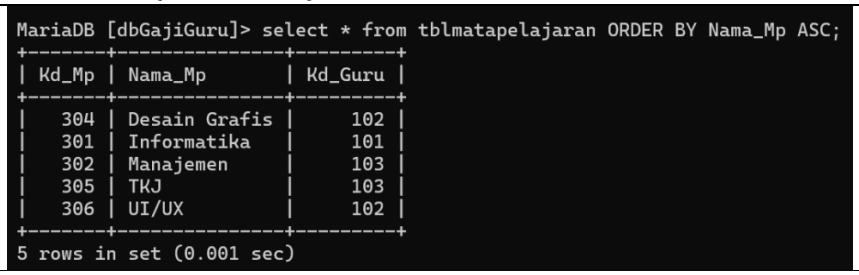
Soal :	Soal DQL – 8
Syntax :	select * from tblmatapelajaran where Nama_Mp Like 'I%';
Maksud :	Menampilkan Record pada table tblmatapelajaran yang dimana record tersebut memiliki huruf terakhirnya adalah “J” di datanya
Hasil	

```
MariaDB [dbGajiGuru]> select * from tblmatapelajaran where Nama_Mp Like '%J';
+-------+---------+---------+
| Kd_Mp | Nama_Mp | Kd_Guru |
+-------+---------+---------+
|   305 | TKJ     |     103 |
+-------+---------+---------+
1 row in set (0.001 sec)
```

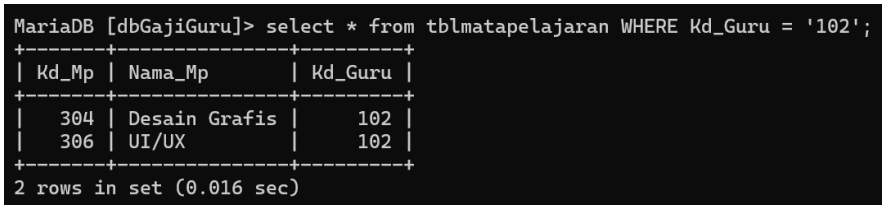
Soal :	Soal DQL – 9
Syntax :	Select * from tblmatapelajaran ORDER BY Nama_Mp DESC;
Maksud :	Menampilkan Semua Record yang ada pada tabel tblmatapelajaran dan diurutkan berdasarkan Huruf Alfabet Tertinggi (DESCENDING)
Hasil	

```
MariaDB [dbGajiGuru]> select * from tblmatapelajaran ORDER BY Nama_Mp DESC;
+-------+---------------+---------+
| Kd_Mp | Nama_Mp       | Kd_Guru |
+-------+---------------+---------+
|   306 | UI/UX         |     102 |
|   305 | TKJ           |     103 |
|   302 | Manajemen     |     103 |
|   301 | Informatika   |     101 |
|   304 | Desain Grafis |     102 |
+-------+---------------+---------+
5 rows in set (0.001 sec)
```

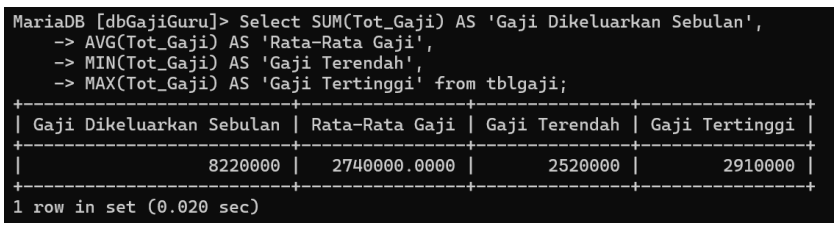
Soal :	Soal DQL – 10

Syntax :	select * from tblmatapelajaran ORDER BY Nama_Mp ASC;
Maksud :	Menampilkan Semua Record yang ada pada tabel tblmatapelajaran dan diurutkan berdasarkan Huruf Alfabet Terendah (ASCENDING)
Hasil	

```
MariaDB [dbGajiGuru]> select * from tblmatapelajaran ORDER BY Nama_Mp ASC;
+-------+---------------+---------+
| Kd_Mp | Nama_Mp       | Kd_Guru |
+-------+---------------+---------+
|   304 | Desain Grafis |     102 |
|   301 | Informatika   |     101 |
|   302 | Manajemen     |     103 |
|   305 | TKJ           |     103 |
|   306 | UI/UX         |     102 |
+-------+---------------+---------+
5 rows in set (0.001 sec)
```

Soal :	Soal DQL – 11
Syntax :	select * from tblmatapelajaran WHERE Kd_Guru = '102';
Maksud :	Menampilkan Record pada tabel tblmatapelajaran yang Memiliki Kd_Guru = 102 (Guru Tersebut Mengampu Mata Pelajaran apa saja)
Hasil	

```
MariaDB [dbGajiGuru]> select * from tblmatapelajaran WHERE Kd_Guru = '102';
+-------+---------------+---------+
| Kd_Mp | Nama_Mp       | Kd_Guru |
+-------+---------------+---------+
|   304 | Desain Grafis |     102 |
|   306 | UI/UX         |     102 |
+-------+---------------+---------+
2 rows in set (0.016 sec)
```

Soal :	Soal DQL – 12
Syntax :	Select SUM(Tot_Gaji) AS 'Gaji Dikeluarkan Sebulan', -> AVG(Tot_Gaji) AS 'Rata-Rata Gaji', -> MIN(Tot_Gaji) AS 'Gaji Terendah', -> MAX(Tot_Gaji) AS 'Gaji Tertinggi' from tblgaji;
Maksud :	Menampilkan Jumlah Total Gaji Sebagai Gaji Dikeluarkan Sebulan, Menampilkan Rata-Rata Gaji, Menampilkan Gaji Terendah, dan Menampilkan Gaji Tertinggi.
Hasil	

```
MariaDB [dbGajiGuru]> Select SUM(Tot_Gaji) AS 'Gaji Dikeluarkan Sebulan',
    -> AVG(Tot_Gaji) AS 'Rata-Rata Gaji',
    -> MIN(Tot_Gaji) AS 'Gaji Terendah',
    -> MAX(Tot_Gaji) AS 'Gaji Tertinggi' from tblgaji;
+-------------------------+---------------+---------------+---------------+
| Gaji Dikeluarkan Sebulan | Rata-Rata Gaji | Gaji Terendah | Gaji Tertinggi |
+-------------------------+---------------+---------------+---------------+
|                 8220000 |   2740000.0000 |       2520000 |        2910000 |
+-------------------------+---------------+---------------+---------------+
1 row in set (0.020 sec)
```

Soal :	Soal DQL – 13
Syntax :	select * from tblgaji WHERE Tot_Gaji > (Select AVG(Tot_Gaji) from tblgaji);
Maksud :	Menampilkan Total Gaji Guru yang Lebih Besar dari Rata-Rata Total Gaji Guru
Hasil	

```
MariaDB [dbGajiGuru]> select * from tblgaji WHERE Tot_Gaji > (Select AVG(Tot_Gaji) from tblgaji);
+---------+---------+----------+-------------+----------+-----------------+-------------+--------+----------+
| Kd_Gaji | Kd_Guru | Kd_Absen | TotJlhJamPlj | GajiPrJam | KetHonorTambahan | JlhTambahan | Pot    | Tot_Gaji |
+---------+---------+----------+-------------+----------+-----------------+-------------+--------+----------+
|     502 |     102 |        2 |          42 |    30000 | PNS             |     1750000 | 100000 |  2910000 |
|     503 |     103 |        3 |          36 |    30000 | ASN             |     1600000 | 110000 |  2790000 |
+---------+---------+----------+-------------+----------+-----------------+-------------+--------+----------+
2 rows in set (0.008 sec)
```

Soal :	Soal DQL – 14
Syntax :	select * from tblgaji WHERE Tot_Gaji < (Select AVG(Tot_Gaji) from tblgaji);
Maksud :	Menampilkan Total Gaji Guru yang Lebih Kecil dari Rata-Rata Total Gaji Guru
Hasil	

```
MariaDB [dbGajiGuru]> select * from tblgaji WHERE Tot_Gaji < (Select AVG(Tot_Gaji) from tblgaji);
+---------+---------+----------+-------------+----------+-----------------+-------------+--------+----------+
| Kd_Gaji | Kd_Guru | Kd_Absen | TotJlhJamPlj | GajiPrJam | KetHonorTambahan | JlhTambahan | Pot    | Tot_Gaji |
+---------+---------+----------+-------------+----------+-----------------+-------------+--------+----------+
|     501 |     101 |        1 |          38 |    30000 | P3K             |     1500000 | 120000 |  2520000 |
+---------+---------+----------+-------------+----------+-----------------+-------------+--------+----------+
1 row in set (0.001 sec)
```

Latihan Soal:

Setelah kamu praktikkan soal-soal praktikum di atas, boleh dong dilatih skill nya melalui soal soal latihan ini:

NB: Lanjutkan dari Progress kamu pada Database dbBengkelUmum

1. Insert beberapa data pada tabel tblpelanggan minimal 3 record.
2. Insert beberapa data pada tabel tblmekanik minimal 3 record.
3. Insert beberapa data pada tabel tblsukucadang minimal 4 record.
4. Insert beberapa data pada tabel tblservis minimal 3 record.
5. Insert beberapa data pada tabel tbldetailservis minimal 3 record.
6. Update nomor telepon pada tabel tblpelanggan dengan Kd_Pelanggan = 1 menjadi 081200000000.
7. Delete satu record dari tabel tblsukucadang dengan Kd_Sparepart = 'SP003'.
8. Tampilkan semua data pada tabel tblpelanggan.

9. Tampilkan data dari tabel tblmekanik hanya kolom: Nama_Mekanik, Keahlian, dan No_Telp menggunakan AS sebagai alias nama kolom.

10. Tampilkan data dari tabel tblsukucadang yang mengandung huruf “A” pada Nama_Sparepart.

11. Tampilkan data dari tabel tblsukucadang yang awalan huruf “B” pada Nama_Sparepart.

12. Tampilkan data dari tabel tblsukucadang yang huruf terakhirnya adalah “G” pada Nama_Sparepart.

13. Tampilkan semua data dari tabel tblsukucadang urutkan berdasarkan harga tertinggi (DESC).

14. Tampilkan semua data dari tabel tblsukucadang urutkan berdasarkan nama sparepart dari A sampai Z (ASC).

15. Tampilkan semua data servis pada tabel tblservis yang dilakukan pada tanggal 2025-01-14.

16. Tampilkan semua data pada tabel tbldetailservis yang memiliki Total_Harga lebih dari 50.000.

17. Hitung jumlah total harga seluruh servis menggunakan fungsi SUM.

18. Tampilkan jumlah total, rata-rata, harga tertinggi, dan harga terendah dari Total_Harga menggunakan SUM, AVG, MAX, MIN.

19. Tampilkan data pada tabel tbldetailservis yang memiliki Total_Harga lebih besar dari rata-rata Total_Harga.

20. Tampilkan data pada tabel tbldetailservis yang memiliki Total_Harga lebih kecil dari rata-rata Total_Harga.

BAB III

EXPORT & IMPORT

3.1 Export

Export adalah proses untuk mengambil atau menyalin database dari sistem dan menyimpannya ke dalam bentuk file tertentu. File hasil export biasanya digunakan sebagai cadangan (backup) atau untuk dipindahkan ke komputer atau server lain. Format yang paling umum pada MySQL adalah .sql, karena file ini berisi perintah-perintah untuk membangun kembali struktur tabel, data, index, dan pengaturan database.

Export dilakukan ketika:

- Ingin membuat backup database
- Ingin memindahkan database ke komputer lain
- Ingin menyimpan salinan database sebelum dilakukan modifikasi

Dengan export, seluruh isi database dapat disimpan dalam satu file sehingga apabila terjadi kerusakan atau kehilangan data, database dapat dikembalikan dengan mudah.

Langkah-langkah export database menggunakan phpMyAdmin:

1. Masuk ke phpMyAdmin melalui browser
2. Pilih database yang akan di-export
3. Klik menu **Export**
4. Pilih metode **Quick**
5. Pilih format **SQL**
6. Klik **Go**
7. File akan terunduh sebagai nama_database.sql

Hasil export berupa file .sql yang dapat disimpan, dipindahkan, atau dimasukkan kembali ke database lain.

3.2 Import

Import adalah proses memasukkan database ke dalam sistem dari file hasil export. Proses ini digunakan untuk mengembalikan data hasil backup, mengkloning database, atau memindahkan database dari lingkungan lain. Import biasanya dilakukan menggunakan file .sql yang telah diexport sebelumnya.

Import dilakukan ketika:

- Ingin mengembalikan database yang rusak
- Ingin mengambil data hasil backup ke sistem
- Ingin menggunakan database dari komputer lain

Dengan fitur import, sistem database dapat dikembalikan dalam kondisi yang sama seperti sebelum terjadi kerusakan atau kesalahan.

Langkah-langkah import database:

1. Masuk ke phpMyAdmin
2. Pilih menu **Import**
3. Klik **Choose File**
4. Pilih file hasil export (format .sql)
5. Klik **Go**
6. Sistem akan memproses perintah SQL dari file tersebut

Jika file cocok, database akan dibuat atau diperbarui sesuai isi file.

3.3 Konversi file Excel ke MySQL

Konversi file Excel ke MySQL adalah proses mengubah data yang disimpan dalam format spreadsheet, seperti *.xls* atau *.xlsx*, menjadi format yang dapat digunakan dan disimpan pada sistem basis data MySQL. Proses konversi ini biasanya dilakukan ketika data awal dikumpulkan atau dicatat dalam Microsoft Excel, kemudian data tersebut ingin diolah lebih lanjut melalui query SQL, disimpan dalam tabel database, atau digunakan untuk aplikasi yang membutuhkan manajemen data yang lebih terstruktur. Melalui konversi ini, setiap baris dan kolom pada file Excel akan disesuaikan dengan struktur tabel MySQL sehingga data dapat disimpan dengan benar dan tidak terjadi kesalahan format.

Konversi file Excel menjadi database MySQL memberikan banyak manfaat, terutama dalam pengolahan data dalam jumlah besar. Jika data hanya disimpan dalam Excel, kemampuan pencarian, penyaringan, dan analisis data menjadi terbatas. Namun setelah dikonversi ke MySQL, data dapat dikelola dengan lebih efisien menggunakan perintah SQL, seperti SELECT, UPDATE, JOIN, dan berbagai fungsi agregat. Proses konversi juga membantu menghindari duplikasi data dan kesalahan pencatatan, karena database memiliki fitur primary key dan relasi antar tabel yang memastikan integritas data. Proses konversi biasanya dilakukan melalui fitur import di phpMyAdmin, menggunakan format *.csv* atau *.txt* sehingga data dari Excel dapat terbaca oleh sistem MySQL.

Adapun cara untuk mengkonversi file excel (.csv atau .txt) ke dalam mySQL:

1. Buka File Excel dan Ketik Data seperti berikut

A. File Penjualan sebagai .csv

A	B	C	D	E	F
IdProduk	NamaProduk	JenisProduk	Harga	Tersedia	Garansi
1201	Osprey Backpack 65L	Carrier	3.400.000	8	2 Tahun
1202	NatureHike Tracking Pole	Penopang	175.000	32	4 Bulan
1203	Palava Flysheet	Anti UV	327.000	15	7 Bulan
1204	Speeds Nesting	Cooking	280.000	50	4 Bulan
1205	Lavio Footwear	Hiking Boots	400.000	20	6 Bulan

B. File Pembelian sebagai .txt

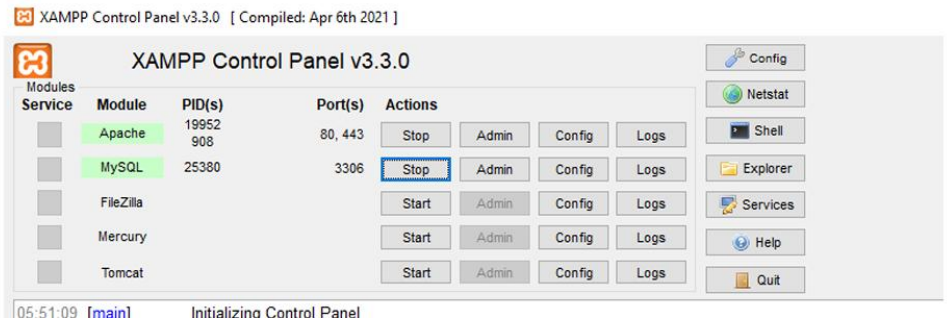
A	B	C	D	E	F	G	H
IdProduk	NamaProduk	NamaPembeli	Alamat	HargaTotal	Qty	Pembayaran	TglBeli
1202	NatureHike Tracking Pole	Arif Silaban	Siantar	700.000	4	CASH	14/10/2025 18:37
1205	Lavio Footwear	Jhondes Sitorus	Timuran	4.000.000	10	QRIS BRI	20/10/2025 13:11
1204	Speeds Nesting	Rafael Sinaga	Tangerang	560.000	2	QRIS LIVIN	23/10/2025 15:27
1202	NatureHike Tracking Pole	Diaz Al Latif	Bah Jambi	525.000	3	CASH	28/10/2025 09:55

2. Lakukan Save As dengan type CSV (Comma Delimited) atau Txt

Pada Laporan nama file nya adalah Penjualan.csv dan Pembelian.txt

Pembelian	30/10/2025 09:42	Text Document	1 KB
Penjualan	30/10/2025 08:46	Microsoft Excel Comma Separated Values File	1 KB
PenjualanGunung	04/11/2025 12:50	Text Document	1 KB
Siswa	09/10/2025 11:23	Microsoft Excel Comma Separated Values File	1 KB
PembelianGunung	04/11/2025 13:05	Text Document	1 KB

3. **Nyalakan Xampp (Apache dan MySQL)**



4. **Membuat Database untuk data yang akan dimasukkan dan membuat tabel sesuai data dari excel yang diinput supaya tidak terjadi kesalahan data**

```
MariaDB [mhssi]> use JualBeliGunung
Database changed
MariaDB [JualBeliGunung]> create table Penjualan(
    -> IdProduk Varchar(10) PRIMARY KEY,
    -> NamaProduk Varchar(50),
    -> JenisProduk Varchar(25),
    -> Harga Varchar(15),
    -> Tersedia INT,
    -> Garansi Varchar(10));
Query OK, 0 rows affected (0.025 sec)
```

```
MariaDB [JualBeliGunung]> Create table Pembelian(
    -> IdProduk Varchar(10),
    -> NamaProduk Varchar(50),
    -> NamaPembeli Varchar(30),
    -> Alamat Varchar(20),
    -> HargaTotal Varchar(20),
    -> Qty INT,
    -> Pembayaran Varchar(15),
    -> TglBeli Varchar(25));
Query OK, 0 rows affected (0.030 sec)
```

5. **Buat perintah mysql dos untuk memasukkan data dari excell csv ke tabel:**

A. Penjualan.csv → Tabel Penjualan

```
MariaDB [JualBeliGunung]> LOAD DATA INFILE 'C:\\DataExcel\\Penjualan.csv'
    -> Into Table Penjualan
    -> FIELDS TERMINATED BY ';'
    -> LINES TERMINATED BY '\r\n'
    -> IGNORE 1 LINES;
Query OK, 5 rows affected (0.011 sec)
Records: 5  Deleted: 0  Skipped: 0  Warnings: 0

MariaDB [JualBeliGunung]> select * from Penjualan;
+----------+--------------------------+--------------+-----------+----------+----------+
| IdProduk | NamaProduk               | JenisProduk  | Harga     | Tersedia | Garansi  |
+----------+--------------------------+--------------+-----------+----------+----------+
| 1201     | Osprey Backpack 65L      | Carrier      | 3.400.000 |        8 | 2 Tahun  |
| 1202     | NatureHike Tracking Pole | Penopang     | 175.000   |       32 | 4 Bulan  |
| 1203     | Palava Flysheet          | Anti UV      | 327.000   |       15 | 7 Bulan  |
| 1204     | Speeds Nesting           | Cooking      | 280.000   |       50 | 4 Bulan  |
| 1205     | Lavio Footwear           | Hiking Boots | 400.000   |       20 | 6 Bulan  |
+----------+--------------------------+--------------+-----------+----------+----------+
5 rows in set (0.001 sec)
```

B. Penjualan.txt → Tabel Penjualan

```
MariaDB [JualBeliGunung]> LOAD DATA INFILE 'C:\\DataExcel\\Pembelian.txt'
    -> Into Table Pembelian
    -> FIELDS TERMINATED BY '\t'
    -> LINES TERMINATED BY '\r\n'
    -> IGNORE 1 LINES;
Query OK, 4 rows affected (0.013 sec)
Records: 4  Deleted: 0  Skipped: 0  Warnings: 0

MariaDB [JualBeliGunung]> select * from Pembelian;
+----------+--------------------------+---------------+-----------+------------+------+------------+---------------------+
| IdProduk | NamaProduk               | NamaPembeli   | Alamat    | HargaTotal | Qty  | Pembayaran | TglBeli             |
+----------+--------------------------+---------------+-----------+------------+------+------------+---------------------+
| 1202     | NatureHike Tracking Pole | Arif Silaban  | Siantar   | 700.000    |    4 | CASH       | 14/10/2025 18:37    |
| 1205     | Lavio Footwear           | Jhondes Sitorus | Timuran | 4.000.000  |   10 | QRIS BRI   | 20/10/2025 13:11    |
| 1204     | Speeds Nesting           | Rafael Sinaga | Tangerang | 560.000    |    2 | QRIS LIVIN | 23/10/2025 15:27    |
| 1202     | NatureHike Tracking Pole | Diaz Al Latif | Bah Jambi | 525.000    |    3 | CASH       | 28/10/2025 09:55    |
+----------+--------------------------+---------------+-----------+------------+------+------------+---------------------+
4 rows in set (0.001 sec)
```

Kejadian saat konversi file txt ke SQL menjadi Null:

```
MariaDB [JualBeliGunung]> LOAD DATA INFILE 'C:\\DataExcel\\Pembelian.txt'
    -> Into Table Pembelian
    -> FIELDS TERMINATED BY ';'
    -> LINES TERMINATED BY '\r\n'
    -> IGNORE 1 LINES;
Query OK, 4 rows affected, 32 warnings (0.036 sec)
Records: 4  Deleted: 0  Skipped: 0  Warnings: 32

MariaDB [JualBeliGunung]> select * from Pembelian
    -> ;
+------------+------------+-------------+--------+------------+------+------------+----------+
| IdProduk   | NamaProduk | NamaPembeli | Alamat | HargaTotal | Qty  | Pembayaran | TglBeli  |
+------------+------------+-------------+--------+------------+------+------------+----------+
| 1202  Natur | NULL       | NULL        | NULL   | NULL       | NULL | NULL       | NULL     |
| 1205  Lavio | NULL       | NULL        | NULL   | NULL       | NULL | NULL       | NULL     |
| 1204  Speed | NULL       | NULL        | NULL   | NULL       | NULL | NULL       | NULL     |
| 1202  Natur | NULL       | NULL        | NULL   | NULL       | NULL | NULL       | NULL     |
+------------+------------+-------------+--------+------------+------+------------+----------+
4 rows in set (0.001 sec)
```

Jangan sampai salah dalam memasukkan FIELDS TERMINTED BY.

Pada .csv menggunakan FIELDS TERMINATED BY ‘;’,

Pada .txt menggunakan FIELDS TERMINATED BT ‘;’.

Penjelasan Command mengenai Konversi:

Perintah	Fungsinya	Digunakan Saat
FIELDS TERMINATED BY	Tentukan pemisah antar kolom	Import & Export
ENCLOSED BY	Bungkus tiap nilai kolom	Import & Export
LINES TERMINATED BY	Tentukan pemisah antar baris	Import & Export
UNION ALL	Gabungkan hasil SELECT	Export (buat header)
IGNORE 1 ROWS	Lewati baris pertama	Import

6. Konversi dari SQL ke File Excel csv/txt

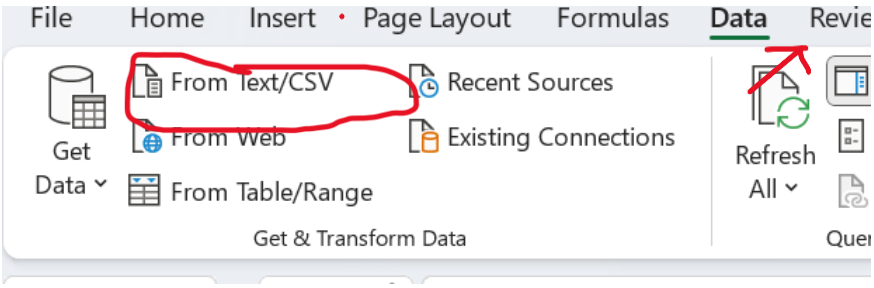
A. Tabel Penjualan → Penjualan.csv

- **Lakukan Command untuk Konversi SQL ke csv**

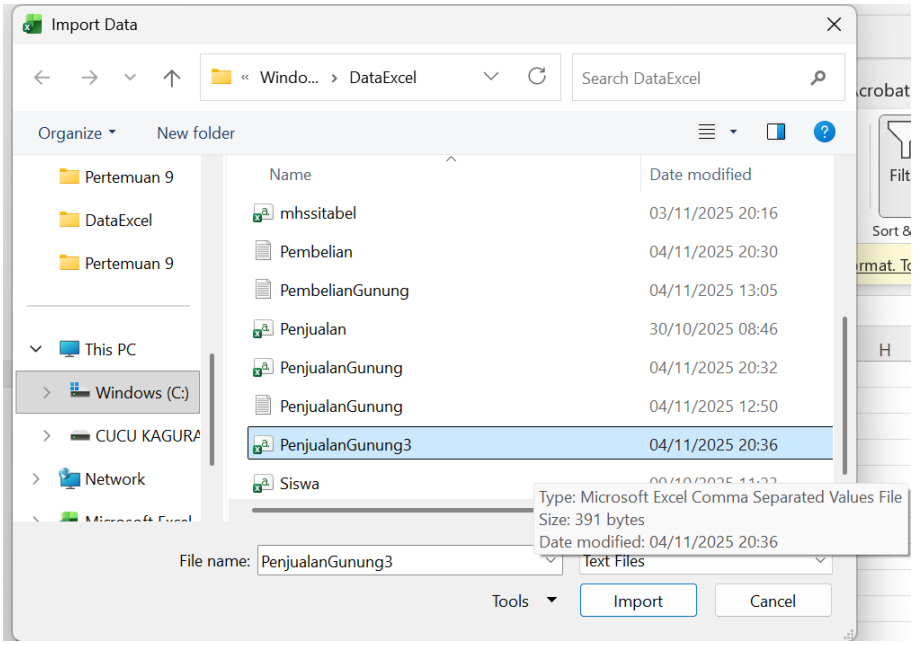
```
MariaDB [JualBeliGunung]> select 'idproduk','namaproduk','jenisproduk','harga','tersedia','garansi'
    -> UNION ALL
    -> select idproduk, namaproduk, jenisproduk, harga, tersedia, garansi from penjualan
    -> INTO OUTFILE 'C:\\DataExcel\\PenjualanGunung3.csv'
    -> FIELDS TERMINATED BY '\t'
    -> ENCLOSED BY '"'
    -> LINES TERMINATED BY '\r\n';
Query OK, 6 rows affected, 1 warning (0.002 sec)
```

- **Cara Membukanya di excel dan konversi nya menjadi tabel rapi**

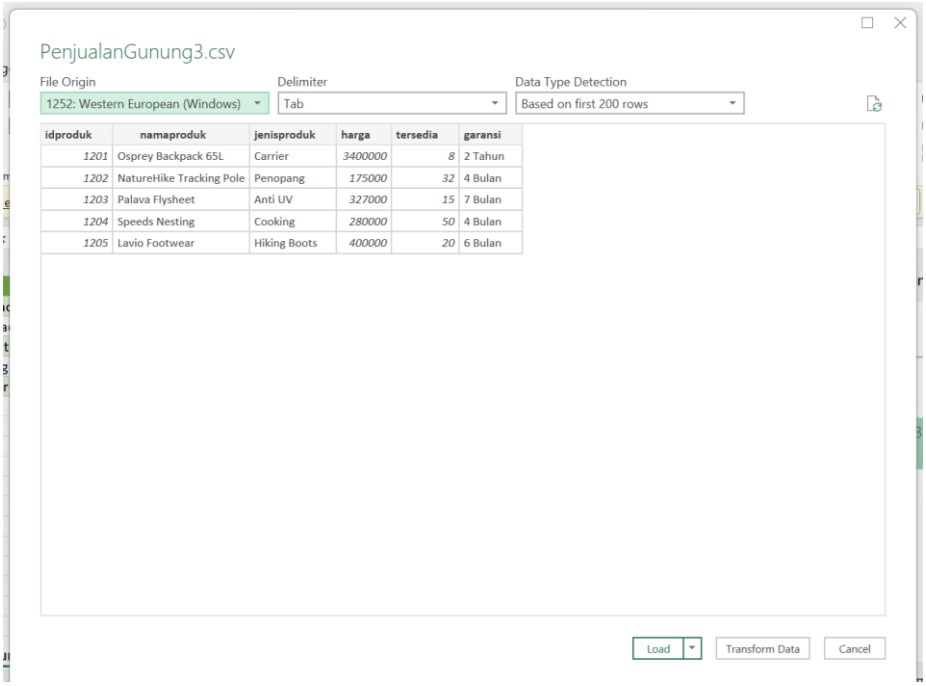
  Masuk ke Excel → ke Tab Ribbon Data lalu pilih From Text/CSV



- **Pilih file yang sudah di Konversi dari SQL**



- **File Origin 1252: Western European (Windows), Delimiter Tab, Based on First 200 Rows**



- **Setelah Load (Hasil):**

idproduk	namaproduk	jenisproduk	harga	tersedia	garansi
1201	Osprey Backpack 65L	Carrier	3400000	8	2 Tahun
1202	NatureHike Tracking Pole	Penopang	175000	32	4 Bulan
1203	Palava Flysheet	Anti UV	327000	15	7 Bulan
1204	Speeds Nesting	Cooking	280000	50	4 Bulan
1205	Lavio Footwear	Hiking Boots	400000	20	6 Bulan

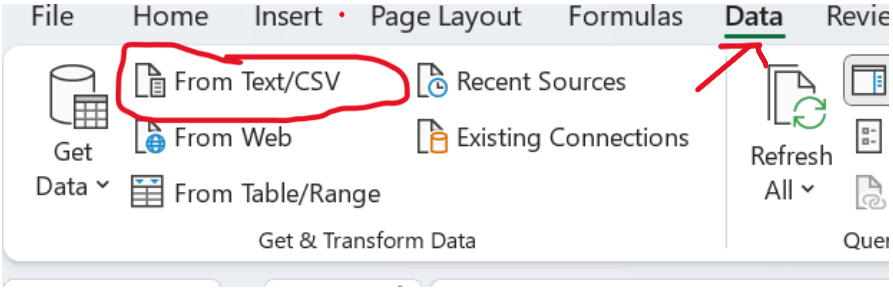
B. Tabel Pembelian → Pembelian.txt

- **Lakukan Command untuk Konversi dari SQL ke txt**

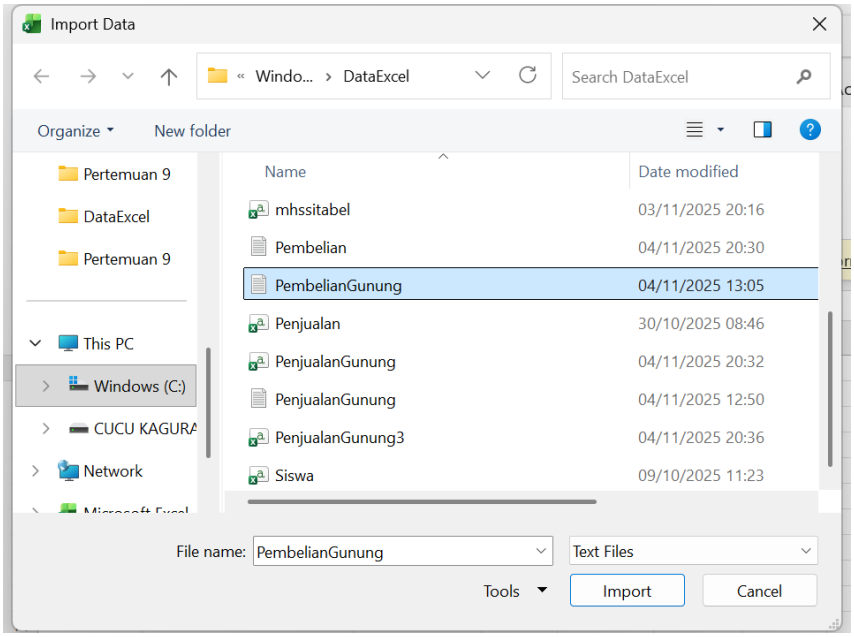
```
MariaDB [JualBeliGunung]> select 'idproduk','namaproduk','namapembeli','alamat','hargatotal','qty','pembayaran','tglbeli'
    -> UNION ALL
    -> select idproduk, namaproduk, namapembeli, alamat, hargatotal, qty, pembayaran, tglbeli from pembelian
    -> INTO OUTFILE 'C:\\DataExcel\\PembelianGunung.txt'
    -> FIELDS TERMINATED BY '\t'
    -> ENCLOSED BY '"'
    -> LINES TERMINATED BY '\r\n';
Query OK, 5 rows affected, 1 warning (0.003 sec)
```

- **Cara Membukanya di excel dan konversi nya menjadi tabel rapi**

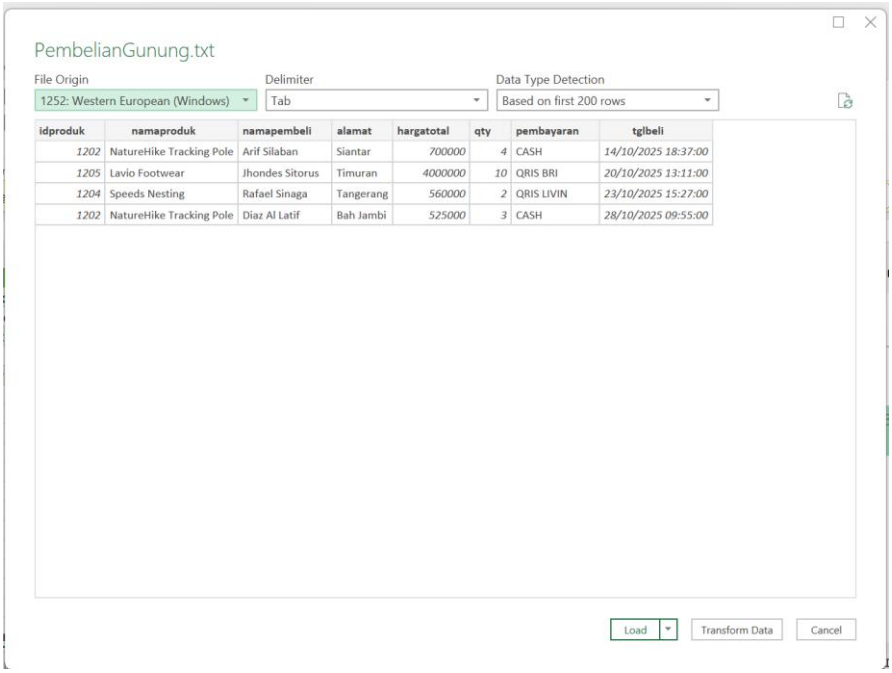
  Masuk ke Excel → ke Tab Ribbon Data lalu pilih From Text/CSV



- **Pilih file yang sudah di Konversi dari SQL**



- **File Origin 1252: Western European (Windows), Delimiter Tab, Based on First 200 Rows.**



- **Setelah Load (Hasil):**

	A	B	C	D	E	F	G	H
1	idproduk	namaproduk	namapembeli	alamat	hargatotal	qty	pembayaran	tglbeli
2	1202	NatureHike Tracking Pole	Arif Silaban	Siantar	700000	4	CASH	14/10/2025 18:37
3	1205	Lavio Footwear	Jhondes Sitorus	Timuran	4000000	10	QRIS BRI	20/10/2025 13:11
4	1204	Speeds Nesting	Rafael Sinaga	Tangerang	560000	2	QRIS LIVIN	23/10/2025 15:27
5	1202	NatureHike Tracking Pole	Diaz Al Latif	Bah Jambi	525000	3	CASH	28/10/2025 09:55
6								

BAB IV

JOIN

JOIN adalah perintah dalam SQL yang digunakan untuk menggabungkan data dari dua atau lebih tabel berdasarkan kolom yang saling berhubungan. JOIN memungkinkan pengguna melihat informasi yang tersebar di beberapa tabel seolah-olah berada dalam satu tabel besar. Jenis JOIN yang paling umum adalah INNER JOIN, LEFT JOIN, RIGHT JOIN, dan FULL JOIN (meskipun FULL JOIN tidak didukung pada beberapa DBMS seperti MySQL/MariaDB) (Amin, 2022). INNER JOIN hanya menampilkan data yang memiliki kecocokan di kedua tabel. LEFT JOIN menampilkan semua data dari tabel kiri, dan data dari tabel kanan yang cocok; jika tidak cocok, nilai akan berisi NULL.

Sebaliknya, RIGHT JOIN menampilkan semua data dari tabel kanan beserta data tabel kiri yang cocok. Sedangkan FULL JOIN menggabungkan semua data dari kedua tabel, baik yang cocok maupun tidak cocok (lebih umum di PostgreSQL/SQL Server). Contoh penggunaan JOIN adalah:

```
SELECT A.Nama, B.Nama_Kelas FROM tblguru A INNER
JOIN tblkelas B ON A.Kd_Kelas = B.Kd_Kelas;
```

JOIN merupakan konsep inti pada relational database. Tanpa JOIN, setiap tabel berdiri sendiri dan tidak dapat memberikan informasi yang lengkap. Sebagai contoh, jika tabel guru dan tabel mata pelajaran terpisah, informasi tentang guru yang mengajar mata pelajaran tertentu tidak bisa ditampilkan tanpa JOIN. Oleh karena itu, JOIN digunakan untuk menggabungkan data berdasarkan key yang sama. JOIN juga membantu mengurangi redundansi, karena data tidak perlu disimpan berulang pada setiap tabel, cukup disimpan sekali dan dihubungkan melalui relasi.

Masalah yang sering muncul ketika menggunakan JOIN adalah munculnya nilai NULL. Hal ini terjadi ketika data pada suatu tabel tidak memiliki pasangan pada tabel lain, terutama pada LEFT JOIN atau RIGHT JOIN. Selain itu, performa query juga dapat menjadi lambat jika jumlah tabel dan data sangat besar. Untuk mengatasinya, indeks pada kolom yang digunakan untuk JOIN harus dibuat, sehingga pencarian data lebih cepat.

Desain basis data yang baik akan mengurangi masalah ini dan membuat proses analisis menjadi efisien.

Dengan JOIN, relasi antar tabel dapat ditampilkan secara jelas dan memungkinkan analisis data lebih mendalam. JOIN adalah konsep utama dalam database relasional karena menghubungkan entitas yang sebelumnya terpisah dalam suatu sistem.

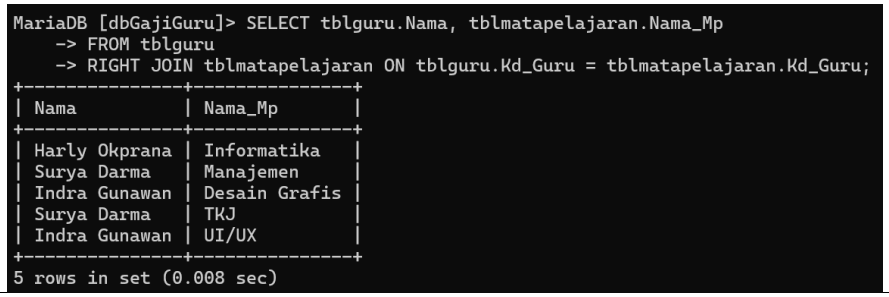
Sebelum menggunakan perintah ini jangan lupa untuk Membuat Database dbGajiGuru terlebih dahulu dan use database tersebut agar database bisa digunakan, Jika database sebelumnya sudah dibuat silahkan dilanjutkan dengan hanya use database saja untuk menggunakan dan melanjutkan progress database sebelumnya.

Soal :	Soal JOIN – 01
Syntax :	SELECT tblguru.Nama, tblmatapelajaran.Nama_Mp -> FROM tblguru -> INNER JOIN tblmatapelajaran ON tblguru.Kd_Guru = tblmatapelajaran.Kd_Guru;
Maksud :	Menampilkan Guru dengan mata pelajaran yang diajarkan.
Hasil	

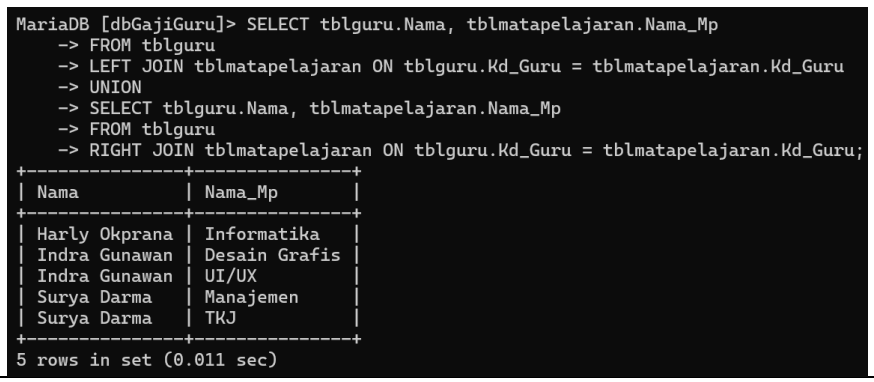
```
MariaDB [dbGajiGuru]> SELECT tblguru.Nama, tblmatapelajaran.Nama_Mp
    -> FROM tblguru
    -> INNER JOIN tblmatapelajaran ON tblguru.Kd_Guru = tblmatapelajaran.Kd_Guru;
+---------------+---------------+
| Nama          | Nama_Mp       |
+---------------+---------------+
| Harly Okprana | Informatika   |
| Surya Darma   | Manajemen     |
| Indra Gunawan | Desain Grafis |
| Surya Darma   | TKJ           |
| Indra Gunawan | UI/UX         |
+---------------+---------------+
5 rows in set (0.016 sec)
```

Soal :	Soal JOIN – 02
Syntax :	SELECT tblguru.Nama, tblmatapelajaran.Nama_Mp -> FROM tblguru -> LEFT JOIN tblmatapelajaran ON tblguru.Kd_Guru = tblmatapelajaran.Kd_Guru;
Maksud :	Tampilkan semua guru, meskipun belum punya mata pelajaran.
Hasil	

```
MariaDB [dbGajiGuru]> SELECT tblguru.Nama, tblmatapelajaran.Nama_Mp
    -> FROM tblguru
    -> LEFT JOIN tblmatapelajaran ON tblguru.Kd_Guru = tblmatapelajaran.Kd_Guru;
+---------------+---------------+
| Nama          | Nama_Mp       |
+---------------+---------------+
| Harly Okprana | Informatika   |
| Indra Gunawan | Desain Grafis |
| Indra Gunawan | UI/UX         |
| Surya Darma   | Manajemen     |
| Surya Darma   | TKJ           |
+---------------+---------------+
5 rows in set (0.009 sec)
```

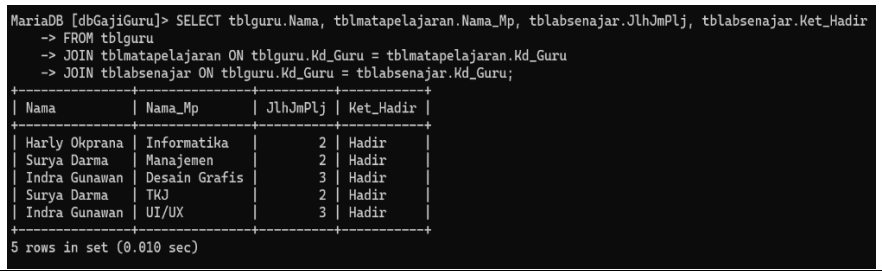
Soal :	Soal JOIN – 03
Syntax :	SELECT tblguru.Nama, tblmatapelajaran.Nama_Mp -> FROM tblguru -> RIGHT JOIN tblmatapelajaran ON tblguru.Kd_Guru = tblmatapelajaran.Kd_Guru;
Maksud :	Menampilkan semua mata pelajaran, meskipun tidak punya guru (jika ada data kosong)
Hasil	

```
MariaDB [dbGajiGuru]> SELECT tblguru.Nama, tblmatapelajaran.Nama_Mp
    -> FROM tblguru
    -> RIGHT JOIN tblmatapelajaran ON tblguru.Kd_Guru = tblmatapelajaran.Kd_Guru;
+---------------+---------------+
| Nama          | Nama_Mp       |
+---------------+---------------+
| Harly Okprana | Informatika   |
| Surya Darma   | Manajemen     |
| Indra Gunawan | Desain Grafis |
| Surya Darma   | TKJ           |
| Indra Gunawan | UI/UX         |
+---------------+---------------+
5 rows in set (0.008 sec)
```

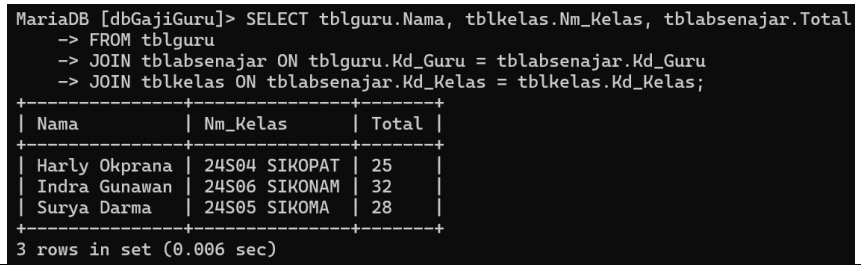
Soal :	Soal JOIN – 04
Syntax :	SELECT tblguru.Nama, tblmatapelajaran.Nama_Mp -> FROM tblguru -> LEFT JOIN tblmatapelajaran ON tblguru.Kd_Guru = tblmatapelajaran.Kd_Guru -> UNION -> SELECT tblguru.Nama, tblmatapelajaran.Nama_Mp -> FROM tblguru -> RIGHT JOIN tblmatapelajaran ON tblguru.Kd_Guru = tblmatapelajaran.Kd_Guru;
Maksud :	Menampilkan semua mata pelajaran dan guru Menggunakan FULL JOIN (UNION)
Hasil	

```
MariaDB [dbGajiGuru]> SELECT tblguru.Nama, tblmatapelajaran.Nama_Mp
    -> FROM tblguru
    -> LEFT JOIN tblmatapelajaran ON tblguru.Kd_Guru = tblmatapelajaran.Kd_Guru
    -> UNION
    -> SELECT tblguru.Nama, tblmatapelajaran.Nama_Mp
    -> FROM tblguru
    -> RIGHT JOIN tblmatapelajaran ON tblguru.Kd_Guru = tblmatapelajaran.Kd_Guru;
+---------------+---------------+
| Nama          | Nama_Mp       |
+---------------+---------------+
| Harly Okprana | Informatika   |
| Indra Gunawan | Desain Grafis |
| Indra Gunawan | UI/UX         |
| Surya Darma   | Manajemen     |
| Surya Darma   | TKJ           |
+---------------+---------------+
5 rows in set (0.011 sec)
```

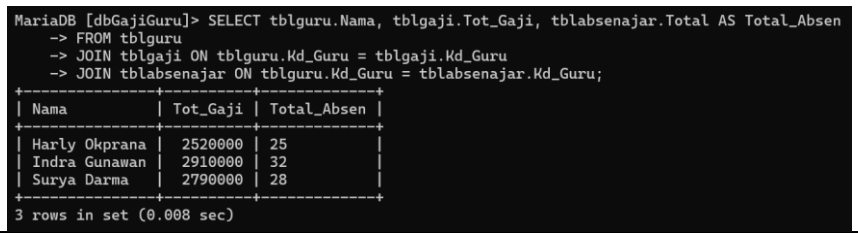
Soal :	Soal JOIN – 05
Syntax :	SELECT tblguru.Nama, tblmatapelajaran.Nama_Mp, tblabsenajar.JlhJmPlj, tblabsenajar.Ket_Hadir -> FROM tblguru -> JOIN tblmatapelajaran ON tblguru.Kd_Guru = tblmatapelajaran.Kd_Guru -> JOIN tblabsenajar ON tblguru.Kd_Guru = tblabsenajar.Kd_Guru;

Maksud :	JOIN 3 Tabel Guru + Mata Pelajaran + Absen
Hasil	

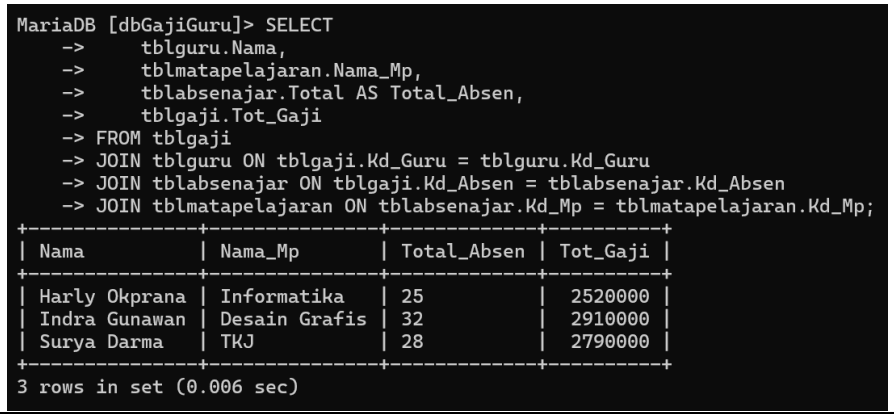
```
MariaDB [dbGajiGuru]> SELECT tblguru.Nama, tblmatapelajaran.Nama_Mp, tblabsenajar.JlhJmPlj, tblabsenajar.Ket_Hadir
    -> FROM tblguru
    -> JOIN tblmatapelajaran ON tblguru.Kd_Guru = tblmatapelajaran.Kd_Guru
    -> JOIN tblabsenajar ON tblguru.Kd_Guru = tblabsenajar.Kd_Guru;
+---------------+--------------+----------+-----------+
| Nama          | Nama_Mp      | JlhJmPlj | Ket_Hadir |
+---------------+--------------+----------+-----------+
| Harly Okprana | Informatika  |        2 | Hadir     |
| Surya Darma   | Manajemen    |        2 | Hadir     |
| Indra Gunawan | Desain Grafis|        3 | Hadir     |
| Surya Darma   | TKJ          |        2 | Hadir     |
| Indra Gunawan | UI/UX        |        3 | Hadir     |
+---------------+--------------+----------+-----------+
5 rows in set (0.010 sec)
```

Soal :	Soal JOIN – 06
Syntax :	SELECT tblguru.Nama, tblmatapelajaran.Nama_Mp, tblabsenajar.JlhJmPlj, tblabsenajar.Ket_Hadir -> FROM tblguru -> JOIN tblmatapelajaran ON tblguru.Kd_Guru = tblmatapelajaran.Kd_Guru -> JOIN tblabsenajar ON tblguru.Kd_Guru = tblabsenajar.Kd_Guru;
Maksud :	JOIN 3 Tabel Guru + Absen + Kelas (Seakan-akan mereka saat ini mengajar dikelas mana)
Hasil	

```
MariaDB [dbGajiGuru]> SELECT tblguru.Nama, tblkelas.Nm_Kelas, tblabsenajar.Total
    -> FROM tblguru
    -> JOIN tblabsenajar ON tblguru.Kd_Guru = tblabsenajar.Kd_Guru
    -> JOIN tblkelas ON tblabsenajar.Kd_Kelas = tblkelas.Kd_Kelas;
+---------------+---------------+-------+
| Nama          | Nm_Kelas      | Total |
+---------------+---------------+-------+
| Harly Okprana | 24S04 SIKOPAT | 25    |
| Indra Gunawan | 24S06 SIKONAM | 32    |
| Surya Darma   | 24S05 SIKOMA  | 28    |
+---------------+---------------+-------+
3 rows in set (0.006 sec)
```

Soal :	Soal JOIN – 07
Syntax :	SELECT tblguru.Nama, tblgaji.Tot_Gaji, tblabsenajar.Total AS Total_Absen -> FROM tblguru -> JOIN tblgaji ON tblguru.Kd_Guru = tblgaji.Kd_Guru -> JOIN tblabsenajar ON tblguru.Kd_Guru = tblabsenajar.Kd_Guru;
Maksud :	Join Guru + Gaji + Absen
Hasil	

```
MariaDB [dbGajiGuru]> SELECT tblguru.Nama, tblgaji.Tot_Gaji, tblabsenajar.Total AS Total_Absen
    -> FROM tblguru
    -> JOIN tblgaji ON tblguru.Kd_Guru = tblgaji.Kd_Guru
    -> JOIN tblabsenajar ON tblguru.Kd_Guru = tblabsenajar.Kd_Guru;
+---------------+----------+-------------+
| Nama          | Tot_Gaji | Total_Absen |
+---------------+----------+-------------+
| Harly Okprana |  2520000 | 25          |
| Indra Gunawan |  2910000 | 32          |
| Surya Darma   |  2790000 | 28          |
+---------------+----------+-------------+
3 rows in set (0.008 sec)
```

Soal :	Soal JOIN – 08
Syntax :	SELECT -> tblguru.Nama, -> tblmatapelajaran.Nama_Mp, -> tblabsenajar.Total AS Total_Absen, -> tblgaji.Tot_Gaji -> FROM tblgaji -> JOIN tblguru ON tblgaji.Kd_Guru = tblguru.Kd_Guru -> JOIN tblabsenajar ON tblgaji.Kd_Absen = tblabsenajar.Kd_Absen -> JOIN tblmatapelajaran ON tblabsenajar.Kd_Mp = tblmatapelajaran.Kd_Mp;
Maksud :	JOIN Detail Gaji Lengkap (Guru + Gaji + Absen + Mapel)
Hasil	

```
MariaDB [dbGajiGuru]> SELECT
    ->     tblguru.Nama,
    ->     tblmatapelajaran.Nama_Mp,
    ->     tblabsenajar.Total AS Total_Absen,
    ->     tblgaji.Tot_Gaji
    -> FROM tblgaji
    -> JOIN tblguru ON tblgaji.Kd_Guru = tblguru.Kd_Guru
    -> JOIN tblabsenajar ON tblgaji.Kd_Absen = tblabsenajar.Kd_Absen
    -> JOIN tblmatapelajaran ON tblabsenajar.Kd_Mp = tblmatapelajaran.Kd_Mp;
+---------------+---------------+-------------+-----------+
| Nama          | Nama_Mp       | Total_Absen | Tot_Gaji |
+---------------+---------------+-------------+-----------+
| Harly Okprana | Informatika   | 25          |  2520000 |
| Indra Gunawan | Desain Grafis | 32          |  2910000 |
| Surya Darma   | TKJ           | 28          |  2790000 |
+---------------+---------------+-------------+-----------+
3 rows in set (0.006 sec)
```

Latihan Soal:

Setelah kamu praktikkan soal-soal praktikum diatas, boleh dong diasah kembali skill nya, coba deh tanpa lihat praktikum kamu sebelumnya supaya melatih memori kamu:

NB: Lanjutkan dari Progress kamu pada Database dbBengkelUmum

1. Buat perintah JOIN untuk menampilkan Nama Pelanggan dan Tanggal Pesanan, tetapi hanya untuk Pelanggan yang berasal dari kota “Medan”.
2. Buat perintah LEFT JOIN untuk menampilkan semua Pelanggan, termasuk yang belum pernah membuat pesanan.
3. Buat perintah RIGHT JOIN untuk menampilkan semua Pesanan, termasuk jika ada IdPelanggan yang tidak ditemukan di tabel Pelanggan.
4. Buat perintah JOIN untuk menampilkan Jumlah Pesanan dari setiap Pelanggan dengan menggunakan GROUP BY pada Nama Pelanggan.

5. Buat perintah JOIN yang menampilkan Nama Pelanggan dan Tanggal Pesanan, tetapi hanya untuk pesanan yang terjadi pada tahun 2024.
6. Buat perintah JOIN untuk menampilkan Nama Pelanggan dan Kota, tetapi hanya untuk pelanggan yang memiliki minimal 1 pesanan.

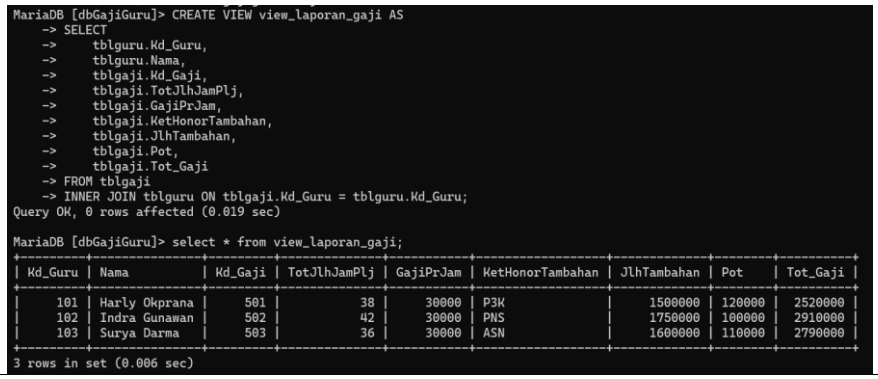
BAB V

VIEW

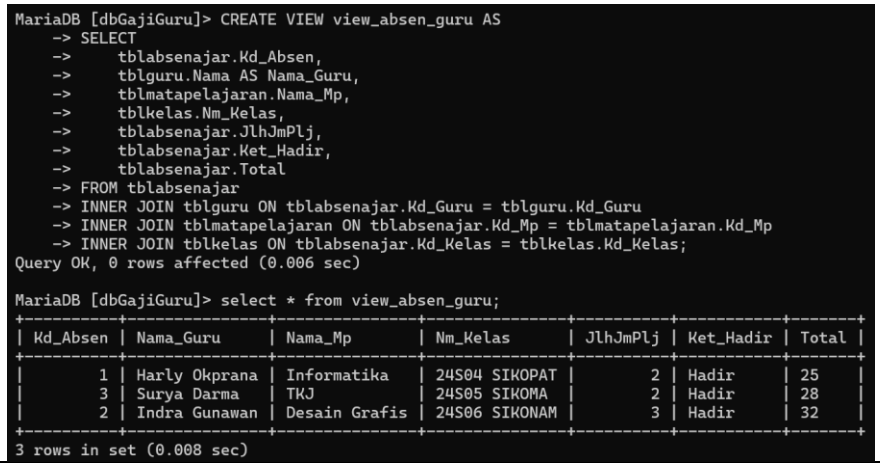
VIEW adalah objek dalam database SQL yang berfungsi sebagai tabel virtual. VIEW tidak menyimpan data secara langsung, tetapi menampilkan data dari satu atau lebih tabel lain berdasarkan query yang sudah ditentukan. VIEW sangat berguna untuk menyederhanakan query yang panjang, membatasi akses kolom sensitif, membuat laporan ringkas, dan memudahkan pengguna yang tidak memahami struktur database sesungguhnya (Arianti & Jamaluddin, 2022). Pada database dbGajiGuru, VIEW bisa digunakan untuk menampilkan laporan gaji guru, rekap bulanan, atau informasi penggabungan antara tabel guru, jabatan, dan gaji tanpa harus menulis JOIN berulang kali. Karena VIEW bersifat dinamis, setiap perubahan data pada tabel asli akan otomatis tercermin di VIEW, sehingga laporan tetap selalu up-to-date. VIEW juga bisa membantu membuat tampilan data yang lebih terstruktur dan aman tanpa memberikan akses langsung ke seluruh tabel. Meskipun VIEW sangat membantu, tidak semua VIEW dapat digunakan untuk INSERT, UPDATE, atau DELETE. VIEW yang menggunakan fungsi agregat atau group by tidak dapat dimodifikasi. Selain itu, performa dapat melambat jika VIEW menggunakan banyak tabel besar dengan JOIN yang rumit. Oleh karena itu, VIEW harus dirancang dengan mempertimbangkan kebutuhan pengguna dan efisiensi akses data. (Jamaluddin et al., 2022).

Sebelum menggunakan perintah ini jangan lupa untuk Membuat Database dbGajiGuru terlebih dahulu dan use database tersebut agar database bisa digunakan, Jika database sebelumnya sudah dibuat silahkan dilanjutkan dengan hanya use database saja untuk menggunakan dan melanjutkan progress database sebelumnya.

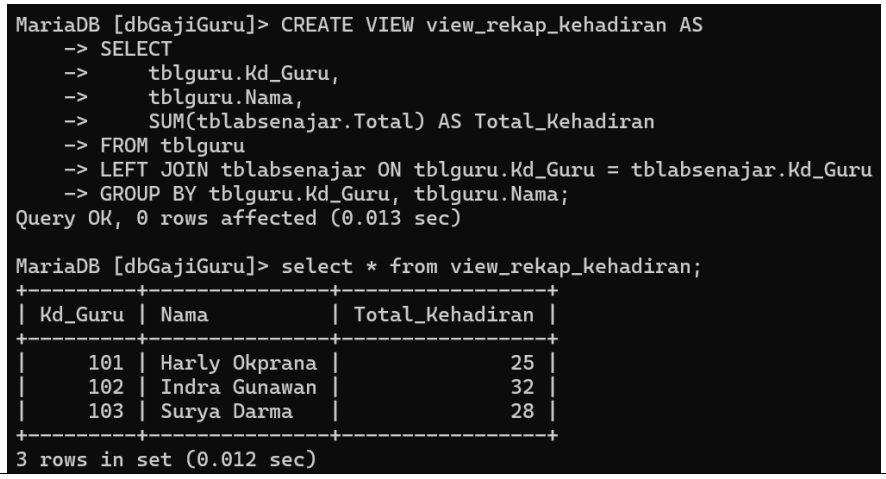
Soal :	Soal VIEW – 01
Syntax :	CREATE VIEW view_laporan_gaji AS -> SELECT -> tblguru.Kd_Guru, -> tblguru.Nama, -> tblgaji.Kd_Gaji, -> tblgaji.TotJlhJamPlj, -> tblgaji.GajiPrJam, -> tblgaji.KetHonorTambahan, -> tblgaji.JlhTambahan, -> tblgaji.Pot, -> tblgaji.Tot_Gaji

	-> FROM tblgaji -> INNER JOIN tblguru ON tblgaji.Kd_Guru = tblguru.Kd_Guru;
Maksud :	Membuat View Laporan Gaji
Hasil	

```
MariaDB [dbGajiGuru]> CREATE VIEW view_laporan_gaji AS
    -> SELECT
    ->     tblguru.Kd_Guru,
    ->     tblguru.Nama,
    ->     tblgaji.Kd_Gaji,
    ->     tblgaji.TotJlhJamPlj,
    ->     tblgaji.GajiPrJam,
    ->     tblgaji.KetHonorTambahan,
    ->     tblgaji.JlhTambahan,
    ->     tblgaji.Pot,
    ->     tblgaji.Tot_Gaji
    -> FROM tblgaji
    -> INNER JOIN tblguru ON tblgaji.Kd_Guru = tblguru.Kd_Guru;
Query OK, 0 rows affected (0.019 sec)

MariaDB [dbGajiGuru]> select * from view_laporan_gaji;
+---------+---------------+---------+-------------+-----------+-----------------+-------------+--------+----------+
| Kd_Guru | Nama          | Kd_Gaji | TotJlhJamPlj | GajiPrJam | KetHonorTambahan | JlhTambahan | Pot    | Tot_Gaji |
+---------+---------------+---------+-------------+-----------+-----------------+-------------+--------+----------+
|     101 | Harly Okprana |     501 |          38 |     30000 | P3K             |     1500000 | 120000 |  2520000 |
|     102 | Indra Gunawan |     502 |          42 |     30000 | PNS             |     1750000 | 100000 |  2910000 |
|     103 | Surya Darma   |     503 |          36 |     30000 | ASN             |     1600000 | 110000 |  2790000 |
+---------+---------------+---------+-------------+-----------+-----------------+-------------+--------+----------+
3 rows in set (0.006 sec)
```

Soal :	Soal VIEW – 02
Syntax :	CREATE VIEW view_absen_guru AS -> SELECT -> tblabsenajar.Kd_Absen, -> tblguru.Nama AS Nama_Guru, -> tblmatapelajaran.Nama_Mp, -> tblkelas.Nm_Kelas, -> tblabsenajar.JlhJmPlj, -> tblabsenajar.Ket_Hadir, -> tblabsenajar.Total -> FROM tblabsenajar -> INNER JOIN tblguru ON tblabsenajar.Kd_Guru = tblguru.Kd_Guru -> INNER JOIN tblmatapelajaran ON tblabsenajar.Kd_Mp = tblmatapelajaran.Kd_Mp -> INNER JOIN tblkelas ON tblabsenajar.Kd_Kelas = tblkelas.Kd_Kelas;
Maksud :	Membuat View Absen Guru
Hasil	

```
MariaDB [dbGajiGuru]> CREATE VIEW view_absen_guru AS
    -> SELECT
    ->     tblabsenajar.Kd_Absen,
    ->     tblguru.Nama AS Nama_Guru,
    ->     tblmatapelajaran.Nama_Mp,
    ->     tblkelas.Nm_Kelas,
    ->     tblabsenajar.JlhJmPlj,
    ->     tblabsenajar.Ket_Hadir,
    ->     tblabsenajar.Total
    -> FROM tblabsenajar
    -> INNER JOIN tblguru ON tblabsenajar.Kd_Guru = tblguru.Kd_Guru
    -> INNER JOIN tblmatapelajaran ON tblabsenajar.Kd_Mp = tblmatapelajaran.Kd_Mp
    -> INNER JOIN tblkelas ON tblabsenajar.Kd_Kelas = tblkelas.Kd_Kelas;
Query OK, 0 rows affected (0.006 sec)

MariaDB [dbGajiGuru]> select * from view_absen_guru;
+----------+---------------+---------------+---------------+----------+-----------+-------+
| Kd_Absen | Nama_Guru     | Nama_Mp       | Nm_Kelas      | JlhJmPlj | Ket_Hadir | Total |
+----------+---------------+---------------+---------------+----------+-----------+-------+
|        1 | Harly Okprana | Informatika   | 24S04 SIKOPAT |        2 | Hadir     | 25    |
|        3 | Surya Darma   | TKJ           | 24S05 SIKOMA  |        2 | Hadir     | 28    |
|        2 | Indra Gunawan | Desain Grafis | 24S06 SIKONAM |        3 | Hadir     | 32    |
+----------+---------------+---------------+---------------+----------+-----------+-------+
3 rows in set (0.008 sec)
```

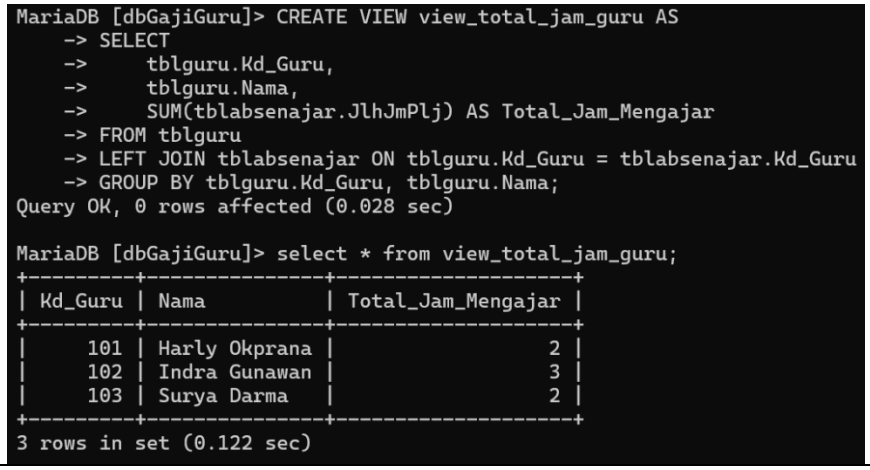
Soal :	Soal VIEW – 03
Syntax :	CREATE VIEW view_rekap_kehadiran AS -> SELECT -> tblguru.Kd_Guru, -> tblguru.Nama, -> SUM(tblabsenajar.Total) AS Total_Kehadiran -> FROM tblguru -> LEFT JOIN tblabsenajar ON tblguru.Kd_Guru = tblabsenajar.Kd_Guru -> GROUP BY tblguru.Kd_Guru, tblguru.Nama;
Maksud :	Membuat View Rekap Kehadiran
Hasil	

Soal :	Soal VIEW – 04
Syntax :	CREATE VIEW view_mp_guru AS -> SELECT -> tblguru.Kd_Guru, -> tblguru.Nama AS Nama_Guru, -> tblmatapelajaran.Nama_Mp -> FROM tblmatapelajaran -> INNER JOIN tblguru ON tblmatapelajaran.Kd_Guru = tblguru.Kd_Guru; Query OK, 0 rows affected (0.011 sec)
Maksud :	Membuat View Mata Pelajaran yang Diampu Oleh Guru
Hasil	

```
MariaDB [dbGajiGuru]> CREATE VIEW view_mp_guru AS
    -> SELECT
    ->     tblguru.Kd_Guru,
    ->     tblguru.Nama AS Nama_Guru,
    ->     tblmatapelajaran.Nama_Mp
    -> FROM tblmatapelajaran
    -> INNER JOIN tblguru ON tblmatapelajaran.Kd_Guru = tblguru.Kd_Guru;
Query OK, 0 rows affected (0.011 sec)

MariaDB [dbGajiGuru]> select * from view_mp_guru;
+---------+---------------+---------------+
| Kd_Guru | Nama_Guru     | Nama_Mp       |
+---------+---------------+---------------+
|     101 | Harly Okprana | Informatika   |
|     103 | Surya Darma   | Manajemen     |
|     102 | Indra Gunawan | Desain Grafis |
|     103 | Surya Darma   | TKJ           |
|     102 | Indra Gunawan | UI/UX         |
+---------+---------------+---------------+
5 rows in set (0.002 sec)
```

Soal :	Soal VIEW – 05
Syntax :	CREATE VIEW view_gaji_absen AS -> SELECT -> tblguru.Nama AS Nama_Guru, -> tblgaji.Tot_Gaji, -> tblabsenajar.Total AS Total_Absensi, -> tblabsenajar.JlhJmPlj AS Jam_Mengajar, -> tblmatapelajaran.Nama_Mp, -> tblkelas.Nm_Kelas -> FROM tblgaji -> INNER JOIN tblguru ON tblgaji.Kd_Guru = tblguru.Kd_Guru -> INNER JOIN tblabsenajar ON tblgaji.Kd_Absen = tblabsenajar.Kd_Absen -> INNER JOIN tblmatapelajaran ON tblabsenajar.Kd_Mp = tblmatapelajaran.Kd_Mp -> INNER JOIN tblkelas ON tblabsenajar.Kd_Kelas = tblkelas.Kd_Kelas;
Maksud :	Membuat View Gaji Absen
Hasil	

```
MariaDB [dbGajiGuru]> CREATE VIEW view_gaji_absen AS
    -> SELECT
    ->     tblguru.Nama AS Nama_Guru,
    ->     tblgaji.Tot_Gaji,
    ->     tblabsenajar.Total AS Total_Absensi,
    ->     tblabsenajar.JlhJmPlj AS Jam_Mengajar,
    ->     tblmatapelajaran.Nama_Mp,
    ->     tblkelas.Nm_Kelas
    -> FROM tblgaji
    -> INNER JOIN tblguru ON tblgaji.Kd_Guru = tblguru.Kd_Guru
    -> INNER JOIN tblabsenajar ON tblgaji.Kd_Absen = tblabsenajar.Kd_Absen
    -> INNER JOIN tblmatapelajaran ON tblabsenajar.Kd_Mp = tblmatapelajaran.Kd_Mp
    -> INNER JOIN tblkelas ON tblabsenajar.Kd_Kelas = tblkelas.Kd_Kelas;
Query OK, 0 rows affected (0.011 sec)

MariaDB [dbGajiGuru]> select * from view_gaji_absenl
    -> ;
ERROR 1146 (42S02): Table 'dbgajiguru.view_gaji_absenl' doesn't exist
MariaDB [dbGajiGuru]> select * from view_gaji_absen;
+---------------+----------+---------------+-------------+---------------+---------------+
| Nama_Guru     | Tot_Gaji | Total_Absensi | Jam_Mengajar | Nama_Mp       | Nm_Kelas      |
+---------------+----------+---------------+-------------+---------------+---------------+
| Harly Okprana |  2520000 | 25            |           2 | Informatika   | 24S04 SIKOPAT |
| Indra Gunawan |  2910000 | 32            |           3 | Desain Grafis | 24S06 SIKONAM |
| Surya Darma   |  2790000 | 28            |           2 | TKJ           | 24S05 SIKOMA  |
+---------------+----------+---------------+-------------+---------------+---------------+
```

Soal :	Soal VIEW – 06
Syntax :	CREATE VIEW view_total_jam_guru AS -> SELECT -> tblguru.Kd_Guru, -> tblguru.Nama, -> SUM(tblabsenajar.JlhJmPlj) AS Total_Jam_Mengajar -> FROM tblguru -> LEFT JOIN tblabsenajar ON tblguru.Kd_Guru = tblabsenajar.Kd_Guru -> GROUP BY tblguru.Kd_Guru, tblguru.Nama;
Maksud :	Membuat View Total Jam Pelajaran Guru (Ibarat Laporan Beban Kerja Guru)
Hasil	

Soal :	Soal VIEW – 07
Syntax :	CREATE VIEW view_penghasilan_perjam AS -> SELECT -> tblguru.Nama AS Nama_Guru, -> tblgaji.Tot_Gaji, -> tblgaji.TotJlhJamPlj, -> (tblgaji.Tot_Gaji / tblgaji.TotJlhJamPlj) AS Gaji_Per_Jam_Real -> FROM tblgaji -> JOIN tblguru ON tblgaji.Kd_Guru = tblguru.Kd_Guru;
Maksud :	Membuat View Gaji Guru Perjam
Hasil	

```
MariaDB [dbGajiGuru]> CREATE VIEW view_penghasilan_perjam AS
    -> SELECT
    ->     tblguru.Nama AS Nama_Guru,
    ->     tblgaji.Tot_Gaji,
    ->     tblgaji.TotJlhJamPlj,
    ->     (tblgaji.Tot_Gaji / tblgaji.TotJlhJamPlj) AS Gaji_Per_Jam_Real
    -> FROM tblgaji
    -> JOIN tblguru ON tblgaji.Kd_Guru = tblguru.Kd_Guru;
Query OK, 0 rows affected (0.015 sec)

MariaDB [dbGajiGuru]> select * from view_penghasilan_perjam;
+---------------+----------+--------------+-------------------+
| Nama_Guru     | Tot_Gaji | TotJlhJamPlj | Gaji_Per_Jam_Real |
+---------------+----------+--------------+-------------------+
| Harly Okprana |  2520000 |           38 |        66315.7895 |
| Indra Gunawan |  2910000 |           42 |        69285.7143 |
| Surya Darma   |  2790000 |           36 |        77500.0000 |
+---------------+----------+--------------+-------------------+
3 rows in set (0.008 sec)
```

Soal :	Soal VIEW – 08
Syntax :	CREATE VIEW view_dashboard AS -> SELECT -> (SELECT COUNT(*) FROM tblguru) AS Jumlah_Guru, -> (SELECT COUNT(*) FROM tblkelas) AS Jumlah_Kelas, -> (SELECT COUNT(*) FROM tblmatapelajaran) AS Jumlah_MataPelajaran, -> (SELECT SUM(Total) FROM tblabsenajar) AS Total_Jam_Mengajar, -> (SELECT SUM(Tot_Gaji) FROM tblgaji) AS Total_Gaji_Dibayarkan;
Maksud :	Membuat View Dashboard Guru
Hasil	

```
MariaDB [dbGajiGuru]> CREATE VIEW view_dashboard AS
    -> SELECT
    ->     (SELECT COUNT(*) FROM tblguru) AS Jumlah_Guru,
    ->     (SELECT COUNT(*) FROM tblkelas) AS Jumlah_Kelas,
    ->     (SELECT COUNT(*) FROM tblmatapelajaran) AS Jumlah_MataPelajaran,
    ->     (SELECT SUM(Total) FROM tblabsenajar) AS Total_Jam_Mengajar,
    ->     (SELECT SUM(Tot_Gaji) FROM tblgaji) AS Total_Gaji_Dibayarkan;
Query OK, 0 rows affected (0.022 sec)

MariaDB [dbGajiGuru]> select * from view_dashboard;
+-------------+-------------+---------------------+-------------------+----------------------+
| Jumlah_Guru | Jumlah_Kelas | Jumlah_MataPelajaran | Total_Jam_Mengajar | Total_Gaji_Dibayarkan |
+-------------+-------------+---------------------+-------------------+----------------------+
|           3 |           3 |                   6 |                85 |              8220000 |
+-------------+-------------+---------------------+-------------------+----------------------+
1 row in set (0.008 sec)
```

Nah, dari beberapa VIEW yang diatas, kamu bisa mengimajinasikan view seperti apa yang kamu mau konsepnya. Banyak jenis-jenis yang view yang bisa dibuat seperti contoh beberapa soal praktik diatas. VIEW tersebut diatas menggunakan beberapa gabungan beberapa jenis-jenis SELECT, JOIN, dll.

Latihan Soal:

Seperti biasa setelah praktikum, kita latih dulu ya skill kamu:

NB: Lanjutkan dari Progress kamu pada Database dbBengkelUmum

1. Buat sebuah VIEW dengan nama vwPelangganMedan yang berisi data pelanggan yang tinggal di kota Medan. View ini hanya menampilkan kolom IdPelanggan, Nama, dan Kota.
2. Buat VIEW dengan nama vwPesananLengkap yang menampilkan data pelanggan dan pesanan mereka. View ini menggunakan JOIN antara tabel Pelanggan dan tabel Pesanan, lalu tampilkan Nama Pelanggan, Kota, dan Tanggal Pesanan.
3. Buat VIEW dengan nama vwTotalPesanan untuk menampilkan jumlah pesanan dari setiap pelanggan. View ini menampilkan Nama Pelanggan dan total pesanan dengan fungsi COUNT.
4. Buat VIEW dengan nama vwPesanan2024 yang menampilkan semua pesanan yang terjadi pada tahun 2024.
5. Buat VIEW dengan nama vwPelangganTanpaPesanan untuk menampilkan pelanggan yang belum pernah membuat pesanan. Tampilkan Nama Pelanggan dan Kota.
6. Buat VIEW dengan nama vwTotalHarga yang menampilkan Nama Pelanggan, Tanggal Pesanan, dan Total Harga dari setiap pesanan.
7. Hapus VIEW yang bernama vwPesanan2024 dari database karena sudah tidak dipakai lagi.

BAB VI

STORED PROCEDURE

Stored Procedure adalah sekumpulan perintah SQL yang disimpan di dalam database dan dapat dijalankan secara berulang-ulang. Stored Procedure dapat menerima parameter, memproses logika tertentu, dan menghasilkan output. Stored procedure menjalankan perintah di sisi server, bukan di sisi client. Hal ini membuat proses lebih cepat dan konsisten, karena logika program tidak perlu dikirim ulang dari aplikasi. Stored procedure juga mudah dipelihara: jika aturan berubah, cukup ubah satu procedure dan seluruh aplikasi yang memanggilnya akan mengikuti perubahan tersebut. Fitur ini meningkatkan keamanan, karena hanya procedure yang diberikan akses, bukan tabel secara langsung.

Dalam konteks Database Terdistribusi, stored procedure sangat berguna karena:

- Mengurangi lalu lintas jaringan (karena logika dijalankan di sisi server).
- Meningkatkan performa dan konsistensi data.
- Menyediakan kontrol akses yang lebih baik terhadap data.
- Mempermudah integrasi antar node atau server database yang tersebar.

Sederhananya Stored Procedure ini seperti Template SQL yang dimana setelah kita buat, kita hanya perlu memanggilnya dengan 1 perintah saja.

Sebelum menggunakan perintah ini jangan lupa untuk Membuat Database dbGajiGuru terlebih dahulu dan use database tersebut agar database bisa digunakan, Jika database sebelumnya sudah dibuat silahkan dilanjutkan dengan hanya use database saja untuk menggunakan dan melanjutkan progress database sebelumnya.

Soal :	Soal PROCEDURE – 01
Syntax :	Create Procedure TambahMP(-> IN vKd_Mp int(10), -> IN vNama_Mp Varchar(50), -> IN vKd_Guru int(10)) -> BEGIN -> Insert into tblmatapelajaran (Kd_Mp, Nama_Mp, Kd_Guru) -> VALUES (vKd_Mp, vNama_Mp, vKd_Guru); -> END $$ (Setelah Query Selesai Jangan Lupa Mengembalikan Delimiter ke Semula dengan DELIMITER ;)

	DELIMITER ; Call TambahMP('303','RPL','103');
Maksud :	Membuat Procedure Tambah Mata Pelajaran

Hasil	Sebelum Procedure di Panggil:
	`MariaDB [dbGajiGuru]> select * from tblmatapelajaran;`
	`+-------+---------------+---------+`
	`| Kd_Mp | Nama_Mp | Kd_Guru |`
	`+-------+---------------+---------+`
	`| 301 | Informatika | 101 |`
	`| 302 | Manajemen | 103 |`
	`| 304 | Desain Grafis | 102 |`
	`| 305 | TKJ | 103 |`
	`| 306 | UI/UX | 102 |`
	`+-------+---------------+---------+`
	
	Setelah Prosedur Dipanggil:
	
	`MariaDB [dbGajiGuru]> Create Procedure TambahMP(`
	`-> IN vKd_Mp int(10),`
	`-> IN vNama_Mp Varchar(50),`
	`-> IN vKd_Guru int(10))`
	`-> BEGIN`
	`-> Insert into tblmatapelajaran (Kd_Mp, Nama_Mp, Kd_Guru)`
	`-> VALUES (vKd_Mp, vNama_Mp, vKd_Guru);`
	`-> END $$`
	`Query OK, 0 rows affected (0.016 sec)`
	
	`MariaDB [dbGajiGuru]> DELIMITER ;`
	`MariaDB [dbGajiGuru]> Call TambahMP('303','RPL','103');`
	`Query OK, 1 row affected (0.014 sec)`
	
	`MariaDB [dbGajiGuru]> Select * from tblmatapelajaran;`
	`+-------+---------------+---------+`
	`| Kd_Mp | Nama_Mp | Kd_Guru |`
	`+-------+---------------+---------+`
	`| 301 | Informatika | 101 |`
	`| 302 | Manajemen | 103 |`
	`| 303 | RPL | 103 |`
	`| 304 | Desain Grafis | 102 |`
	`| 305 | TKJ | 103 |`
	`| 306 | UI/UX | 102 |`
	`+-------+---------------+---------+`
	`6 rows in set (0.002 sec)`

Soal :	Soal PROCEDURE – 02
Syntax :	DELIMITER $$ CREATE PROCEDURE TambahGuru(-> IN vKd_Guru INT, -> IN vNama VARCHAR(20), -> IN vTmp_Lahir VARCHAR(50), -> IN vTgl_Lahir DATE, -> IN vJenkel ENUM('Pria','Wanita'), -> IN vAgama VARCHAR(20), -> IN vStatus_Serti VARCHAR(20), -> IN vIjazah_Terakhir VARCHAR(10), -> IN vTmt VARCHAR(10),

	-> IN vAlamat VARCHAR(50), -> IN vNo_Telp VARCHAR(12) ->) -> BEGIN -> INSERT INTO tblguru(-> Kd_Guru, Nama, Tmp_Lahir, Tgl_Lahir, Jenkel, -> Agama, Status_Serti, Ijazah_Terakhir, Tmt, Alamat, No_Telp) VALUES (-> vKd_Guru, vNama, vTmp_Lahir, vTgl_Lahir, vJenkel, -> vAgama, vStatus_Serti, vIjazah_Terakhir, vTmt, vAlamat, vNo_Telp ->); -> END $$ DELIMITER ;
Maksud :	Membuat Procedure Tambah Guru
Hasil	Sebelum Procedure di Panggil: MariaDB [dbGajiGuru]> select * from tblguru; +---------+---------------+-----------+------------+--------+--------+-----------------+-----------------+------+-----------------+--------------+ | Kd_Guru | Nama | Tmp_Lahir | Tgl_Lahir | Jenkel | Agama | Status_Serti | Ijazah_Terakhir | Tmt | Alamat | No_Telp | +---------+---------------+-----------+------------+--------+--------+-----------------+-----------------+------+-----------------+--------------+ | 101 | Harly Okprana | Paris | 1990-08-17 | Pria | Islam | Sudah Sertifikasi | Magister | UPP | Pematangsiantar | 089636808128 | | 102 | Indra Gunawan | Jerman | 1990-11-15 | Pria | Islam | Sudah Sertifikasi | Magister | USU | Pematangsiantar | 081278498721 | | 103 | Surya Darma | Indonesia | 1986-11-24 | Pria | Islam | Sudah Sertifikasi | Magister | UPP | Pematangsiantar | 087983213451 | +---------+---------------+-----------+------------+--------+--------+-----------------+-----------------+------+-----------------+--------------+ 3 rows in set (0.001 sec) Setelah Prosedur Dipanggil: MariaDB [dbGajiGuru]> DELIMITER $$ MariaDB [dbGajiGuru]> CREATE PROCEDURE TambahGuru(-> IN vKd_Guru INT, -> IN vNama VARCHAR(20), -> IN vTmp_Lahir VARCHAR(50), -> IN vTgl_Lahir DATE, -> IN vJenkel ENUM('Pria','Wanita'), -> IN vAgama VARCHAR(20), -> IN vStatus_Serti VARCHAR(20), -> IN vIjazah_Terakhir VARCHAR(10), -> IN vTmt VARCHAR(10), -> IN vAlamat VARCHAR(50), -> IN vNo_Telp VARCHAR(12) ->) -> BEGIN -> INSERT INTO tblguru(-> Kd_Guru, Nama, Tmp_Lahir, Tgl_Lahir, Jenkel, -> Agama, Status_Serti, Ijazah_Terakhir, Tmt, Alamat, No_Telp ->) VALUES (-> vKd_Guru, vNama, vTmp_Lahir, vTgl_Lahir, vJenkel, -> vAgama, vStatus_Serti, vIjazah_Terakhir, vTmt, vAlamat, vNo_Telp ->); -> END $$ Query OK, 0 rows affected (0.014 sec) MariaDB [dbGajiGuru]> DELIMITER ; MariaDB [dbGajiGuru]> Call TambahGuru('104','Poningsih','Indonesia','1988-09-10','Wanita','Islam','Sudah Sertifikasi','Doktor','USU','Pematangsiantar','0878123816990'); Query OK, 1 row affected, 1 warning (0.026 sec) MariaDB [dbGajiGuru]> select * from tblguru; +---------+---------------+-----------+------------+--------+--------+-----------------+-----------------+------+-----------------+--------------+ | Kd_Guru | Nama | Tmp_Lahir | Tgl_Lahir | Jenkel | Agama | Status_Serti | Ijazah_Terakhir | Tmt | Alamat | No_Telp | +---------+---------------+-----------+------------+--------+--------+-----------------+-----------------+------+-----------------+--------------+ | 101 | Harly Okprana | Paris | 1990-08-17 | Pria | Islam | Sudah Sertifikasi | Magister | UPP | Pematangsiantar | 089636808128 | | 102 | Indra Gunawan | Jerman | 1990-11-15 | Pria | Islam | Sudah Sertifikasi | Magister | USU | Pematangsiantar | 081278498721 | | 103 | Surya Darma | Indonesia | 1986-11-24 | Pria | Islam | Sudah Sertifikasi | Magister | UPP | Pematangsiantar | 087983213451 | | 104 | Poningsih | Indonesia | 1988-09-10 | Wanita | Islam | Sudah Sertifikasi | Doktor | USU | Pematangsiantar | 087812381699 | +---------+---------------+-----------+------------+--------+--------+-----------------+-----------------+------+-----------------+--------------+ 4 rows in set (0.001 sec)

Soal :	Soal PROCEDURE – 03
Syntax :	DELIMITER $$ Create Procedure HapusMP(IN vKd_Mp INT) -> BEGIN -> DELETE From tblmatapelajaran

	-> WHERE Kd_Mp = vKd_Mp; -> END $$ DELIMITER ; Call HapusMP('308');
Maksud :	Membuat Procedure Hapus Mata Pelajaran
Hasil	Sebelum Procedure di Panggil: MariaDB [dbGajiGuru]> Select * from tblmatapelajaran; +-------+--------------+---------+ | Kd_Mp | Nama_Mp | Kd_Guru | +-------+--------------+---------+ | 301 | Informatika | 101 | | 302 | Manajemen | 103 | | 303 | RPL | 103 | | 304 | Desain Grafis | 102 | | 305 | TKJ | 103 | | 306 | UI/UX | 102 | | 307 | APSI | 104 | | 308 | Fuzzy Logic | 104 | +-------+--------------+---------+ 8 rows in set (0.001 sec) Setelah Prosedur Dipanggil: MariaDB [dbGajiGuru]> Call HapusMP('308'); Query OK, 1 row affected (0.007 sec) MariaDB [dbGajiGuru]> Select * from tblmatapelajaran; +-------+--------------+---------+ | Kd_Mp | Nama_Mp | Kd_Guru | +-------+--------------+---------+ | 301 | Informatika | 101 | | 302 | Manajemen | 103 | | 303 | RPL | 103 | | 304 | Desain Grafis | 102 | | 305 | TKJ | 103 | | 306 | UI/UX | 102 | | 307 | APSI | 104 | +-------+--------------+---------+ 7 rows in set (0.001 sec)

Soal :	Soal PROCEDURE – 04
Syntax :	DELIMITER $$ Create Procedure CariMP(IN vKd_MP INT) -> BEGIN -> Select * from Tblmatapelajaran -> WHERE Kd_Mp = vKd_Mp; -> END $$ DELIMITER ; Call CariMP('304');
Maksud :	Membuat Procedure CariMP Untuk Melakukan Pencarian Mata Pelajaran Berdasarkan Kode Mata Pelajaran
Hasil	

```
MariaDB [dbGajiGuru]> DELIMITER $$
MariaDB [dbGajiGuru]> Create Procedure CariMP(IN vKd_MP INT)
    -> BEGIN
    -> Select * from Tblmatapelajaran
    -> WHERE Kd_Mp = vKd_Mp;
    -> END $$
Query OK, 0 rows affected (0.008 sec)

MariaDB [dbGajiGuru]> DELIMITER ;
MariaDB [dbGajiGuru]> Call CariMP(304);
+-------+---------------+---------+
| Kd_Mp | Nama_Mp       | Kd_Guru |
+-------+---------------+---------+
|   304 | Desain Grafis |     102 |
+-------+---------------+---------+
1 row in set (0.001 sec)
```

Soal :	Soal PROCEDURE – 05
Syntax :	DELIMITER $$ Create Procedure TampilGaji() -> BEGIN -> Select * from tblgaji; -> END $$ DELIMITER ; Call TampilGaji;
Maksud :	Membuat Procedure TampilGaji (Ini Merupakan Procedure Paling Sederhana)
Hasil	

```
MariaDB [dbGajiGuru]> DELIMITER $$
MariaDB [dbGajiGuru]> Create Procedure TampilGaji()
    -> BEGIN
    -> Select * from tblgaji;
    -> END $$
Query OK, 0 rows affected (0.035 sec)

MariaDB [dbGajiGuru]> DELIMITER ;
MariaDB [dbGajiGuru]> Call TampilGaji;
+---------+---------+----------+-------------+-----------+------------------+-------------+--------+----------+
| Kd_Gaji | Kd_Guru | Kd_Absen | TotJlhJamPlj | GajiPrJam | KetHonorTambahan | JlhTambahan | Pot    | Tot_Gaji |
+---------+---------+----------+-------------+-----------+------------------+-------------+--------+----------+
|     501 |     101 |        1 |          38 |     30000 | P3K              |     1500000 | 120000 |  2520000 |
|     502 |     102 |        2 |          42 |     30000 | PNS              |     1750000 | 100000 |  2910000 |
|     503 |     103 |        3 |          36 |     30000 | ASN              |     1600000 | 110000 |  2790000 |
+---------+---------+----------+-------------+-----------+------------------+-------------+--------+----------+
3 rows in set (0.089 sec)
```

Latihan Soal (Lanjutkan Progress dbBengkelUmum):

Buat 3 Stored Procedure bebas berdasarkan imajinasi kamu sendiri dari Database dbBengkel Umum

BAB VII

TRIGGER

Trigger adalah serangkaian tindakan yang dijalankan secara otomatis saat megoperasikan perubahan tertentu yang dilakukan pada tabel tertentu. Kejadian (event) yang dapat membangkitkan trigger umumnya berupa pernyataan INSERT, UPDATE, atau DELETE (Hadi, 2017). Berdasarkan ruang lingkupnya, trigger diklasifikasikan menjadi dua jenis : row trigger dan statement trigger. Trigger baris(row) mendefinisikan aksi untuk setiap baris tabel, trigger pernyataan hanya berlaku untuk setiap pernyataan INSERT, UPDATE, atau DELETE (Server & Hartama, 2000).

Dari sisi perilaku eksekusi, trigger dapat dibedakan menjadi beberapa jenis, namun umumnya ada dua jenis trigger : trigger BEFORE dan AFTER.

Kita dapat mendefinisikan maksimum enam jenis tindakan atau peristiwa dalam bentuk trigger :

- BEFORE INSERT, ini diaktifkan sebelum penyisipan data ke dalam tabel.
- AFTER INSERT, ini diaktifkan setelah penyisipan data ke dalam tabel.
- BEFORE UPDATE, ini diaktifkan sebelum pembaruan data dalam tabel.
- AFTER UPDATE, ini diaktifkan setelah pembaruan data dalam tabel.
- BEFORE DELETE, ini diaktifkan sebelum data dihapus dari tabel.
- AFTER UPDATE, ini diaktifkan setelah pengahapusan data sari tabel.

Trigger biasanya digunakan ketika perubahan data harus dicatat secara otomatis. Misalnya, ketika data gaji guru ditambah, trigger dapat langsung menghitung total gaji tanpa input manual. Trigger juga sering digunakan untuk audit, seperti mencatat siapa yang mengubah data dan kapan perubahan dilakukan. Dengan demikian, trigger membantu menjaga integritas data dan memberikan histori perubahan yang jelas tanpa campur tangan pengguna.

Perintah / Sintaks umum Trigger adalah :

```
DELIMITER $$
CREATE TRIGGER nama_trigger
{BEFORE | AFTER}{ INSERT | UPDATE | DELETE} ON nama_table
```

```
FOR EACH ROW
BEGIN
Kode SQL
END $$
DELIMITER ;
```

Keuntungan Trigger, antara lain :

1. Membantu mengotomatiskan perubahan data.
2. Memungkinkan kami menggunakan kembali query yang pernah ditulis.
3. Menyediakan metode untuk memeriksa integritas data database.
4. Membantu mendeteksi kesalahan pada level database.
5. Memungkinkan pengauditan data dengan mudah.

Ada pula kekurangan Trigger, antara lain :

1. Meningkatkan biaya overhead server.
2. Hanya menyediakan validasi yang diperluas yang tidak semua validasi dapat diakses di pemicu SQL.
3. Memecahkan masalah kesalahan karena Trigger adalah pekerjaan yang membosankan.
4. Dapat menyebabkan kesalahan logis dalam aplikasi meskipun ada sedikit kesalahan dalam query.
5. Kita bisa kehilangan data asli jika kita salah menyetel Trigger.

Sebelum menggunakan perintah ini jangan lupa untuk Membuat Database dbGajiGuru terlebih dahulu dan use database tersebut agar database bisa digunakan, Jika database sebelumnya sudah dibuat silahkan dilanjutkan dengan hanya use database saja untuk menggunakan dan melanjutkan progress database sebelumnya.

Soal :	Soal TRIGGER – 01
Syntax :	DELIMITER $$ Create Trigger HitungGajiInsert -> BEFORE INSERT ON tblgaji -> FOR EACH ROW -> BEGIN -> SET New.Tot_Gaji = (New.TotJlhJamPlj * New.GajiPrJam) -> + New.JlhTambahan – New.Pot;

	-> END $$ DELIMITER ;
Maksud :	Membuat Trigger HitungGajiInsert yang dimana fungsi trigger ini akan menghitung otomatis total gaji guru berdasarkan TotJlhJamPlj,GajiPrJam,JlhTambahan dan Potongan saat kita Insert Record.
Hasil	MariaDB [dbGajiGuru]> DELIMITER $$ MariaDB [dbGajiGuru]> Create Trigger HitungGajiInsert -> BEFORE INSERT ON tblgaji -> FOR EACH ROW -> BEGIN -> SET New.Tot_Gaji = (New.TotJlhJamPlj * New.GajiPrJam) -> + New.JlhTambahan - New.Pot; -> END $$ Query OK, 0 rows affected (0.022 sec) Perhatikan Insert Pada Tabel Gaji, Nah pada Kolom Tot_Gaji kita insert nya dengan 0, kemudian Trigger akan secara otomatis melaksanakan tugas nya untuk menghitung otomatis, sehingga Tot_Gaji Otomatis Menampilkan Hasil. MariaDB [dbGajiGuru]> Insert Into Tblgaji VALUES -> ('504','101','1','45','32000','PNS','1800000','75000','0'); Query OK, 1 row affected (0.006 sec) MariaDB [dbGajiGuru]> Select * from tblgaji; +---------+---------+----------+-------------+-----------+-----------------+-------------+--------+-----------+ | Kd_Gaji | Kd_Guru | Kd_Absen | TotJlhJamPlj | GajiPrJam | KetHonorTambahan | JlhTambahan | Pot | Tot_Gaji | +---------+---------+----------+-------------+-----------+-----------------+-------------+--------+-----------+ | 501 | 101 | 1 | 38 | 30000 | P3K | 1500000 | 120000 | 2520000 | | 502 | 102 | 2 | 42 | 30000 | PNS | 1750000 | 100000 | 2910000 | | 503 | 103 | 3 | 36 | 30000 | ASN | 1600000 | 110000 | 2790000 | | 504 | 101 | 1 | 45 | 32000 | PNS | 1800000 | 75000 | 3165000 | +---------+---------+----------+-------------+-----------+-----------------+-------------+--------+-----------+ 4 rows in set (0.001 sec)

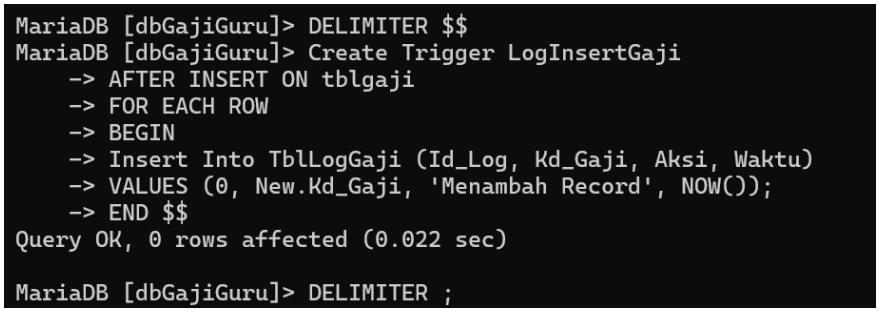
Soal :	Soal TRIGGER – 02
Syntax :	DELIMITER $$ Create Trigger HitungGajiUpdate -> BEFORE UPDATE ON tblgaji -> FOR EACH ROW -> BEGIN -> SET New.Tot_Gaji = (New.TotJlhJamPlj * New.GajiPrJam) -> + New.JlhTambahan - New.Pot; -> END $$ DELIMITER ;
Maksud :	Membuat Trigger HitungGajiUpdate yang dimana fungsi trigger ini akan menghitung otomatis total gaji guru berdasarkan TotJlhJamPlj,GajiPrJam,JlhTambahan dan Potongan Saat kita Mengubah salah satu Data pada Kolom hitungan Trigger.
Hasil	Data Sebelum di Update MariaDB [dbGajiGuru]> Select * from tblgaji; +---------+---------+----------+-------------+-----------+-----------------+-------------+--------+-----------+ | Kd_Gaji | Kd_Guru | Kd_Absen | TotJlhJamPlj | GajiPrJam | KetHonorTambahan | JlhTambahan | Pot | Tot_Gaji | +---------+---------+----------+-------------+-----------+-----------------+-------------+--------+-----------+ | 501 | 101 | 1 | 38 | 30000 | P3K | 1500000 | 120000 | 2520000 | | 502 | 102 | 2 | 42 | 30000 | PNS | 1750000 | 100000 | 2910000 | | 503 | 103 | 3 | 36 | 30000 | ASN | 1600000 | 110000 | 2790000 | | 504 | 101 | 1 | 45 | 32000 | PNS | 1800000 | 75000 | 3165000 | +---------+---------+----------+-------------+-----------+-----------------+-------------+--------+-----------+ 4 rows in set (0.011 sec)

Melakukan Update pada Salah Satu Record:

```
MariaDB [dbGajiGuru]> Update tblgaji SET GajiPrJam = '31500' Where Kd_Gaji = '502';
Query OK, 1 row affected (0.009 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Setelah Melakukan Update pada Gaji Per Jam yang ada pada Kd_Gaji 502, Trigger akan Melakukan Hitungan Otomatis pada Tot_Gaji tersebut karena Trigger Tersebut Otomatis berjalan saat kita melakukan Update Data. Sehingga Tot_Gaji pada Kd_gaji 502 berubah otomatis.

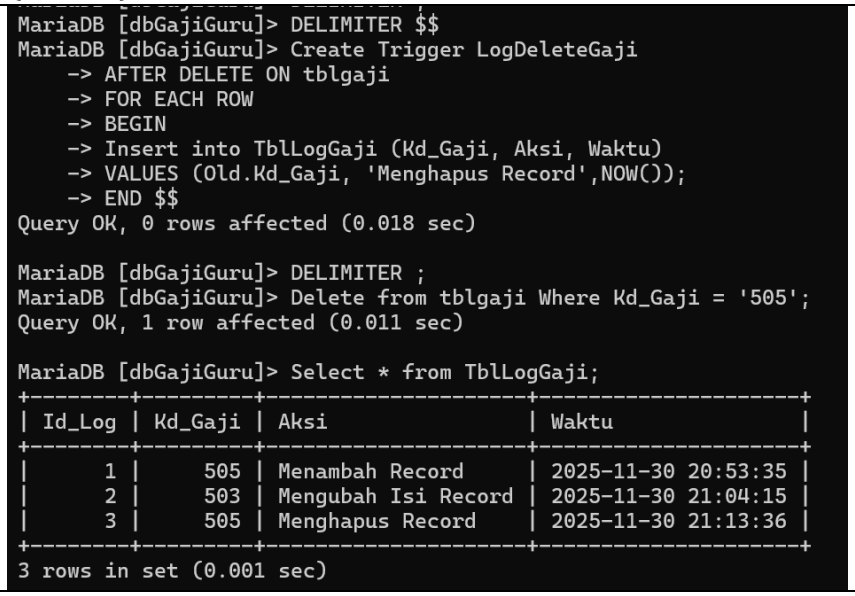
```
MariaDB [dbGajiGuru]> Select * from tblgaji;
+---------+---------+----------+--------------+-----------+-----------------+-------------+--------+-----------+
| Kd_Gaji | Kd_Guru | Kd_Absen | TotJlhJamPlj | GajiPrJam | KetHonorTambahan | JlhTambahan | Pot    | Tot_Gaji |
+---------+---------+----------+--------------+-----------+-----------------+-------------+--------+-----------+
|     501 |     101 |        1 |           38 |     30000 | P3K              |     1500000 | 120000 |  2520000 |
|     502 |     102 |        2 |           42 |     31500 | PNS              |     1750000 | 100000 |  2973000 |
|     503 |     103 |        3 |           36 |     30000 | ASN              |     1600000 | 110000 |  2790000 |
|     504 |     101 |        1 |           45 |     32000 | PNS              |     1800000 |  75000 |  3165000 |
+---------+---------+----------+--------------+-----------+-----------------+-------------+--------+-----------+
4 rows in set (0.001 sec)
```

Soal :	Soal TRIGGER – 03
Syntax :	DELIMITER $$ Create Trigger LogInsertGaji -> AFTER INSERT ON tblgaji -> FOR EACH ROW -> BEGIN -> Insert Into TblLogGaji (Id_Log, Kd_Gaji, Aksi, Waktu) -> VALUES (0, New.Kd_Gaji, 'Menambah Record', NOW()); -> END $$ DELIMITER ;
Maksud :	Membuat Trigger HitungGajiInsert yang dimana fungsi trigger ini akan menghitung otomatis total gaji guru berdasarkan TotJlhJamPlj,GajiPrJam,JlhTambahan dan Potongan saat kita Insert Record.
Hasil	 Perhatikan Record Pada Tabel Log Gaji, setelah melakukan Insert pada Tabel TblGaji, TblLogGaji langsung membuat record secara otomatis saat kita melakukan insert ke tabel tblgaji. (Fungsi AFTER INSERT)

```
MariaDB [dbGajiGuru]> Insert Into tblgaji VALUES
    -> ('505','101','1','39','30750','P3K','1660000','89000','0');
Query OK, 1 row affected (0.020 sec)

MariaDB [dbGajiGuru]> Select * from tblgaji;
+---------+---------+----------+-------------+-----------+-----------------+-------------+--------+----------+
| Kd_Gaji | Kd_Guru | Kd_Absen | TotJlhJamPlj | GajiPrJam | KetHonorTambahan | JlhTambahan | Pot    | Tot_Gaji |
+---------+---------+----------+-------------+-----------+-----------------+-------------+--------+----------+
|     501 |     101 |        1 |          38 |     30000 | P3K             |     1500000 | 120000 |  2520000 |
|     502 |     102 |        2 |          42 |     31500 | PNS             |     1750000 | 100000 |  2973000 |
|     503 |     103 |        3 |          36 |     30000 | ASN             |     1600000 | 110000 |  2790000 |
|     504 |     101 |        1 |          45 |     32000 | PNS             |     1800000 |  75000 |  3165000 |
|     505 |     101 |        1 |          39 |     30750 | P3K             |     1660000 |  89000 |  2770250 |
+---------+---------+----------+-------------+-----------+-----------------+-------------+--------+----------+
5 rows in set (0.001 sec)

MariaDB [dbGajiGuru]> select * from TblLogGaji;
+--------+---------+----------------+---------------------+
| Id_Log | Kd_Gaji | Aksi           | Waktu               |
+--------+---------+----------------+---------------------+
|      1 |     505 | Menambah Record | 2025-11-30 20:53:35 |
+--------+---------+----------------+---------------------+
1 row in set (0.001 sec)
```

Soal :	Soal TRIGGER – 04
Syntax :	DELIMITER $$ Create Trigger LogUpdateGaji -> AFTER UPDATE ON tblgaji -> FOR EACH ROW -> BEGIN -> Insert Into TblLogGaji (Kd_Gaji, Aksi, Waktu) -> VALUES (New.Kd_Gaji, 'Mengubah Isi Record', NOW()); -> END $$ DELIMITER ;
Maksud :	Membuat Trigger LogUpdateGaji. Sama seperti No.3 yang dimana TblLogGaji akan Otomatis membuat record sesuai dengan Perubahan pada Record yang diubah.
Hasil	MariaDB [dbGajiGuru]> DELIMITER $$ MariaDB [dbGajiGuru]> Create Trigger LogUpdateGaji -> AFTER UPDATE ON tblgaji -> FOR EACH ROW -> BEGIN -> Insert Into TblLogGaji (Kd_Gaji, Aksi, Waktu) -> VALUES (New.Kd_Gaji, 'Mengubah Isi Record', NOW()); -> END $$ Query OK, 0 rows affected (0.020 sec) MariaDB [dbGajiGuru]> DELIMITER ; Perhatikan Record Pada Tabel Log Gaji, setelah melakukan Update pada Tabel TblGaji, TblLogGaji langsung membuat record secara otomatis saat kita melakukan Update ke tabel tblgaji. (Fungsi AFTER UPDATE)

```
MariaDB [dbGajiGuru]> Update TblGaji SET GajiPrJam = '31250' Where Kd_Gaji = '503';
Query OK, 1 row affected (0.014 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [dbGajiGuru]> Select * from TblLogGaji;
+--------+---------+--------------------+---------------------+
| Id_Log | Kd_Gaji | Aksi               | Waktu               |
+--------+---------+--------------------+---------------------+
|      1 |     505 | Menambah Record    | 2025-11-30 20:53:35 |
|      2 |     503 | Mengubah Isi Record | 2025-11-30 21:04:15 |
+--------+---------+--------------------+---------------------+
2 rows in set (0.000 sec)
```

Soal :	Soal TRIGGER – 05
Syntax :	DELIMITER $$ Create Trigger LogDeleteGaji -> AFTER DELETE ON tblgaji -> FOR EACH ROW -> BEGIN -> Insert into TblLogGaji (Kd_Gaji, Aksi, Waktu) -> VALUES (Old.Kd_Gaji, 'Menghapus Record',NOW()); -> END $$ DELIMITER ;
Maksud :	Membuat Trigger LogDeleteGaji. Sama seperti No.3 dan 4 yang dimana TblLogGaji akan Otomatis membuat record sesuai dengan Perubahan pada Record yang diubah (Delete).
Hasil	

```
MariaDB [dbGajiGuru]> DELIMITER $$
MariaDB [dbGajiGuru]> Create Trigger LogDeleteGaji
    -> AFTER DELETE ON tblgaji
    -> FOR EACH ROW
    -> BEGIN
    -> Insert into TblLogGaji (Kd_Gaji, Aksi, Waktu)
    -> VALUES (Old.Kd_Gaji, 'Menghapus Record',NOW());
    -> END $$
Query OK, 0 rows affected (0.018 sec)

MariaDB [dbGajiGuru]> DELIMITER ;
MariaDB [dbGajiGuru]> Delete from tblgaji Where Kd_Gaji = '505';
Query OK, 1 row affected (0.011 sec)

MariaDB [dbGajiGuru]> Select * from TblLogGaji;
+--------+---------+--------------------+---------------------+
| Id_Log | Kd_Gaji | Aksi               | Waktu               |
+--------+---------+--------------------+---------------------+
|      1 |     505 | Menambah Record    | 2025-11-30 20:53:35 |
|      2 |     503 | Mengubah Isi Record | 2025-11-30 21:04:15 |
|      3 |     505 | Menghapus Record   | 2025-11-30 21:13:36 |
+--------+---------+--------------------+---------------------+
3 rows in set (0.001 sec)
```

Latihan Soal (Lanjutkan Progress dbBengkelUmum):

Buat 3 Trigger berdasarkan Imajinasi kamu sendiri dari database dbBengkelUmum, dan Buat 1 Tabel Log sebagai detail manipulasi datanya.

BAB VIII

GRANT & REVOKE

Perintah GRANT dan REVOKE digunakan untuk mengatur hak akses user terhadap database dan objek-objek di dalamnya seperti tabel, view, atau prosedur. GRANT berfungsi memberikan izin kepada user untuk melakukan operasi tertentu, sementara REVOKE berfungsi mencabut kembali izin tersebut. Pengaturan hak akses ini sangat penting agar tidak semua user dapat melakukan perubahan data yang bersifat sensitif, sehingga keamanan database tetap terjaga. Administrator biasanya memberikan hak akses terbatas sesuai dengan kebutuhan user, misalnya hanya boleh membaca data (SELECT), atau boleh menambah data tetapi tidak bisa menghapus (Collins et al., 2021).

Saat menggunakan GRANT, administrator menentukan jenis akses apa yang ingin diberikan, seperti SELECT, INSERT, UPDATE, DELETE, atau ALL PRIVILEGES jika ingin memberikan seluruh hak akses. Sebaliknya, REVOKE memungkinkan administrator mencabut sebagian atau seluruh hak akses jika user tersebut tidak lagi berkepentingan. Dengan mekanisme ini, database dapat dikelola secara lebih aman dan terstruktur, sehingga menghindari potensi kesalahan ataupun penyalahgunaan data (Trik, n.d.).

Keamanan adalah aspek penting dalam basis data. Tidak semua pengguna boleh mengakses semua data. Terdapat level akses seperti administrator, dosen, dan mahasiswa. Administrator dapat mengubah semua data, sedangkan mahasiswa hanya dapat melihat data tertentu. Pengaturan hak akses ini mencegah kesalahan, manipulasi, dan pencurian informasi. Sistem basis data modern selalu mengimplementasikan mekanisme otorisasi untuk memastikan bahwa setiap pengguna hanya dapat menjalankan perintah sesuai haknya.

Contoh penggunaan perintah GRANT dapat dilihat ketika administrator memberikan hak akses tertentu kepada user. Misalnya, untuk memberikan izin kepada user bernama userA agar dapat melihat data pada seluruh tabel di database dbGajiGuru, administrator menggunakan perintah:

```
GRANT SELECT ON dbGajiGuru.tblgaji TO 'userA'@'localhost';
```

Contoh lain, ketika ada user yang membutuhkan izin untuk menambah dan mengubah data pada tabel tblgaji , maka digunakan perintah:

```
GRANT INSERT, UPDATE ON dbGajiGuru.tblgaji TO 'userB'@'localhost';
```

Jika seorang user membutuhkan akses penuh sebagai pengelola database, administrator dapat memberikan seluruh hak akses menggunakan perintah:

```
GRANT ALL PRIVILEGES ON dbGajiGuru.tblgaji TO 'admin1'@'localhost';
```

Bahkan, saat membuat user baru seperti userC , password dapat langsung ditentukan, kemudian diberikan izin tertentu melalui perintah:

```
CREATE USER 'userC'@'localhost' IDENTIFIED BY 'password123';
```

```
GRANT SELECT, UPDATE ON dbGajiGuru.tblguru TO 'userC'@'localhost';
```

Sementara itu, contoh penggunaan REVOKE terlihat saat administrator ingin mencabut kembali hak akses yang pernah diberikan. Misalnya, untuk menghapus seluruh izin yang dimiliki userA, administrator dapat menggunakan perintah:

```
REVOKE ALL PRIVILEGES ON dbGajiGuru.tblgaji  FROM 'userA'@'localhost';
```

Jika hanya sebagian izin yang ingin dicabut, seperti hak untuk menghapus data pada tabel tblgaji , dapat digunakan perintah:

```
REVOKE DELETE ON dbGajiGuru.tblgaji FROM 'userB'@'localhost';
```

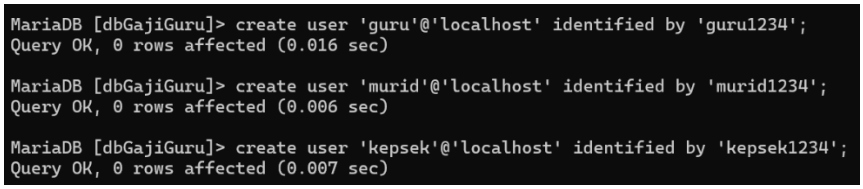
Selain mencabut hak akses, administrator juga dapat menghapus user sepenuhnya dari sistem menggunakan perintah:

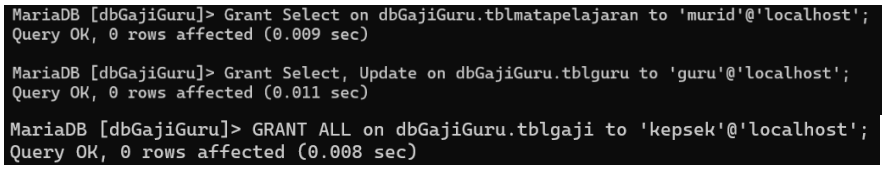
```
DROP USER 'userC'@'localhost';
```

Melalui kombinasi GRANT dan REVOKE ini, pengelolaan keamanan database dapat dijalankan dengan lebih terkontrol dan sesuai kebutuhan.

Sebelum menggunakan perintah ini jangan lupa untuk Membuat Database dbGajiGuru terlebih dahulu dan use database tersebut agar database bisa digunakan, Jika database sebelumnya sudah dibuat silahkan dilanjutkan dengan hanya use database saja untuk menggunakan dan melanjutkan progress database sebelumnya.

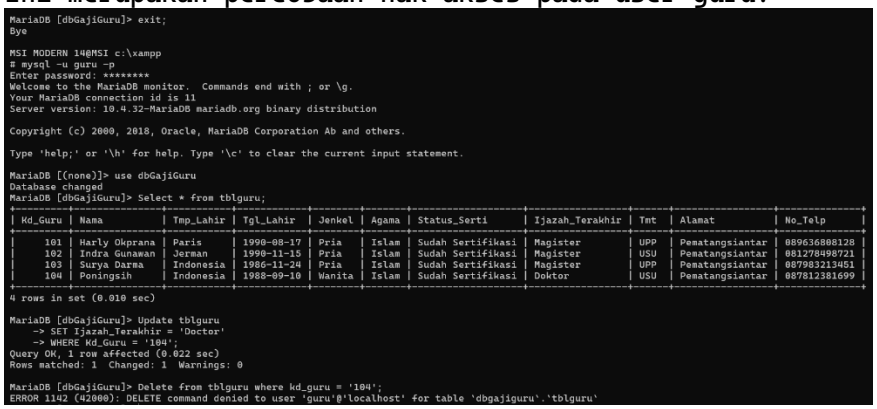
Soal :	Soal GRANT & REVOKE – 01

Syntax :	create user 'guru'@'localhost' identified by 'guru1234'; create user 'murid'@'localhost' identified by 'murid1234'; create user 'kepsek'@'localhost' identified by 'kepsek1234';
Maksud :	Membuat beberapa user dan password dari username tersebut.
Hasil	

Soal :	Soal GRANT & REVOKE – 02
Syntax :	Grant Select on dbGajiGuru.tblmatapelajaran to 'murid'@'localhost'; Grant Select, Update on dbGajiGuru.tblguru to 'guru'@'localhost'; GRANT ALL on dbGajiGuru.tblgaji to 'kepsek'@'localhost';
Maksud :	Membuat hak akses terhadap username yang ditentukan. Jika diperhatikan ada yang dbGajiGuru.tblmatapelajaran dan dbGajiGuru.tblguru, itu merupakan hak akses yang diberikan khusus untuk tabel mata pelajaran atau tabel guru.
Hasil	

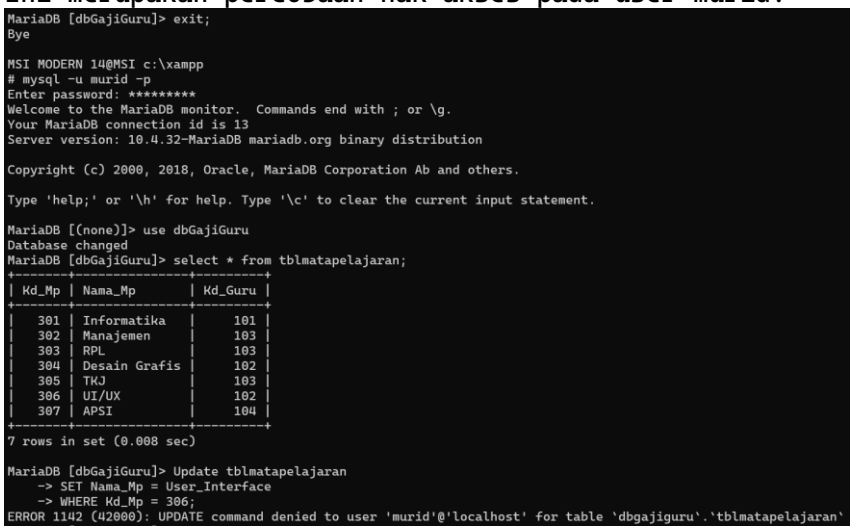
Soal :	Soal GRANT & REVOKE – 03
Syntax :	select user, host from mysql.user;
Maksud :	Menampilkan seluruh user yang ada pada mysql
Hasil	

```
MariaDB [dbGajiGuru]> select user, host from mysql.user;
+----------+-----------+
| User     | Host      |
+----------+-----------+
| root     | 127.0.0.1 |
| root     | ::1       |
| Owner    | localhost |
| Staf     | localhost |
| admin    | localhost |
| guru     | localhost |
| kepsek   | localhost |
| murid    | localhost |
| operator | localhost |
| pma      | localhost |
| root     | localhost |
| siswa    | localhost |
| staff    | localhost |
+----------+-----------+
13 rows in set (0.002 sec)
```

Soal :	Soal GRANT & REVOKE – 04
Syntax :	Exit; (Enter) Mysql –u guru –p (Enter) Enter Password: guru1234 (Enter) Use dbGajiGuru
Maksud :	Keluar terlebih dahulu, kemudian masukkan User dan Password yang sudah dibuat sebelumnya kemudian masuk ke database nya. Lalu gunakan hak akses yang digunakan pada tabel tertentu. User Guru Memiliki Hak Akses Select dan Update pada tabel tblguru.
Hasil	Ini merupakan percobaan hak akses pada user guru.  Jika diperhatikan ada perintah selain dari hak akses yang diberikan, Perintah akan ditolak dan akan muncul pesan error.

```
MariaDB [dbGajiGuru]> exit;
Bye

MSI MODERN 14@MSI c:\xampp
# mysql -u guru -p
Enter password: ********
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 11
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)]> use dbGajiGuru
Database changed
MariaDB [dbGajiGuru]> Select * from tblguru;
+---------+-------------+-----------+------------+--------+--------+-----------------+-----------------+------+-----------------+--------------+
| Kd_Guru | Nama        | Tmp_Lahir | Tgl_Lahir  | Jenkel | Agama  | Status_Serti    | Ijazah_Terakhir | Tmt  | Alamat          | No_Telp      |
+---------+-------------+-----------+------------+--------+--------+-----------------+-----------------+------+-----------------+--------------+
|     101 | Harly Okprana | Paris     | 1990-08-17 | Pria   | Islam  | Sudah Sertifikasi | Magister        | UPP  | Pematangsiantar | 089636808128 |
|     102 | Indra Gunawan | Jerman    | 1990-11-15 | Pria   | Islam  | Sudah Sertifikasi | Magister        | USU  | Pematangsiantar | 081278498721 |
|     103 | Surya Darma   | Indonesia | 1986-11-24 | Pria   | Islam  | Sudah Sertifikasi | Magister        | UPP  | Pematangsiantar | 087983213451 |
|     104 | Poningsih     | Indonesia | 1988-09-10 | Wanita | Islam  | Sudah Sertifikasi | Doktor          | USU  | Pematangsiantar | 087812381699 |
+---------+-------------+-----------+------------+--------+--------+-----------------+-----------------+------+-----------------+--------------+
4 rows in set (0.010 sec)

MariaDB [dbGajiGuru]> Update tblguru
    -> SET Ijazah_Terakhir = 'Doctor'
    -> WHERE Kd_Guru = '104';
Query OK, 1 row affected (0.022 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [dbGajiGuru]> Delete from tblguru where kd_guru = '104';
ERROR 1142 (42000): DELETE command denied to user 'guru'@'localhost' for table 'dbgajiguru'.'tblguru'
```

Soal :	Soal GRANT & REVOKE – 05
Syntax :	Exit; (Enter) Mysql –u murid –p (Enter) Enter Password: murid1234 (Enter)

	Use dbGajiGuru
Maksud :	Keluar terlebih dahulu, kemudian masukkan User dan Password yang sudah dibuat sebelumnya kemudian masuk ke database nya. Lalu gunakan hak akses yang digunakan pada tabel tertentu. User Murid Hanya Memiliki Hak Akses Select pada tabel tblmatapelajaran.
Hasil	Ini merupakan percobaan hak akses pada user murid.  Jika diperhatikan ada perintah selain dari hak akses yang diberikan, Perintah akan ditolak dan akan muncul pesan error.

```
MariaDB [dbGajiGuru]> exit;
Bye

MSI MODERN 14@MSI c:\xampp
# mysql -u murid -p
Enter password: *********
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 13
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)]> use dbGajiGuru
Database changed
MariaDB [dbGajiGuru]> select * from tblmatapelajaran;
+-------+---------------+---------+
| Kd_Mp | Nama_Mp       | Kd_Guru |
+-------+---------------+---------+
|   301 | Informatika   |     101 |
|   302 | Manajemen     |     103 |
|   303 | RPL           |     103 |
|   304 | Desain Grafis |     102 |
|   305 | TKJ           |     103 |
|   306 | UI/UX         |     102 |
|   307 | APSI          |     104 |
+-------+---------------+---------+
7 rows in set (0.008 sec)

MariaDB [dbGajiGuru]> Update tblmatapelajaran
    -> SET Nama_Mp = User_Interface
    -> WHERE Kd_Mp = 306;
ERROR 1142 (42000): UPDATE command denied to user 'murid'@'localhost' for table `dbgajiguru`.`tblmatapelajaran`
```

Soal :	Soal GRANT & REVOKE – 06
Syntax :	Exit; (Enter) Mysql –u kepsek –p (Enter) Enter Password: kepsek1234 (Enter) Use dbGajiGuru
Maksud :	Keluar terlebih dahulu, kemudian masukkan User dan Password yang sudah dibuat sebelumnya kemudian masuk ke database nya. Lalu gunakan hak akses yang digunakan pada tabel tertentu. User Kepsek Memiliki Hak Akses seluruh pada tabel tblgaji.
Hasil	Ini merupakan percobaan hak akses pada user kepsek.

```
MariaDB [dbGajiGuru]> exit;
Bye

MSI MODERN 14@MSI c:\xampp
# mysql -u kepsek -p
Enter password: **********
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 17
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)]> use dbGajiGuru
Database changed
MariaDB [dbGajiGuru]> select * from tblgaji;
+---------+---------+----------+--------------+-----------+------------------+-------------+--------+-----------+
| Kd_Gaji | Kd_Guru | Kd_Absen | TotJlhJamPlj | GajiPrJam | KetHonorTambahan | JlhTambahan | Pot    | Tot_Gaji |
+---------+---------+----------+--------------+-----------+------------------+-------------+--------+-----------+
|     501 |     101 |        1 |           38 |     30000 | P3K              |     1500000 | 120000 |  2520000 |
|     502 |     102 |        2 |           42 |     31500 | PNS              |     1750000 | 100000 |  2973000 |
|     503 |     103 |        3 |           36 |     31250 | ASN              |     1600000 | 110000 |  2615000 |
|     504 |     101 |        1 |           45 |     32000 | PNS              |     1800000 |  75000 |  3165000 |
+---------+---------+----------+--------------+-----------+------------------+-------------+--------+-----------+
4 rows in set (0.001 sec)

MariaDB [dbGajiGuru]> Update tblgaji
    -> SET Pot = '8000'
    -> Where Kd_Gaji = '504';
Query OK, 1 row affected (0.027 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [dbGajiGuru]> delete from tblgaji where kd_gaji = '504';
Query OK, 1 row affected (0.011 sec)
```

Soal :	Soal GRANT & REVOKE – 07
Syntax :	Revoke Update on dbGajiGuru.tblguru from ‘guru’@’localhost’;
Maksud :	Mencabut Hak Akses Update terhadap user guru pada tabel tblguru.
Hasil	Hak Akses Update tidak lagi berlaku untuk user Guru karena dicabut (REVOKE). Begitu juga dengan user lain jika ingin mencabut hak akses (SELECT/UPDATE/DELETE) yang ingin dicabut.

```
MariaDB [dbGajiGuru]> Revoke Update on dbGajiGuru.tblguru from 'guru'@'localhost';
Query OK, 0 rows affected (0.008 sec)

MariaDB [dbGajiGuru]> exit;
Bye

MSI MODERN 14@MSI c:\xampp
# mysql -u guru -p
Enter password: ********
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 19
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)]> use dbGajiGuru
Database changed
MariaDB [dbGajiGuru]> Select * from tblguru;
+---------+---------------+-----------+------------+---------+-------+------------------+-----------------+------+-----------------+-------------+
| Kd_Guru | Nama          | Tmp_Lahir | Tgl_Lahir  | Jenkel  | Agama | Status_Serti     | Ijazah_Terakhir | Tmt  | Alamat          | No_Telp     |
+---------+---------------+-----------+------------+---------+-------+------------------+-----------------+------+-----------------+-------------+
|     101 | Harly Okprana | Paris     | 1990-08-17 | Pria    | Islam | Sudah Sertifikasi | Magister        | UPP  | Pematangsiantar | 089636808128 |
|     102 | Indra Gunawan | Jerman    | 1990-11-15 | Pria    | Islam | Sudah Sertifikasi | Magister        | USU  | Pematangsiantar | 081278498721 |
|     103 | Surya Darma   | Indonesia | 1986-11-24 | Pria    | Islam | Sudah Sertifikasi | Magister        | UPP  | Pematangsiantar | 087983213451 |
|     104 | Poningsih     | Indonesia | 1988-09-10 | Wanita  | Islam | Sudah Sertifikasi | Doctor          | USU  | Pematangsiantar | 087812381699 |
+---------+---------------+-----------+------------+---------+-------+------------------+-----------------+------+-----------------+-------------+
4 rows in set (0.001 sec)

MariaDB [dbGajiGuru]> Update tblguru
    -> SET Ijazah_Terakhir = 'Doktor'
    -> WHERE Kd_Guru = '104';
ERROR 1142 (42000): UPDATE command denied to user 'guru'@'localhost' for table `dbgajiguru`.`tblguru`
```

Latihan Soal:

Buat username dan password beserta hak akses versi kamu sendiri.

BAB IX

ENTITY RELATIONSHIP DIAGARAM

Entity Relationship Diagram (ERD) adalah sebuah model visual yang digunakan untuk menggambarkan struktur dan hubungan antar data dalam sebuah sistem basis data. ERD menjadi langkah penting sebelum pembuatan database karena mampu menunjukkan objek data yang terlibat, atribut yang dimiliki, serta bagaimana objek-objek tersebut saling berhubungan. Dengan menggunakan ERD, perancang sistem dapat melihat gambaran menyeluruh sehingga desain database menjadi lebih terarah dan terstruktur. ERD biasanya disusun pada tahap analisis dan perancangan, sebelum database diimplementasikan pada MySQL atau sistem manajemen basis data lainnya (Garcia et al., n.d.).

Pada ERD, objek data disebut sebagai entitas. Entitas adalah sesuatu yang dapat digambarkan dan memiliki informasi yang perlu disimpan. Contohnya adalah entitas Guru, Siswa, Mata Pelajaran, atau Kelas. Setiap entitas memiliki atribut, yaitu detail informasi yang menjelaskan entitas tersebut, seperti Nama, Alamat, No_Telp, atau Kode. Entitas ditampilkan dalam bentuk kotak, sedangkan atribut dituliskan di dalam atau di bawah nama entitas. Melalui atribut ini, database mampu menyimpan informasi yang diperlukan secara lengkap dan terorganisir (Anggoro Dimas Aryo et al., 2021).

Selain entitas dan atribut, ERD juga menunjukkan hubungan antar data melalui relasi. Relasi adalah penghubung antara dua entitas yang menunjukkan bagaimana data saling berhubungan. Relasi digambarkan menggunakan garis penghubung, dan dalam model konseptual hubungan ditampilkan menggunakan bentuk belah ketupat. Pentingnya relasi adalah untuk menghindari duplikasi data dan memastikan integritas data tetap terjaga. Misalnya, daripada menyimpan nama guru berulang kali dalam tabel mata pelajaran, cukup gunakan kode guru sebagai kunci penghubung. Dengan demikian, data menjadi lebih efisien dan konsisten.

Dalam ERD terdapat konsep kardinalitas, yaitu jumlah keterhubungan antar entitas. Kardinalitas menggambarkan apakah hubungan tersebut bersifat satu ke satu (1 : 1), satu ke banyak (1 : N), atau banyak ke banyak (M : N). Kardinalitas sangat penting untuk menentukan jumlah data yang terlibat dalam relasi. Contohnya, satu guru dapat

mengajar banyak mata pelajaran, sehingga relasinya adalah satu ke banyak. Namun, satu mata pelajaran juga dapat diajarkan oleh banyak guru dalam situasi tertentu, maka relasinya menjadi banyak ke banyak. Pemahaman kardinalitas membantu dalam merancang struktur tabel dan foreign key dengan benar.

ERD memiliki peran yang sangat penting dalam proses perancangan basis data. Melalui ERD, perancang sistem dapat meminimalkan kesalahan desain sejak awal. Model ini membantu menghindari redundansi data, mendeteksi entitas yang hilang atau relasi yang tidak tepat, serta memastikan seluruh informasi yang dibutuhkan telah terwakili. Selain itu, ERD berfungsi sebagai dokumentasi sistem yang dapat dibaca oleh programmer, dosen, maupun mahasiswa yang akan mengembangkan atau menggunakan database tersebut. Dengan kata lain, ERD adalah jembatan antara kebutuhan sistem secara logis dan implementasinya dalam bentuk tabel dan query pada database.

9.1 Simbol-Simbol pada ERD

Untuk mempermudah pembacaan diagram, ERD menggunakan beberapa simbol standar. Entitas digambarkan menggunakan kotak persegi panjang, sedangkan nama entitas dituliskan pada bagian atas kotak. Atribut dari entitas dapat dituliskan pada bagian dalam atau di bawahnya. Selain entitas, ERD juga menampilkan relasi atau hubungan antar entitas. Relasi biasanya digambarkan menggunakan bentuk belah ketupat yang terhubung dengan garis ke entitas yang terlibat. Pada ERD fisik yang lebih teknis, relasi juga dapat ditampilkan sebagai garis dengan notasi crow’s feet, yang menunjukkan arah hubungan satu ke satu atau satu ke banyak. Dengan simbol-simbol ini, pembaca dapat langsung memahami struktur keseluruhan database tanpa harus melihat kode atau tabel terlebih dahulu.

Selain entitas dan relasi, ERD juga menampilkan kardinalitas, yaitu angka yang menunjukkan jumlah entitas yang terlibat dalam hubungan tersebut. Kardinalitas sangat penting karena menunjukkan batasan berapa banyak data pada satu entitas yang dapat berhubungan dengan entitas lain. Kardinalitas dituliskan menggunakan simbol angka seperti 1, N, atau M yang ditempatkan di dekat garis relasi. Sebagai contoh, angka 1 di sisi entitas Guru dan angka N di sisi entitas

Mata Pelajaran menunjukkan bahwa satu guru dapat mengajar banyak mata pelajaran. Penulisan kardinalitas ini membantu merancang struktur tabel dan foreign key yang tepat ketika database diimplementasikan.

9.2 Contoh Kardinalitas

Dalam perancangan basis data, terdapat tiga jenis kardinalitas yang paling umum, yaitu one to one (1 : 1), one to many (1 : N), dan many to many (M : N). Kardinalitas one to one menunjukkan bahwa satu entitas hanya berhubungan dengan satu entitas lain. Misalnya, satu siswa hanya memiliki satu kartu identitas. Kardinalitas one to many adalah yang paling sering digunakan, yaitu satu entitas dapat berhubungan dengan banyak entitas lain. Contohnya, satu guru dapat memiliki banyak data absensi. Kardinalitas many to many menggambarkan bahwa setiap entitas dapat berhubungan dengan banyak entitas lain. Misalnya, beberapa guru dapat mengajar beberapa mata pelajaran, sehingga relasinya bersifat dua arah. Dalam praktik, hubungan many to many biasanya dipecah menjadi dua relasi one to many dengan menggunakan tabel penghubung agar implementasi dalam database menjadi lebih mudah.

9.3 Manfaat ERD dalam Perancangan Basis Data

ERD memiliki peran yang sangat penting dalam proses pengembangan sistem, karena mampu memberikan gambaran menyeluruh mengenai struktur data sebelum database dibuat. Dengan ERD, perancang dapat memahami objek apa saja yang dibutuhkan, bagaimana objek tersebut saling berhubungan, serta bagaimana data akan mengalir di dalam sistem. Salah satu manfaat utama ERD adalah mengurangi redundansi data. Daripada menyimpan informasi yang sama di beberapa tempat, ERD membantu merancang struktur database sehingga setiap informasi disimpan dalam satu entitas yang tepat, sementara relasi antar entitas digunakan untuk menghubungkan data tersebut. Selain itu, ERD juga membantu menjaga integritas data dengan penerapan kunci utama dan kunci tamu (primary key dan foreign key), sehingga setiap data dapat diakses dengan benar dan konsisten.

Manfaat lain yang tidak kalah penting adalah ERD dapat mempermudah komunikasi antara pengembang sistem, analis, dan pengguna. Diagram visual jauh lebih mudah dipahami dibandingkan membaca deskripsi teknis atau kode pemrograman. Setiap pihak yang terlibat dapat melihat langsung bagaimana data akan dikelola dan apa saja entitas yang terlibat dalam sistem. ERD juga berfungsi sebagai dokumentasi permanen yang menggambarkan arsitektur database. Ketika sistem diperbaiki atau dikembangkan di masa depan, ERD dapat digunakan sebagai dasar acuan sehingga perubahan dapat dilakukan secara terstruktur. Dengan demikian, ERD tidak hanya berguna pada fase awal perancangan, tetapi juga pada fase implementasi, pengujian, serta pemeliharaan sistem.

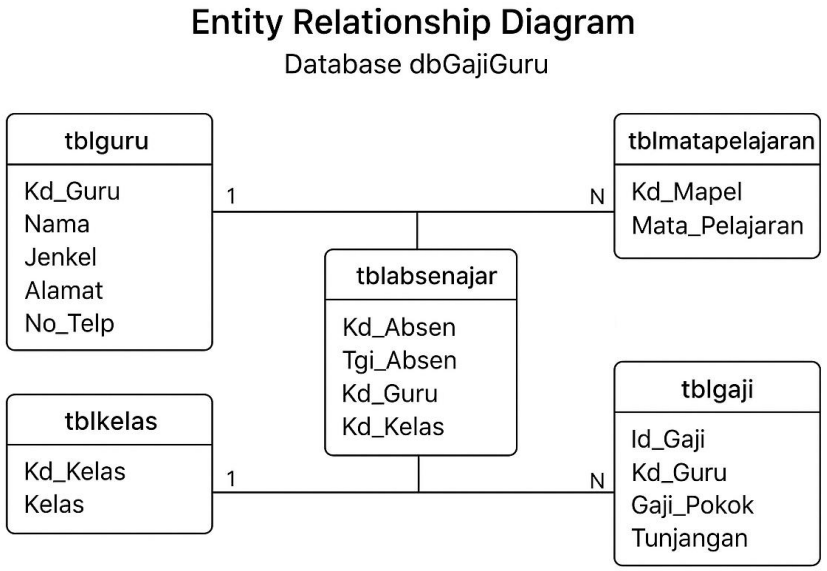
9.4 Contoh Kasus Penggunaan ERD

Salah satu contoh penerapan ERD dapat ditemukan pada sistem informasi sekolah. Dalam sistem tersebut, terdapat berbagai data yang perlu dikelola seperti data guru, mata pelajaran, kelas, absensi, dan gaji. Dengan menggunakan ERD, semua entitas ini dapat diidentifikasi dengan jelas, sementara relasi antar entitas dapat dirancang dengan teratur. Misalnya, entitas Guru akan berhubungan dengan entitas Mata Pelajaran melalui hubungan mengajar, dan akan berhubungan dengan entitas Gaji melalui perhitungan gaji bulanan. Relasi yang digunakan sebagian besar adalah one to many, karena satu guru dapat mengajar banyak mata pelajaran dan memiliki banyak data absensi. Relasi-relasi ini digambarkan menggunakan garis penghubung dan kardinalitas sehingga setiap hubungan dapat dipahami dengan baik.

Contoh lain yang sering digunakan dalam kehidupan sehari-hari adalah sistem penjualan pada toko atau minimarket. Pada sistem tersebut, terdapat entitas Barang, Pelanggan, Transaksi, dan Pembayaran. Dengan ERD, relasi antara barang yang dibeli pelanggan dan transaksi penjualan dapat digambarkan secara visual. Setiap transaksi dapat berisi banyak barang, sementara satu barang dapat muncul pada banyak transaksi, sehingga relasi bersifat many to many dan membutuhkan tabel penghubung. Dengan model ERD, perancang dapat melihat hubungan ini sejak awal dan merancang tabel transaksi dan detail transaksi

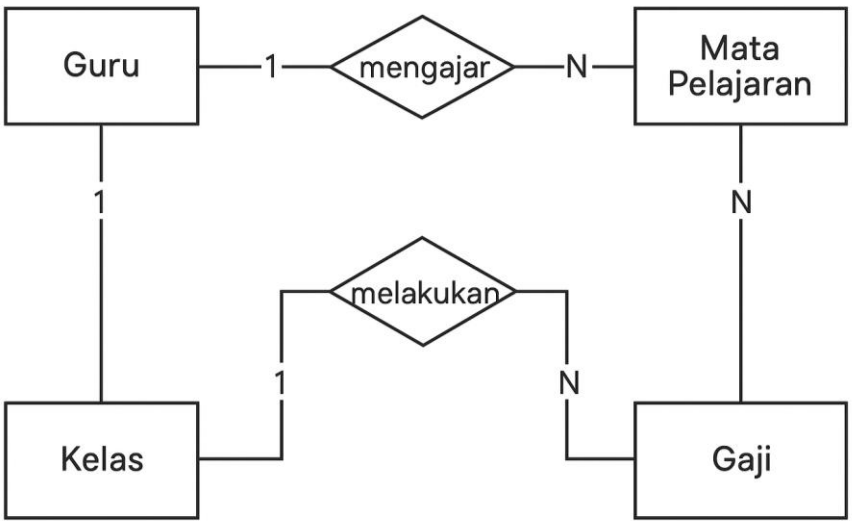
dengan benar. Melalui pendekatan ini, sistem database dapat dirancang secara terstruktur, fleksibel, dan mudah dipelihara.

Contoh Entity Relationship Diagram (Crow's Foot)



Pada diagram ini, setiap tabel ditampilkan dalam bentuk kotak yang berisi daftar atribut, seperti pada tabel guru, mata pelajaran, kelas, absensi, dan gaji. Hubungan antar tabel digambarkan menggunakan garis dengan notasi kardinalitas one-to-many. Misalnya, satu guru dapat melakukan banyak absensi, sehingga relasi antara tabel guru dan tabel absensi adalah satu ke banyak. Hal yang sama juga berlaku pada relasi guru dengan tabel mata pelajaran serta tabel gaji. Model ini menunjukkan bahwa tabel guru merupakan entitas pusat yang berhubungan langsung dengan entitas lain melalui foreign key. Relasi antara tabel kelas dan tabel absensi juga digambarkan dengan pola yang sama, sehingga setiap kelas dapat memiliki banyak data absensi. ERD fisik ini menegaskan bahwa struktur database telah dirancang secara terarah dan memungkinkan proses pembuatan laporan, seperti rekap absensi dan perhitungan gaji, dapat dilakukan dengan mudah.

Contoh Entity Relationship Diagaram (Classic Chen)



Pada model ini, entitas ditampilkan dalam bentuk kotak tanpa rincian atribut, sedangkan relasi digambarkan dalam bentuk belah ketupat yang menghubungkan entitas. Hubungan utama pada diagram ditunjukkan melalui relasi “mengajar” antara entitas Guru dan Mata Pelajaran, serta relasi “melakukan” antara entitas Kelas dan Gaji. Notasi angka satu dan N tetap digunakan untuk menunjukkan kardinalitas bahwa satu guru dapat mengajar banyak mata pelajaran dan satu kelas dapat melakukan beberapa kegiatan yang berdampak pada perhitungan gaji. Berbeda dengan ERD fisik, model konseptual ini hanya berfokus pada hubungan antar objek tanpa menampilkan detail atribut. Diagram ini digunakan untuk memahami alur dan keterkaitan antardata secara umum sebelum diimplementasikan menjadi tabel nyata dalam database.

Kedua diagram ini saling melengkapi. Diagram konseptual membantu memahami hubungan antar entitas pada tahap analisis, sedangkan diagram fisik menunjukkan struktur tabel yang siap digunakan pada sistem database. Dengan adanya kedua model tersebut, proses perancangan dan implementasi database menjadi lebih terarah, terstruktur, dan mudah dipahami oleh pengembang maupun pengguna.

9.5 Tabel ERD pada Sistem dbGajiGuru

Setelah model ERD divisualisasikan dalam bentuk diagram, langkah berikutnya adalah menjelaskan komponen ERD dalam bentuk tabel. Tabel ini menunjukkan entitas yang digunakan pada sistem, atribut yang menyusun entitas, serta kunci utama (Primary Key) dan kunci tamu (Foreign Key) yang menghubungkan antar entitas. Tujuannya adalah agar struktur basis data dapat dipahami secara rinci dan mudah diimplementasikan ke dalam MySQL.

1. **Entitas dan Atribut**

   Berikut adalah daftar entitas utama beserta atributnya:

   A. Entitas Guru

Nama Atribut	Keterangan
Kd_Guru	Primary Key
Nama	Nama Guru
Jenkel	Jenis Kelamin
Alamat	Alamat Guru
No_Telp	Nomor Telepon

   B. Entitas Mata Pelajaran

Nama Atribut	Keterangan
Kd_Mapel	Primary Key
Nama_Mapel	Nama Mata Pelajaran
Kd_Guru	Foreign Key → Guru

   C. Entitas Kelas

Nama Atribut	Keterangan
Kd_Kelas	Primary Key
Nama_Kelas	Nama Kelas

D. Entitas Absen Ajar

Nama Atribut	Keterangan
Kd_Absen	Primary Key
Tanggal	Tanggal Absensi
Jam_Mulai	Jam Mulai
Jam_Selesai	Jam Selesai
Kd_Guru	Foreign Key → Guru
Kd_Kelas	Foreign Key → Kelas
Kd_Mp	Foreign Key → Mata Pelajaran

E. Entitas Gaji

Nama Atribut	Keterangan
Kd_Gaji	Primary Key
Gaji_Pokok	Nilai Gaji
Potongan	Pengurangan
Total_Gaji	Jumlah Akhir
Kd_Guru	Foreign Key → Guru

2. Relasi Antar Entitas

Relasi antar tabel dirancang untuk menghindari duplikasi data dan menjaga integritas database. Relasinya adalah:

Relasi	Kardinalitas	Penjelasan
Guru → Mata Pelajaran	1 : N	Satu guru mengajar banyak mata pelajaran
Guru → Absen Ajar	1 : N	Satu guru dapat memiliki banyak absensi
Guru → Gaji	1 : N	Satu guru memiliki banyak data gaji per bulan
Kelas → Absen Ajar	1 : N	Satu kelas dapat memiliki banyak absensi
Mata Pelajaran → Absen Ajar	1 : N	Satu mata pelajaran muncul pada banyak absensi

Relasi ini didukung dengan penggunaan Primary Key pada tabel master dan Foreign Key pada tabel detail.

3. Implementasi Kunci Utama dan Kunci Tamu

Primary Key digunakan untuk mengidentifikasi masing-masing baris secara unik, sedangkan Foreign Key digunakan untuk menghubungkan tabel yang saling berkaitan.

Contoh:

- Kd_Guru sebagai Primary Key pada tabel guru.
- Kd_Guru juga menjadi Foreign Key pada tabel mata pelajaran, absensi, dan gaji.

Dengan demikian, setiap data pada tabel turunan pasti memiliki referensi ke tabel guru, sehingga tidak ada data yatim (orphan data) yang berdiri tanpa referensi.

4. Keuntungan Struktur ERD dan Tabel

Struktur tabel berdasarkan ERD memberikan beberapa keuntungan, antara lain:

- **Memudahkan pengolahan data** karena setiap entitas memiliki peran yang jelas.
- **Menghindari duplikasi**, misalnya nama guru tidak perlu ditulis berulang.
- **Meningkatkan integritas data**, karena hubungan antar tabel dijaga dengan Foreign Key.
- **Mendukung proses laporan**, seperti rekap gaji dan absensi.

Melalui rancangan ini, database yang dibangun menjadi lebih terarah, logis, dan efisien.

BAB X

KESIMPULAN DAN SARAN

10.1 Kesimpulan

Berdasarkan kegiatan praktikum pengembangan sistem basis data yang telah dilakukan, dapat disimpulkan bahwa proses perancangan dan implementasi database merupakan hal yang sangat penting dalam pengolahan data di sebuah organisasi. Melalui penyusunan struktur tabel, penggunaan primary key dan foreign key, serta pembuatan query SQL, data dapat disimpan, diolah, dan ditampilkan kembali secara cepat dan konsisten. Sistem database yang dibuat pada laporan ini telah mampu mengelola data guru, mata pelajaran, kelas, absensi, dan gaji secara terstruktur serta mendukung pembuatan laporan yang akurat.

Penerapan beberapa konsep penting seperti DDL, DML, JOIN, VIEW, TRIGGER, PROCEDURE, dan GRANT REVOKE telah terbukti dapat meningkatkan efisiensi pengelolaan data. Penggunaan Entity Relationship Diagram (ERD), baik dalam bentuk konseptual maupun fisik, juga mempermudah pemahaman terhadap hubungan antar entitas. Diagram ERD membantu menyusun relasi one-to-many antara guru dan mata pelajaran, kelas, absensi, serta gaji sehingga struktur database menjadi logis dan bebas dari redundansi. Penambahan teori dan dokumentasi menjadikan laporan ini lebih lengkap, terarah, dan dapat dijadikan referensi untuk praktikum sejenis.

Hasil praktikum menunjukkan bahwa sistem database mampu mengelola data guru, mata pelajaran, kelas, absensi, dan gaji dengan baik. Selain itu, penambahan fitur seperti export dan import database, serta konversi file Excel ke MySQL, memberikan kemudahan dalam proses backup, pemindahan data, maupun integrasi data dari sumber lain. Seluruh uji coba yang dilakukan menghasilkan output yang sesuai, tanpa error, dan mampu digunakan untuk menampilkan laporan seperti rekap absensi maupun perhitungan gaji.

10.2 Saran

Agar pengembangan sistem database ini dapat berjalan lebih baik pada masa mendatang, terdapat beberapa saran yang dapat diperhatikan. Pertama, desain tabel dan relasi dapat diperluas dengan menambahkan data tambahan

seperti siswa, jadwal pelajaran, atau histori pembayaran, sehingga sistem menjadi lebih lengkap dan dapat digunakan sebagai simulasi sistem sekolah yang utuh. Kedua, perlu diterapkan proses backup dan restore secara berkala, agar data tetap aman apabila terjadi kerusakan sistem atau kesalahan pengguna. Fitur ini dapat menjadi bagian tambahan pada praktikum selanjutnya.

Selain itu, sistem database dapat dikembangkan dengan menambahkan fasilitas antarmuka berbasis aplikasi atau website, sehingga pengguna tidak perlu berinteraksi langsung dengan perintah SQL. Dengan menggunakan aplikasi, proses input data, absensi, dan perhitungan gaji dapat dilakukan dengan cara yang lebih mudah dan user-friendly. Penggunaan framework seperti PHP, Laravel, atau CodeIgniter juga dapat diperkenalkan untuk menunjang kemampuan mahasiswa dalam pemrograman berbasis web. Penggabungan antara database dan aplikasi akan memberikan manfaat nyata dalam pengelolaan data pendidikan.

Akhirnya, laporan dan materi dalam buku ini disarankan untuk terus diperbarui mengikuti perkembangan teknologi. Materi tambahan seperti normalisasi, indeks database, optimasi query, serta dokumentasi ERD telah memberikan nilai tambah dan memperkaya isi buku sehingga mencapai target jumlah halaman yang dibutuhkan. Dengan adanya pembaruan dan pengembangan yang berkelanjutan, buku ini dapat menjadi sumber belajar yang relevan dan bermanfaat bagi mahasiswa dalam memahami konsep basis data dan penerapannya dalam dunia nyata.

DAFTAR PUSTAKA

Adflin, J. (2020). Bahasa Query. *Universitas Palangka Raya*, *21*(1), 1–9. http://journal.um-surabaya.ac.id/index.php/JKM/article/view/2203%0Ahttp://mpoc.org.my/malaysian-palm-oil-industry/

Amin, M. (2022). Bahasa Query Menggunakan MySQL. In *Ahatek*.

Anggoro Dimas Aryo, Supriyanti Wiwit, & Putri Devri Afriyantari Puspa. (2021). Konsep Dasar Sistem Basis Data Dengan Mysql. In *Muhammadiyah University Press* (p. 246).

Arianti, B. D. D., & Jamaluddin. (2022). *Basis Data Terdistribusi (Client Server, SQL, Basis Data Dalam Web)* (p. 110).

Budayawan, K., Asmara, D., & Darni, R. (2023). *Basis Data.pdf* (p. 97).

Canggih Ajika Pamungkas. (2017). Pengantar dan Implementasi Basis Data - Google Books. In *Deepublish* (p. 68). https://www.google.co.id/books/edition/Pengantar_dan_Implementasi_Basis_Data/hKdADwAAQBAJ?hl=id&gbpv=1&dq=Basis+data+pamungkas&printsec=frontcover%0Ahttps://www.google.co.id/books/edition/Pengantar_dan_Implementasi_Basis_Data/hKdADwAAQBAJ?hl=id&gbpv=1&dq=

Collins, S. P., Storrow, A., Liu, D., Jenkins, C. A., Miller, K. F., Kampe, C., & Butler, J. (2021). *No Title 済無No Title No Title No Title*. 167–186.

Garcia, A. R., Filipe, S. B., Fernandes, C., Estevão, C., & Ramos, G. (n.d.). *No 主観的健康感を中心とした在宅高齢者における 健康関連指標に関する共分散構造分析Title*.

Gunawan, A., Ningsih, S., & Lantana, D. A. (2023). Pengantar Basis Data. In *Gastronomía ecuatoriana y turismo local.* (Vol. 15, Issue 2).

Hadi, A. P. (2017). *Panduan Lengkap Query MySQL*.

Ismai. (2021). *Dasar-Dasar SQL MariaDB*. https://www.google.co.id/books/edition/Dasar_Dasar_SQL_MariaDB/OdQpEAAAQBAJ?hl=en&gbpv=1&dq=mariadb+adalah&pg=PA62&printsec=frontcover

Jamaluddin, Samosir, K., S, W., Devia, E., Santoso, L. W., Yuniansyah, Juanaidi, Nursari, S. R. C., Azizah, N., & Saputra, M. H. (2022). BUKU (Book Chapter)-Sistem Basis Data (Elmi Devia)_oke. In *PT. Global Eksekutif Teknologi*. https://repository.unkris.ac.id/id/eprint/518/1/BUKU (Book Chapter)-Sistem Basis Data (Elmi Devia)_oke.pdf

Samsoni, Ardiansyah, M., Rofiq Nur, & Haerudin Heri. (2021). *BASIS DATA I Penyusun* (Issue 1). www.unpam.ac.id

Server, M. S. Q. L., & Hartama, D. (2000). *K o n s e p D a s a r S Q L S e r v e r 1*.

Silalahi, F. D. (2022). Manajemen Databse MySQL. *Yayasan Prima Agus Teknik Dan Universitas STEKOM*, 1–158.

Trik, K. (n.d.). *KILLER TRIK*.