LAPORAN AKHIR
Mata Kuliah Pemrograman Basis Data
Sistem Basis Data Penjualan Kasir (Point of Sale) pada Warung Makan SE

Dosen Pengampu
Ridwan Dwi Irawan, S.Kom., M.Kom.

Disusun oleh:

Ejra Bagus Alvianto	240103160
Muhammad Hanif Nur Romadhoni	240103167
Muhammad Safiq Wijaya	240103171

UNIVERSITAS DUTA BANGSA SURAKARTA
FAKULTAS ILMU KOMPUTER
2026

KATA PENGANTAR

Puji syukur kehadirat Tuhan Yang Maha Esa atas rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan Laporan Akhir Mata Kuliah Pemrograman Basis Data dengan judul *"Sistem Basis Data Penjualan Kasir (Point of Sale) pada Warung Makan SE"* dengan baik dan tepat waktu.

Laporan ini disusun sebagai bentuk pemenuhan Ujian Akhir Semester pada mata kuliah Pemrograman Basis Data. Tujuan dari penyusunan laporan ini adalah untuk menerapkan konsep-konsep pemrograman basis data yang telah dipelajari, mulai dari perancangan database, normalisasi, hingga implementasi query SQL menggunakan DBMS MySQL. Penulis mengucapkan terima kasih kepada Bapak Ridwan Dwi Irawan, S.Kom., M.Kom. selaku dosen pengampu mata kuliah Pemrograman Basis Data atas bimbingan dan arahan yang telah diberikan. Penulis juga mengucapkan terima kasih kepada semua pihak yang telah membantu dalam penyusunan laporan ini.

Penulis menyadari bahwa laporan ini masih memiliki keterbatasan. Oleh karena itu, penulis mengharapkan kritik dan saran yang membangun demi perbaikan di masa mendatang. Semoga laporan ini dapat memberikan manfaat bagi pembaca.

DAFTAR ISI

BAB I

PENDAHULUAN

1.1.Latar Belakang

Perkembangan teknologi informasi telah mendorong berbagai sektor usaha untuk beralih dari sistem manual ke sistem terkomputerisasi, termasuk pada bidang usaha kuliner. Salah satu aspek penting dalam operasional usaha kuliner adalah pencatatan transaksi penjualan yang dilakukan oleh kasir.

Pada Warung Makan SE, proses pencatatan transaksi masih dilakukan secara manual, sehingga berpotensi menimbulkan kesalahan pencatatan, keterlambatan pelayanan, dan kesulitan dalam pengolahan data penjualan. Oleh karena itu, diperlukan sebuah sistem basis data penjualan kasir yang terstruktur dan terintegrasi.

Sistem basis data terkomputerisasi diharapkan mampu membantu pengelolaan data transaksi secara lebih efisien, akurat, dan konsisten, serta memudahkan proses analisis data penjualan untuk pengambilan keputusan.

1.2.Rumusan Masalah

Berdasarkan latar belakang tersebut, rumusan masalah dalam proyek ini adalah sebagai berikut:

1. Bagaimana merancang basis data penjualan kasir yang terstruktur?
2. Bagaimana melakukan normalisasi data hingga bentuk normal ketiga (3NF)?
3. Bagaimana mengimplementasikan basis data menggunakan MySQL?

1.2.Tujuan

Tujuan dari penyusunan proyek ini adalah:

1. Merancang database sistem Point of Sale (POS).
2. Mengimplementasikan SQL untuk pengelolaan data.
3. Menyediakan data yang konsisten dan terintegrasi.

1.3. Ruang Lingkup dan Batasan

Ruang lingkup proyek ini dibatasi pada:

1. Perancangan dan implementasi basis data penjualan kasir.
2. Tidak membahas antarmuka pengguna (UI)
3. DBMS yang digunakan adalah MySQL.

A.Pembagian Jobdesk

1 Ejra Bagus Alvianto : Bertugas mencari studi kasus, membuat poster, membuat Normalisasi 1NF - 3NF, membuat ERD

2 Muhammad Hanif Nur : Romadhoni Bertugas membuat laporan projek, mebuat database serta kode sql

3 Muhammad Safiq Wijaya :Bertugas mencari studi kasus, membuat repository di GitHub, membuat Normalisasi 1NF - 3NF, membuat ERD

BAB II

LANDASAN TEORI

2.1.MySQL

MySQL adalah sebuah perangkat lunak system manajemen basis data SQL (DBMS)yang multithread, dan multi-user. MySQL adalah implementasi dari system manajemen4 basisdata relasional (RDBMS). MySQL dibuah oleh TcX dan telah diperc aya mengelolasystem dengan 40 buah database berisi 10.000 tabel dan 500 di antaranya memiliki 7 juta baris.MySQL AB merupakan perusahaan komersial Swedia yang mensponsori dan yangmemiliki MySQL. Pendiri MySQL AB adalah dua orang Swedia yang bernama David Axmark,

Allan Larsson dan satu orang Finlandia bernama Michael “Monty”. Setiap pengguna MySQL dapat menggunakannya secara bebas yang didistribusikan gratis dibawah lisensi GPL(GeneralPublic License) namun tidak boleh menjadikan produk turunan yang bersifat komersial.Pada saat ini MySQL merupakan database server yang sangat terkenal di dunia, semuaitu tak lain karena bahasa dasar yang digunakan untuk mengakses database yaitu SQL. SQL(Structured Query Language) pertama kali diterapkan pada sebuah proyek riset padalaboratorium riset San Jose, IBM yang bernama system R. Kemudian SQL juga dikembanganoleh Oracle, Informix dan Sybase. Dengan menggunakan SQL, proses pengaksesan databaselebih user-friendly dibandingan dengan yang lain, misalnya dBase atau Clipper karena merekamasih menggunakan perintah-perintah pemrograman murni.SQL dapat digunakan secara berdiri sendiri maupun di lekatkan pada bahasa pemograman seperti C

2.2.Basis data

Basis data (bahasa Inggris: *database*) adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut. Perangkat lunak yang digunakan untuk mengelola dan memanggil kueri (*query*) basis data disebut Sistem Manajemen Basis Data (*Database Management System* atau DBMS).

Sistem basis data dipelajari dalam ilmu informasi. Meskipun istilah "basis data" kini memiliki makna yang semakin luas hingga ke luar bidang elektronika, secara spesifik istilah ini merujuk pada basis data komputer.

Sejarah dan Konsep Dasar

Catatan yang mirip dengan basis data sebenarnya sudah ada sebelum revolusi industri, yaitu dalam bentuk buku besar, kuitansi, dan kumpulan data yang berhubungan dengan bisnis.

Konsep dasar dari basis data adalah kumpulan dari catatan-catatan atau potongan pengetahuan. Sebuah basis data memiliki penjelasan terstruktur dari jenis fakta yang tersimpan di dalamnya yang disebut dengan skema. Skema menggambarkan objek yang diwakili suatu basis data serta hubungan di antara objek-objek tersebut.

2.3.Entity Relationship Diagram (ERD)

Entity Relationship Diagram (ERD) adalah suatu diagram konseptual yang digunakan untuk memodelkan struktur basis data dengan menggambarkan hubungan antara entitas-entitas yang terlibat di dalam suatu sistem. ERD berfungsi sebagai alat bantu perancangan basis data sebelum sistem diimplementasikan ke dalam Database Management System (DBMS).

Dalam ERD, entitas merepresentasikan objek atau konsep yang datanya perlu disimpan, seperti pelanggan, barang, atau transaksi. Setiap entitas memiliki atribut yang menjelaskan karakteristik dari entitas tersebut, misalnya nama, kode, atau tanggal. Selain itu, ERD juga menampilkan relasi yang menunjukkan hubungan antar entitas, seperti hubungan satu ke banyak (one-to-many) atau banyak ke banyak (many-to-many).

ERD sangat penting dalam perancangan basis data karena membantu perancang dalam memahami struktur data secara menyeluruh, menghindari redundansi data, serta memastikan integritas dan konsistensi data. Dengan adanya ERD, proses implementasi basis data menjadi lebih terarah dan sistematis karena struktur tabel, primary key, dan foreign key telah direncanakan sejak awal.

2.4.Normalisasi Basis Data

Normalisasi basis data adalah proses pengorganisasian data ke dalam tabel-tabel yang lebih kecil dan terstruktur dengan tujuan untuk mengurangi redundansi data dan meningkatkan integritas data. Normalisasi dilakukan melalui beberapa tahap yang disebut sebagai bentuk normal (normal form), dimulai dari bentuk normal pertama hingga bentuk normal lanjutan.

2.4.1 First Normal Form (1NF)

Bentuk normal pertama (1NF) mensyaratkan bahwa setiap atribut dalam tabel harus memiliki nilai yang bersifat atomik atau tidak dapat dibagi lagi. Selain itu, tidak boleh terdapat kelompok data berulang dalam satu kolom. Penerapan 1NF bertujuan untuk memastikan bahwa struktur tabel sederhana dan mudah dikelola.

2.4.2 Second Normal Form (2NF)

Bentuk normal kedua (2NF) mengharuskan tabel telah memenuhi 1NF dan setiap atribut non-kunci harus bergantung sepenuhnya pada primary key. Pada tahap ini, ketergantungan parsial dihilangkan, khususnya pada tabel yang memiliki primary key gabungan (composite key). Dengan 2NF, data menjadi lebih terstruktur dan redundansi semakin berkurang.

2.4.3 Third Normal Form (3NF)

Bentuk normal ketiga (3NF) mensyaratkan bahwa tabel telah memenuhi 2NF dan tidak terdapat ketergantungan transitif, yaitu atribut non-kunci tidak bergantung pada atribut non-kunci lainnya. Penerapan 3NF bertujuan untuk memisahkan data ke dalam tabel yang benar-benar independen sehingga integritas dan konsistensi data dapat terjaga dengan baik.

Tujuan dan Manfaat Normalisasi

Tujuan utama normalisasi basis data adalah:

a) Mengurangi duplikasi atau redundansi data
b) Menghindari anomali data (insert, update, delete anomaly)
c) Meningkatkan efisiensi penyimpanan data

d) Menjaga integritas dan konsistensi basis data

Dengan menerapkan ERD dan normalisasi secara tepat, basis data yang dirancang akan menjadi lebih efisien, terstruktur, dan mudah dikembangkan di masa depan.

2.5 Structured Query Language (SQL)

Structured Query Language (SQL) adalah bahasa standar yang digunakan untuk mengelola dan memanipulasi basis data relasional. SQL memungkinkan pengguna untuk melakukan berbagai operasi terhadap data, mulai dari pembuatan struktur basis data, pengelolaan data, hingga pengambilan informasi yang tersimpan di dalam basis data secara efisien dan terstruktur. SQL digunakan oleh hampir seluruh Database Management System (DBMS) relasional, seperti MySQL, PostgreSQL, Oracle, dan SQL Server.

Dalam penggunaannya, SQL terbagi ke dalam beberapa kategori utama, yaitu Data Definition Language (DDL)**,** Data Manipulation Language (DML)**,** Transaction Control Language (TCL)**,** serta query lanjutan seperti JOIN**,** agregasi, dan subquery**.**

2.5.1 Data Definition Language (DDL)

Data Definition Language (DDL) merupakan bagian dari SQL yang digunakan untuk mendefinisikan dan mengatur struktur basis data. Perintah DDL berfungsi untuk membuat, mengubah, dan menghapus objek basis data seperti database, tabel, dan constraint. Contoh perintah DDL antara lain CREATE, ALTER, dan DROP. Dengan DDL, perancang basis data dapat menentukan struktur tabel, tipe data, serta aturan integritas seperti primary key dan foreign key.

2.5.2 Data Manipulation Language (DML)

Manipulation Language (DML) digunakan untuk mengelola isi data yang terdapat di dalam tabel. Perintah DML memungkinkan pengguna untuk menambahkan, mengubah, menghapus, dan menampilkan data. Contoh perintah DML meliputi INSERT, UPDATE, DELETE, dan SELECT. DML sangat penting dalam operasional sistem karena berhubungan langsung dengan proses pengolahan data sehari-hari.

2.5.3 Transaction Control Language (TCL)

Transaction Control Language (TCL) digunakan untuk mengendalikan transaksi dalam basis data agar integritas data tetap terjaga. Perintah TCL memastikan bahwa sekumpulan operasi DML dapat dijalankan secara utuh atau dibatalkan jika terjadi kesalahan. Perintah yang termasuk dalam TCL antara lain START TRANSACTION, COMMIT, dan ROLLBACK. Dengan TCL, konsistensi dan keandalan data dapat dipertahankan terutama pada sistem yang melibatkan banyak transaksi.

2.5.4 Query JOIN

JOIN merupakan teknik dalam SQL yang digunakan untuk menggabungkan data dari dua atau lebih tabel berdasarkan hubungan tertentu, biasanya melalui primary key dan foreign key. JOIN memungkinkan pengguna memperoleh informasi yang lebih kompleks dan bermakna dari basis data yang terpisah. Beberapa jenis JOIN yang umum digunakan antara lain INNER JOIN, LEFT JOIN, dan RIGHT JOIN.

2.5.5 Query Agregasi

Query agregasi digunakan untuk melakukan perhitungan terhadap sekumpulan data dalam sebuah tabel. Fungsi agregasi yang sering digunakan antara lain SUM, COUNT, AVG, MAX, dan MIN. Query agregasi biasanya dikombinasikan dengan klausa GROUP BY untuk mengelompokkan data berdasarkan kriteria tertentu, serta HAVING untuk memberikan kondisi pada hasil agregasi.

2.5.6 Subquery

Subquery adalah query yang ditulis di dalam query lain. Subquery digunakan untuk menghasilkan data sementara yang akan diproses lebih lanjut oleh query utama. Subquery dapat ditempatkan pada klausa SELECT, FROM, atau WHERE, dan sangat berguna untuk menyelesaikan permasalahan yang membutuhkan perhitungan atau penyaringan data secara bertingkat.

2.5.7 DDL (Data Definition Language)

DDL (Data Definition Language) DDL, atau Data Definition Language, adalah jenis perintah SQL yang digunakan untuk mendefinisikan, mengelola, dan mengubah struktur objek database. Objek-objek ini mencakup tabel, indeks, tampilan, dan sebagainya. Salah satu perintah DDL yang umum digunakan adalah CREATE, yang

digunakan untuk membuat objek database baru seperti tabel atau indeks. DDL juga mencakup perintah ALTER untuk mengubah struktur objek yang sudah ada, serta perintah DROP untuk menghapus objek dari database. Perintah DDL berperan penting dalam merancang dan memelihara skema database. Dengan menggunakan DDL, administrator database dapat membuat dan mengelola entitas-entitas yang membentuk struktur dasar database. Sebagai contoh, jika suatu organisasi ingin menambahkan kolom baru ke dalam tabel database untuk mencerminkan perubahan kebutuhan bisnis, mereka dapat menggunakan perintah DDL seperti ALTER TABLE untuk mengubah struktur tabel tersebut tanpa menghapus data yang sudah ada. Sebagai inti dari proses perancangan dan pemeliharaan database, pemahaman yang baik tentang DDL sangatlah penting dalam pengelolaan sistem database.

2.6 DML (Data Manipulation Language)

Data Manipulation Language (DML) adalah bagian dari bahasa SQL yang digunakan untuk memanipulasi data yang ada dalam database. Fokus utamanya adalah pada operasi operasi seperti penyisipan data baru, pembaruan data yang sudah ada, dan penghapusan data dari tabel. Perintah-perintah DML mendasarkan operasinya pada struktur yang telah ditetapkan menggunakan Data Definition Language (DDL), seperti membuat tabel atau mengubah skema. Dalam penggunaan sehari-hari, DML memberikan kemampuan kepada pengguna atau aplikasi untuk berinteraksi dengan data dalam database. Misalnya, dengan menggunakan perintah INSERT, pengguna dapat menambahkan baris baru ke dalam tabel. Perintah UPDATE memungkinkan untuk memperbarui nilai-nilai yang sudah ada, sementara DELETE digunakan untuk menghapus data yang tidak diperlukan. Dengan demikian, DML adalah bagian kunci dari SQL yang memungkinkan pengguna mengelola dan mengubah data dalam sistem basis data.

BAB III

PERANCANGAN DAN IMPLEMENTASI

3.1 Analisis Kebutuhan Sistem

Skenario Sistem

Studi kasus ini mengambil proses bisnis di "Warung Makan SE", sebuah usaha kuliner yang melayani penjualan makanan dan minuman. Skenario sistem yang ingin dibuat adalah "Sistem Basis Data Penjualan Kasir (Point of Sale)".

Sistem ini bertujuan untuk mengelola dan mencatat setiap transaksi penjualan yang terjadi di warung makan tersebut.

Alur skenario adalah sebagai berikut:

1. Seorang Pelanggan (misalnya 'fatin') datang dan memilih jenis layanan (misalnya 'free table' atau Dine-in).
2. Kasir (Admin) mengidentifikasi pelanggan dan membuka Transaksi baru.
3. Pelanggan memesan beberapa Barang (item menu).
4. Kasir memasukkan setiap item pesanan ke dalam sistem, yang akan dicatat dalam Detail Transaksi.
5. Sistem menghitung total pembayaran.

Setelah pembayaran (misalnya via 'Transfer'), sistem akan menghasilkan Nota sebagai bukti transaksi.Data dari nota inilah yang akan dijadikan dasar (Tabel Umum/Unnormalized) untuk proses normalisasi dan perancangan basis data. Data Transaksi (Contoh Nota) Berikut adalah contoh data transaksi yang diambil dari satu nota penjualan di Warung Makan SE, yang menjadi dasar studi kasus ini:

WARUNG SE

No Nota : CY23/501018/0090
Waktu : 13 Okt 25 12:25
Kasir : Yut
Jenis Order : Free Table
Nama Order : Fatin

1 NASI AYAM GEPREK + Lv 2	9,000
1 ES TEH JUMBO	2,000
1 Nasi Ayam Geprek Sambal Ijo	1.200
1 ES JERUK	3,000
2 Tahu Goreng	1,000
2 TEMPE GORENG	1,000
Subtotal 8 Produk	28,000
Total	28.000
Transfer	28,000
Total Bayar	28,000

Terimakasih :)

insta:warung.se

3.2 Tabel Umum (Unnormalized Table / Tabel Awal)

No Nota	Waktu	Jenis_order	Nama_order	Jumlah_Barang	Nama_Barang	Harga_Barang	Total_Bayar	Transfer
CS/27/251018/0090	10/18/2025	free table	fatin	1	Nasi Ayam Geprek +1 lv 2	9000	28000	28000
				1	Es Teh Jumbo	2000		
				1	nasi Ayam Sambal Ijo	12000		
				1	Es Jeruk	3000		
				2	Tahu Goreng	1000		
				2	Tempe Goreng	1000		

Tabel ini adalah Tabel Umum (Unnormalized Table / Tabel Awal) yang mewakili data mentah dari satu nota transaksi penjualan di "Warung Makan SE." Tabel ini mencakup detail kepala transaksi seperti No Nota (CS/27/251018/0090), Waktu (10/18/2025), jenis layanan (Jenis_order: free

table), dan nama pemesan (Nama_order: fatin). Selain itu, tabel ini juga mencatat ringkasan pembayaran (Total_Bayar dan Tabel ini adalah Tabel Umum (Unnormalized Table / Tabel Awal) yang mewakili data mentah dari satu nota transaksi penjualan di "Warung Makan SE." Tabel ini mencakup detail kepala transaksi seperti No Nota (CS/27/251018/0090), Waktu (10/18/2025), jenis layanan (Jenis_order: free table), dan nama pemesan (Nama_order: fatin). Selain itu, tabel ini juga mencatat ringkasan pembayaran (Total_Bayar dan Transfer), yang dalam contoh ini adalah Rp 28.000. Tabel awal ini bersifat *unnormalized* karena tujuannya adalah untuk mengumpulkan semua data, menjadikannya dasar untuk proses normalisasi basis data.

3.3 Normalisasi 1NF (First Normal Form)

Tabel_Normalisasi_Pertama(1NF)

Id_barang	No_Nota	Waktu	Jenis_order	Nama_order	Nama_Barang	Jumlah_Barang	Harga_Barang	Total_Bayar	Transfer
BRG01	CS/27/251018/0090	18/10/2025	free table	fatin	Nasi Ayam Geprek +1 lv 2	1	9000	28000	28000
BRG02	CS/27/251018/0090	18/10/2025	free table	fatin	Es Teh Jumbo	1	2000	28000	28000
BRG03	CS/27/251018/0090	18/10/2025	free table	fatin	Nasi Ayam Sambal ijo	1	12000	28000	28000
BRG04	CS/27/251018/0090	18/10/2025	free table	fatin	Es Jeruk	1	3000	28000	28000
BRG05	CS/27/251018/0090	18/10/2025	free table	fatin	Tahu Goreng	2	1000	28000	28000
BRG06	CS/27/251018/0090	18/10/2025	free table	fatin	Tempe Goreng	2	1000	28000	28000

B. 3NF: Menghilangkan ketergantungan transitif

Tabel ini menangkap informasi lengkap mengenai sebuah transaksi, mulai dari waktu, jenis layanan, nama pelanggan, hingga detail setiap barang yang dipesan beserta jumlah dan harganya. Namun, dapat diamati adanya redundansi data atau pengulangan informasi pada beberapa kolom. Sebagai contoh, data seperti No_Nota, Waktu, Jenis_Order, Nama_Order, Total_Bayar, dan Transfer bernilai sama untuk semua baris. Redundansi ini mengindikasikan bahwa tabel ini masih rentan terhadap anomali data dan memerlukan normalisasi lebih lanjut ke bentuk Normal Kedua (2NF) dan Normal

Ketiga (3NF) untuk memisahkan data transaksi, data pelanggan, dan data detail item

menjadi tabel-tabel yang terpisah, sehingga efisiensi penyimpanan dan integritas data dapat lebih terjaga.

3.4 Normalisasi 2NF (Second Normal Form)

A.Tabel Barang

Id_barang (PK)	Nama_Barang	Harga_Barang
BRG01	Nasi Ayam Geprek +1 lv 2	9000
BRG02	Es Teh Jumbo	2000
BRG03	Nasi Ayam Sambal ijo	12000
BRG04	Es Jeruk	3000
BRG05	Tahu Goreng	1000
BRG06	Tempe Goreng	1000

Tabel Barang di atas merupakan hasil normalisasi bentuk kedua (2NF) yang berfungsi untuk menyimpan data master atau referensi dari setiap item menu yang dijual. Dalam tabel ini, Id_Barang bertindak sebagai Primary Key (PK) yang mengidentifikasi setiap barang secara unik, sementara atribut Nama_Barang dan Harga_Barang sepenuhnya bergantung pada kunci utama ini, sehingga telah memenuhi syarat 2NF dengan menghilangkan ketergantungan parsial. Pemisahan data barang ke dalam tabel tersendiri ini menghilangkan redundansi yang terjadi pada tabel sebelumnya, seperti pengulangan nama dan harga barang untuk setiap transaksi, sehingga meningkatkan efisiensi penyimpanan dan memudahkan dalam maintenance data, misalnya ketika perlu mengubah harga suatu barang, perubahan hanya dilakukan sekali pada tabel ini.

B.Tabel Detail Transaksi

No_Nota (PK, FK)	Id_barang (PK, FK)	Jumlah
CS/27/251018/0090	BRG01	1
CS/27/251018/0090	BRG02	1
CS/27/251018/0090	BRG03	1
CS/27/251018/0090	BRG04	1
CS/27/251018/0090	BRG05	2
CS/27/251018/0090	BRG06	2

Tabel Detail Transaksi di atas merupakan hasil penerapan normalisasi bentuk kedua (2NF) yang berfungsi sebagai tabel penghubung untuk merekam

setiap item barang yang dipesan dalam suatu transaksi. Tabel ini memiliki kunci utama komposit (Composite Primary Key) yang terdiri dari No_Nota dan Id_Barang, dimana keduanya juga berperan sebagai Foreign Key (FK) yang merujuk ke tabel transaksi induk dan tabel master barang. Struktur ini memastikan bahwa untuk setiap transaksi (yang diidentifikasi oleh No_Nota), dapat tercatat multiple barang (yang diidentifikasi oleh Id_Barang) beserta Jumlah pemesanannya, sehingga hubungan many-to-many antara entitas transaksi dan barang dapat terpecahkan dengan benar.

Dengan memisahkan detail item pesanan ke dalam tabel tersendiri, redundansi data yang signifikan telah berhasil dihilangkan. Informasi seperti nama barang dan harga barang tidak diulang di setiap transaksi, melainkan hanya disimpan sekali di Tabel Barang, sementara tabel ini hanya menyimpan kunci penjabaran dan kuantitasnya. Hal ini tidak hanya mengoptimalkan penyimpanan tetapi juga menjaga konsistensi dan integritas data, karena setiap entri di tabel ini sepenuhnya bergantung pada seluruh kunci utama kompositnya, yang merupakan syarat terpenuhinya 2NF.

3.4.1 Normalisasi 3NF (Third Normal Form)

A.Tabel Pelanggan

Tabel_Pelanggan(3NF)

ID_PELANGGAN (PK)	Nama_pelanggan	Jenis_Pelanggan
PLG01	fatin	freetable

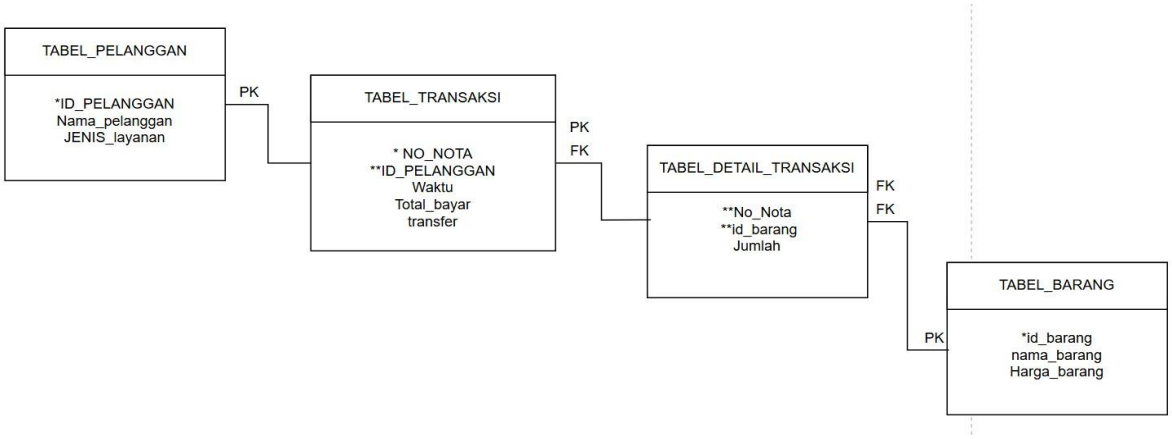
Tabel Pelanggan (3NF) berfungsi untuk menyimpan data identitas pelanggan yang melakukan transaksi di warung makan. Tabel ini terdiri dari tiga kolom utama, yaitu ID_PELANGGAN sebagai primary key yang menjadi kode unik setiap pelanggan, Nama_pelanggan yang menyimpan nama pelanggan, dan Jenis_Pelanggan yang menunjukkan jenis layanan seperti freetable atau take away. Data dalam tabel ini sudah berada pada bentuk normal ketiga (3NF), sehingga setiap atribut hanya bergantung pada kunci utama tanpa redundansi data. Tabel ini juga memiliki hubungan dengan tabel Transaksi melalui kolom ID_PELANGGAN sebagai foreign key, yang memungkinkan satu pelanggan memiliki banyak transaksi.

B.Tabel Transaksi

Tabel_Transaksi(3NF)				
No_Nota (PK)	Waktu	Total_Bayar	Transfer	ID_PELANGGAN (FK)
CS/27/251018/0090	18/10/2025	28000	28000	PLG01

Tabel Transaksi menunjukkan bahwa jumlah tersebut ditransfer, dan nilai di kolom tersebut sama dengan "Total_Bayar", mengindikasikan bahwa pembayaran dilakukan secara penuh sesuai dengan total yang harus dibayar. Kode "PLG01" k merujuk pada identifikasi pelanggan atau kode transaksi tertentu. Secara keseluruhan, tabel ini merepresentasikan sebuah transaksi transfer yang telah diselesaikan dengan nominal kecil, mungkin untuk pembayaran barang, jasa, atau tagihan tertentu.

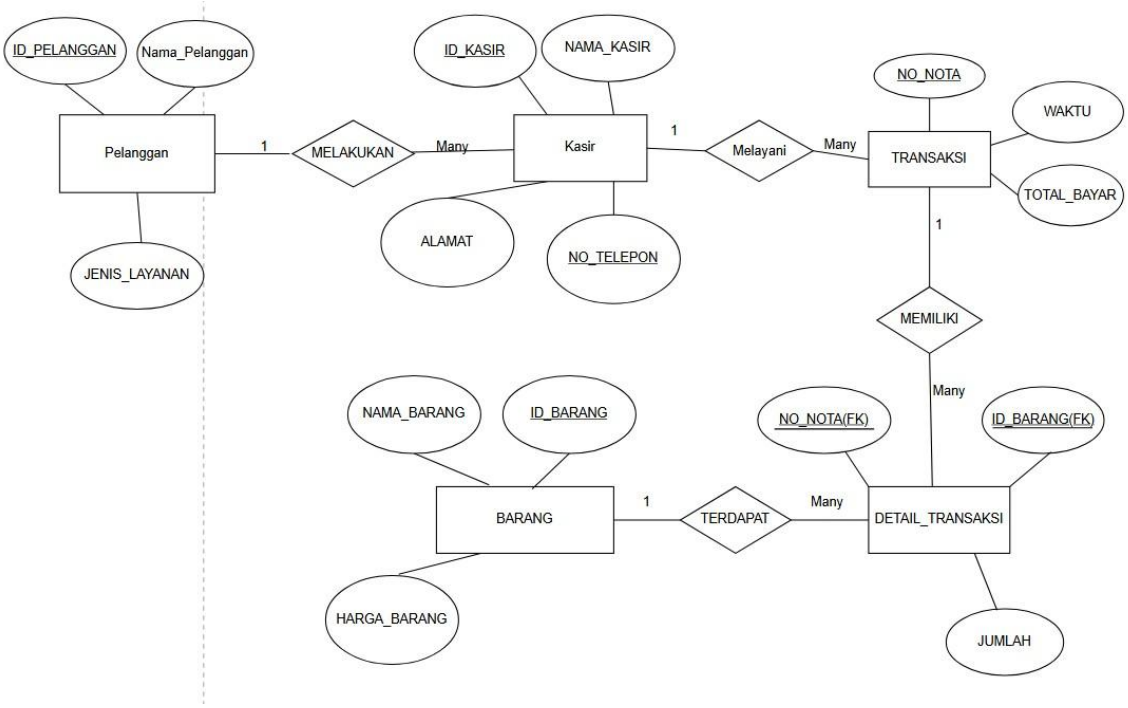
3.5 Konsep Relasi Tabel



Berdasarkan diagram relasi yang ditampilkan, dapat dijelaskan bahwa struktur basis data ini terdiri dari empat tabel utama yang saling terhubung untuk mengelola operasional sebuah warung. Tabel TABEL_PELANGGAN dan TABEL_BARANG berperan sebagai tabel master yang menyimpan data dasar, dimana TABEL_TRANSAKSI (kemungkinan dimaksudkan sebagai TABEL_TRANSAKSI, bukan TRANSMCSI) berfungsi sebagai tabel transaksi inti yang mencatat setiap penjualan dan terhubung ke TABEL_PELANGGAN melalui ID_PELANGGAN. Sementara itu, TABEL_DETAIL_TRANSAKSI berperan sebagai tabel penghubung yang menjembatani hubungan many-to-many antara TABEL_TRANSAKSI dan TABEL_BARANG dengan menggunakan composite primary key dari No_Nota dan id_barang, sehingga memungkinkan pencatatan detail item yang dibeli dalam setiap transaksi. Desain relasi ini memastikan integritas data,

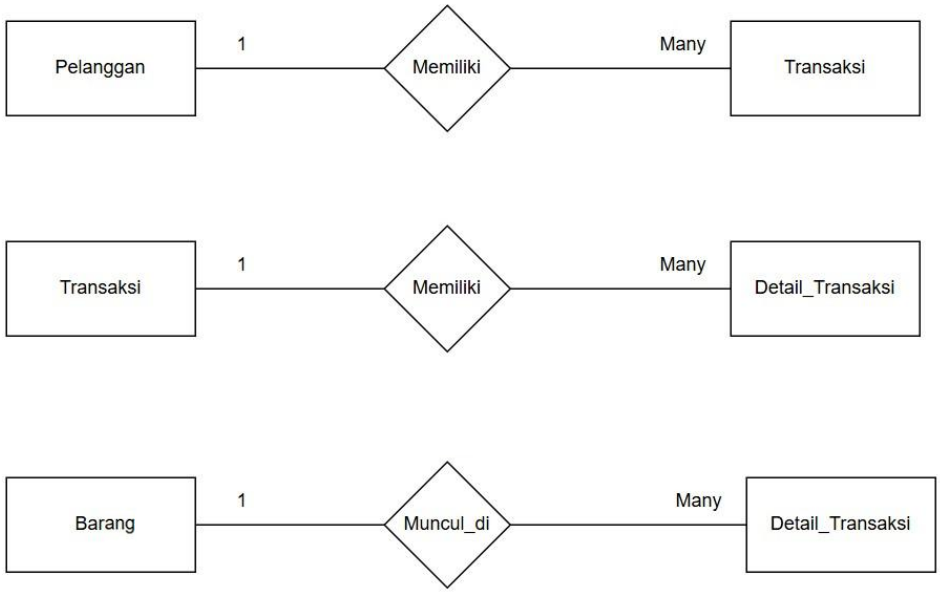
menghindari redundansi informasi, dan mendukung pelacakan lengkap mulai dari identitas pelanggan, waktu transaksi, hingga detail barang yang terjual.

3.6 Entity Relationship Diagram (ERD)



Berdasarkan Entity Relationship Diagram (ERD) yang ditampilkan, dapat dijelaskan bahwa diagram ini menggambarkan hubungan logis antara empat entitas utama dalam sistem database warung, yaitu Pelanggan, Kasir, Transaksi, dan Barang. Entitas Pelanggan dan Kasir terhubung ke entitas Transaksi dalam hubungan one-tomany, yang berarti satu pelanggan atau satu kasir dapat melakukan banyak transaksi. Selanjutnya, entitas Transaksi dan Barang dihubungkan oleh entitas DETAIL_TRANSAKSI dalam hubungan many-to-many, yang berarti satu transaksi dapat mencakup banyak barang, dan satu barang dapat muncul di banyak transaksi. Entitas penghubung ini mencatat JUMLAH pembelian untuk setiap barang, sehingga melengkapi alur data dari pelanggan memesan, kasir memproses, hingga detail barang yang terjual tercatat dengan rapi. Desain ERD ini memastikan integritas referensial dan menjadi blueprint untuk membangun basis data yang terstruktur dan efisien.

3.7 Derajat Kardinalitas



Derajat Kardinalitas ERD Warung Makan:

1. Pelanggan — Transaksi

   Kardinalitas: 1 : N

   A).Satu pelanggan dapat memiliki banyak transaksi.

   B).Setiap transaksi terkait satu pelanggan.

2. Transaksi — Detail_Transaksi

   Kardinalitas: 1 : N

   A).Satu transaksi memiliki banyak detail transaksi.

   B).Setiap detail terkait satu transaksi.

3. Barang — Detail_Transaksi

   Kardinalitas: 1 : N

   A).Satu barang dapat muncul di banyak detail transaksi.

   B).Setiap detail hanya berisi satu barang.

Ringkasan Kardinalitas

Entitas 1	Relasi	Entitas 2	Kardinalitas
Pelanggan	memiliki	Transaksi	1 : N
Transaksi	memiliki	Detail_Transaksi	1 : N
Barang	muncul di	Detail_Transaksi	1 : N

3.8 Kamus basis data

3.8.1 Kamus Basis Data Tabel Pelanggan

Basis Kamus Data Nama Database	Data_Warung
Nama Tabel	Tabel_Pelanggan
Fungsi Tabel	Menyimpan_Data_Pe;anggan

Nama Field	Tipe Data	Panjang Karakter	Keterangan
ID_PELANGGAN	CHAR	5	PRIMARY KEY
Nama_pelanggan	VARCHAR	50	Nama pemesan/pelanggan
JENIS_LAYANAN	VARCHAR	20	Jenis layanan (freetable, take away)

Tabel ini merupakan struktur basis data yang mendefinisikan skema untuk menyimpan informasi pelanggan pada sebuah sistem warung. Tabel dengan nama Tabel_Pelanggan dalam database Data_Warung dirancang untuk mencatat data-data utama pelanggan, yang terdiri dari tiga bidang utama: ID_PELANGGAN sebagai kunci unik (PRIMARY KEY) untuk mengidentifikasi setiap pelanggan secara spesifik, Nama_pelanggan untuk mencatat nama individu yang memesan, dan
JENIS_LAYANAN untuk membedakan metode layanan yang dipilih, seperti "ditempat" (freetable) atau "bawa pulang" (take away). Struktur ini memastikan bahwa data pelanggan dapat disimpan, dielola, dan diambil dengan efisien, mendukung operasional warung dengan membedakan preferensi layanan setiap pelanggan.

3.8.2 Kamus Basis Data Tabel Barang

Kamus Basis Data	
Nama Database	Data_Warung
Nama Tabel	Tabel_Barang
Fungsi Tabel	Menyimpa_Data_Tabel_Barang

Nama Field	Tipe Data	Panjang Karakter	Keterangan
Id_barang	CHAR	5	PRIMARY KEY
Nama_Barang	VARCHAR	100	Nama item/menu makanan
Harga_Barang	INT	-	Harga jual barang

Tabel ini merupakan struktur basis data yang mendefinisikan skema untuk menyimpan informasi barang atau menu pada sebuah warung. Tabel dengan nama Tabel_Barang dalam database Data_Warung dirancang untuk mencatat data inventaris atau menu yang dijual, yang terdiri dari tiga bidang utama: id_barang sebagai kunci unik (PRIMARY KEY) yang mengidentifikasi setiap barang dengan kode tertentu, Nama_Barang untuk mencatat nama item atau menu makanan, serta Harga_Barang yang menyimpan nilai harga dalam tipe data integer. Struktur ini memastikan bahwa data produk dapat dikelola dengan terstruktur, memudahkan proses pencatatan stok, penjualan, dan pengambilan laporan keuangan berdasarkan item yang tersedia.

3.8.2 Kamus Basis Data Tabel Transaksi

Kamus Basis Data	
Nama Database	Data_Warung
Nama Tabel	Tabel_Transaksi
Fungsi Tabel	Menyimpan_Data_Transaksi

Nama Field	Tipe Data	Panjang Karakter	Keterangan
No_Nota	VARCHAR	17	PRIMARY KEY
Waktu	DATE/DATETIME	-	Tanggal dan waktu transaksi
Total_Bayar	INT	-	Total uang yang dibayarkan
Transfer	INT	-	Jumlah yang dibayar via transfer
ID_PELANGGAN	CHAR	5	FOREIGN KEY ke Tabel Pelanggan

Tabel ini merupakan struktur basis data yang dirancang untuk merekam seluruh transaksi yang terjadi pada sebuah warung. Tabel dengan nama Tabel_Transaksi dalam database Data_Warung berfungsi sebagai pusat pencatatan penjualan, dengan No_Nota bertindak sebagai kunci unik (PRIMARY KEY) yang mengidentifikasi setiap transaksi secara individual. Tabel ini mencatat informasi penting seperti Waktu terjadinya transaksi, Total_Bayar yang harus dibayar customer, dan Transfer yang mencatat bagian dari pembayaran yang dilakukan secara non-tunai. Keunggulan desain tabel ini terletak pada penggunaan ID_PELANGGAN sebagai FOREIGN KEY yang menghubungkan setiap transaksi ke Tabel_Pelanggan, sehingga memungkinkan pelacakan riwayat pembelian dan preferensi setiap pelanggan,

yang sangat berguna untuk analisis bisnis dan manajemen hubungan pelanggan.

3.8.3 Kamus Basis Data Tabel Detail Transaksi

Kamus Basis Data	
Nama Database	Data_Warung
Nama Tabel	Tabel_Detail_Transaksi
Fungsi Tabel	Menyimpan_Data_Transaksi

Nama Field	Tipe Data	Panjang Karakter	Keterangan
No_Nota	VARCHAR	17	PRIMARY KEY, FOREIGN KEY ke Tabel Transaksi
Id_barang	CHAR	5	PRIMARY KEY, FOREIGN KEY ke Tabel Barang
Jumlah	INT	-	Jumlah item yang dibeli

Tabel ini merupakan struktur basis data yang berfungsi untuk mencatat detail item dari setiap transaksi yang terjadi, yang dalam desain basis data dikenal sebagai tabel bridge atau tabel penghubung. Tabel dengan nama Tabel_Detail_Transaksi ini menghubungkan dua entitas utama, yaitu Tabel_Transaksi dan Tabel_Barang, dengan menggunakan kombinasi No_Nota dan Id_barang sebagai COMPOSITE PRIMARY KEY. Kolom No_Nota juga berperan sebagai

FOREIGN KEY yang merujuk ke transaksi induk, sementara Id_barang adalah FOREIGN KEY yang merujuk ke item yang dibeli, dan kolom Jumlah mencatat kuantitas pembelian untuk barang tersebut. Struktur ini memungkinkan satu transaksi (satu nota) mencatat multiple barang yang dibeli, sehingga membentuk hubungan many-to-many antara transaksi dan

barang, yang merupakan fondasi untuk menghitung total penjualan, analisis produk terlaris, dan manajemen inventori.

3.8.4 Kamus Basis Data Tabel Pelanggan

Kamus Basis Data	
Nama Database	Data_Warung
Nama Tabel	Tabel_Kasir
Fungsi Tabel	Menyimpan_Data_Kasir

Nama Field	Tipe Data	Panjang Karakter	Keterangan
ID_KASIR	CHAR	5	PRIMARY KEY, Kode unik Kasir [3]
Nama_Kasir	VARCHAR	50	Nama lengkap Kasir [4]
ALAMAT	VARCHAR	100	Alamat tempat tinggal Kasir (Opsional) [5]
NO_TELEPON	VARCHAR	15	Nomor telepon/kontak Kasir (Opsional) [6]

Tabel Tabel_Kasir ini pada intinya berfungsi sebagai Tabel Master atau "Daftar Pegawai Kasir" yang menyimpan data utama setiap kasir (seperti ID, Nama, dan No. Telepon) dalam satu lokasi terpusat. Tujuan utamanya adalah untuk dihubungkan ke Tabel_Transaksi melalui *field* ID_KASIR, yang akan berperan sebagai *Foreign Key*. Dengan adanya hubungan ini, sistem dapat secara jelas melacak dan memastikan bahwa setiap nota penjualan yang terjadi di Warung Makan SE memiliki data penanggung jawab (kasir) yang tercatat.

3.9 Penerapan ERD ke DBMS

(MySQL Workbench) Penerapan ERD ke dalam DBMS pada proyek ini dilakukan menggunakan MySQL Workbench. ERD yang telah dirancang diimplementasikan melalui fitur forward engineering, sehingga rancangan visual dapat diubah secara otomatis menjadi perintah SQL. Proses ini menghasilkan database beserta tabel-tabel yang saling berelasi sesuai dengan ERD, lengkap dengan penentuan primary key dan foreign key. Dengan demikian, struktur basis data yang terbentuk pada DBMS telah sesuai dengan rancangan konseptual yang dibuat sebelumnya.

3.10 Implementasi Basis Data

3.10.1 DDL (Data Definition Language)

Implementasi Data Definition Language (DDL) merupakan tahap awal dalam penerapan basis data yang bertujuan untuk membangun struktur database sesuai dengan hasil perancangan yang telah dibuat sebelumnya. Pada tahap ini, seluruh objek basis data didefinisikan menggunakan perintah SQL agar dapat digunakan oleh sistem penjualan kasir pada Warung Makan SE.

```sql
-- SISTEM   : POS Warung Makan SE
-- ========================================================

DROP DATABASE IF EXISTS data_warung;
CREATE DATABASE data_warung;
USE data_warung;

-- ======================
-- TABLE PELANGGAN
-- ======================
CREATE TABLE tabel_pelanggan (
    id_pelanggan CHAR(5) PRIMARY KEY,
    nama_pelanggan VARCHAR(50) NOT NULL,
    jenis_layanan VARCHAR(20) NOT NULL
);

-- ======================
-- TABLE KASIR
-- ======================
CREATE TABLE tabel_kasir (
    id_kasir CHAR(5) PRIMARY KEY,
    nama_kasir VARCHAR(50) NOT NULL,
    no_telepon VARCHAR(15)
);
```

```sql
-- ======================
-- TABLE BARANG
-- ======================
CREATE TABLE tabel_barang (
    id_barang CHAR(5) PRIMARY KEY,
    nama_barang VARCHAR(100) NOT NULL,
    harga_barang INT NOT NULL
);

-- ======================
-- TABLE TRANSAKSI
-- ======================
CREATE TABLE tabel_transaksi (
    no_nota VARCHAR(17) PRIMARY KEY,
    waktu DATETIME NOT NULL,
    total_bayar INT NOT NULL,
    transfer INT NOT NULL,
    id_pelanggan CHAR(5) NOT NULL,
    id_kasir CHAR(5) NOT NULL,
    CONSTRAINT fk_pelanggan FOREIGN KEY (id_pelanggan)
        REFERENCES tabel_pelanggan(id_pelanggan),
    CONSTRAINT fk_kasir FOREIGN KEY (id_kasir)
        REFERENCES tabel_kasir(id_kasir)
);
```

```sql
-- ======================
-- TABLE DETAIL TRANSAKSI
-- ======================
CREATE TABLE tabel_detail_transaksi (
    no_nota VARCHAR(17),
    id_barang CHAR(5),
    jumlah INT NOT NULL,
    PRIMARY KEY (no_nota, id_barang),
    CONSTRAINT fk_transaksi FOREIGN KEY (no_nota)
        REFERENCES tabel_transaksi(no_nota),
    CONSTRAINT fk_barang FOREIGN KEY (id_barang)
        REFERENCES tabel_barang(id_barang)
);
```

Penjelasan :

Proses implementasi diawali dengan pembuatan database data_warung menggunakan perintah CREATE DATABASE, yang kemudian diaktifkan dengan perintah USE. Selanjutnya dilakukan pembuatan tabel-tabel utama, yaitu tabel_pelanggan**,** tabel_kasir**,** tabel_barang**,** tabel_transaksi**,** dan tabel_detail_transaksi menggunakan perintah CREATE TABLE.

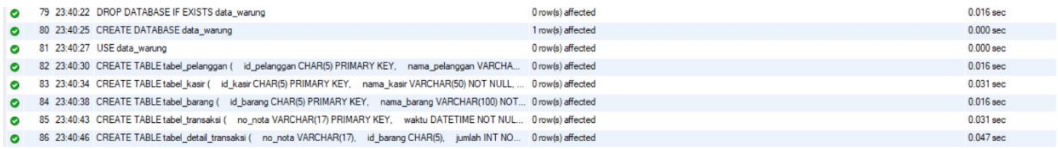
Setiap tabel dirancang dengan penetapan Primary Key (PK) untuk mengidentifikasi data secara unik. Selain itu, diterapkan Foreign Key (FK) pada tabel transaksi dan tabel

detail transaksi untuk menghubungkan data antar tabel sesuai dengan relasi yang telah ditentukan pada ERD. Penerapan foreign key ini bertujuan untuk menjaga integritas referensial, sehingga data transaksi hanya dapat disimpan apabila data pelanggan, kasir, dan barang telah tersedia.

Tabel tabel_detail_transaksi menggunakan primary key gabungan (composite key) yang terdiri dari nomor nota dan kode barang, sehingga satu transaksi dapat mencatat lebih dari satu barang. Seluruh tabel menggunakan storage engine InnoDB**,** yang mendukung penerapan constraint dan relasi antar tabel.

Dengan implementasi DDL ini, struktur basis data berhasil dibangun secara terorganisasi, konsisten, dan siap digunakan untuk proses pengolahan data serta pengujian query SQL pada tahap selanjutnya.

Lalu selanjutnya membuat sebuah tabel dengan cara CREATE TABLE nama tabel yang ingin dibuat (…) didalam kurung di isi kolom kolom yang inign digunakan contohnya seperti pada gambar lalu menentukan tipe datanya apakah varchar, integer, date dll. Setelah di run maka hasilnya akan langsung terbuatlah database di PHP MyAdmin, dan akan ada notifikasi centang warna hijau seperti ini:



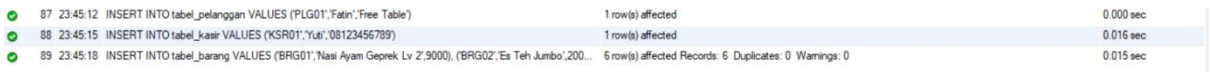
Ini adalah hasil running dari DDL

3.10.2 DML (Data Manipulation Language)

```
-- ======================
-- INSERT DATA (DML)
-- ======================
INSERT INTO tabel_pelanggan VALUES
('PLG01','Fatin','Free Table');

INSERT INTO tabel_kasir VALUES
('KSR01','Yuti','08123456789');

INSERT INTO tabel_barang VALUES
('BRG01','Nasi Ayam Geprek Lv 2',9000),
('BRG02','Es Teh Jumbo',2000),
('BRG03','Nasi Ayam Sambal Ijo',12000),
('BRG04','Es Jeruk',3000),
('BRG05','Tahu Goreng',1000),
('BRG06','Tempe Goreng',1000);
```

Implementasi Data Manipulation Language (DML) dilakukan untuk mengisi data awal ke dalam tabel-tabel yang telah dibuat sebelumnya. Pada tahap ini digunakan perintah INSERT INTO untuk menambahkan data ke tabel tabel_pelanggan**,** tabel_kasir, dan tabel_barang**.** Data pelanggan digunakan sebagai identitas pemesan, data kasir sebagai petugas yang melayani transaksi, dan data barang sebagai daftar menu yang dijual di Warung Makan SE.

Keberhasilan eksekusi perintah INSERT tanpa error menunjukkan bahwa struktur tabel, tipe data, serta penerapan primary key dan foreign key telah berjalan dengan baik. Data awal yang telah dimasukkan ini selanjutnya digunakan sebagai dasar dalam proses transaksi penjualan dan pengujian query lanjutan seperti JOIN, agregasi, dan pembuatan laporan penjualan.



3.10.3 TCL (Transaction Control Language)

digunakan untuk mengatur proses transaksi agar perubahan data berlangsung dengan aman. TCL memastikan bahwa data hanya disimpan jika proses berjalan dengan benar, dan dapat dibatalkan jika terjadi kesalahan, sehingga menjaga konsistensi data

```
-- TRANSACTION (TCL)
-- ======================
START TRANSACTION;

INSERT INTO tabel_transaksi VALUES
('CS/27/251018/0090','2025-10-18 12:25:00',28000,28000,'PLG01','KSR01');

INSERT INTO tabel_detail_transaksi VALUES
('CS/27/251018/0090','BRG01',1),
('CS/27/251018/0090','BRG02',1),
('CS/27/251018/0090','BRG03',1),
('CS/27/251018/0090','BRG04',1),
('CS/27/251018/0090','BRG05',2),
('CS/27/251018/0090','BRG06',2);

COMMIT;
```

Implementasi Transaction Control Language (TCL) digunakan untuk mengelola proses transaksi agar penyimpanan data ke dalam basis data dilakukan secara aman dan konsisten. Pada gambar yang ditampilkan, proses transaksi diawali dengan perintah START TRANSACTION, yang menandakan dimulainya satu rangkaian transaksi penjualan. Selanjutnya dilakukan penyimpanan data ke tabel tabel_transaksi dan tabel_detail_transaksi, yang mencakup informasi nomor nota, waktu transaksi, total pembayaran, serta rincian barang yang dibeli.

Setelah seluruh data transaksi dan detail transaksi berhasil dimasukkan, perintah COMMIT dijalankan untuk menyimpan seluruh perubahan secara permanen ke dalam basis data. Keberhasilan proses ini menunjukkan bahwa mekanisme transaksi berjalan dengan baik, di mana seluruh data transaksi tersimpan secara utuh dan konsisten. Penerapan TCL ini memastikan bahwa apabila terjadi kesalahan pada salah satu proses, transaksi dapat dibatalkan sehingga integritas data tetap terjaga.

✓	93	23:54:21	START TRANSACTION	0 row(s) affected	0.000 sec
✓	94	23:54:25	INSERT INTO tabel_transaksi VALUES ('CS/27/251018/0090','2025-10-18 12:25:00',28000,28000,'PLG01','...	1 row(s) affected	0.031 sec
✓	95	23:54:28	INSERT INTO tabel_detail_transaksi VALUES ('CS/27/251018/0090','BRG01',1), ('CS/27/251018/0090','BR...	6 row(s) affected Records: 6 Duplicates: 0 Warnings: 0	0.000 sec
✓	96	23:54:32	COMMIT	0 row(s) affected	0.000 sec

3.10.4 Query

Query adalah perintah yang digunakan untuk mengambil, menampilkan, dan mengolah data yang tersimpan di dalam basis data. Melalui query, pengguna dapat menentukan data apa yang ingin ditampilkan, dari tabel mana data tersebut diambil, serta bagaimana data tersebut diproses. Query memungkinkan data yang tersimpan dalam banyak tabel diolah menjadi informasi yang berguna, seperti daftar transaksi, laporan pendapatan, atau ringkasan data tertentu. Dengan adanya query, basis data tidak hanya

berfungsi sebagai tempat penyimpanan, tetapi juga sebagai sumber informasi yang dapat diakses dan dianalisis sesuai kebutuhan

3.10.4.1 Join

Query JOIN digunakan untuk menggabungkan data dari beberapa tabel yang saling berelasi. Pada sistem ini, JOIN menghubungkan tabel transaksi dengan tabel pelanggan, kasir, dan barang, sehingga data transaksi dapat ditampilkan secara lengkap dalam satu tampilan, seperti nama pelanggan, nama kasir, jenis layanan, jumlah, dan tanggal transaksi. Dengan JOIN, informasi yang tersebar di beberapa tabel dapat disajikan menjadi satu data yang utuh dan mudah dibaca.

```
-- ======================
-- JOIN QUERY
-- ======================
SELECT t.no_nota, p.nama_pelanggan, b.nama_barang, d.jumlah,
       b.harga_barang, (d.jumlah * b.harga_barang) AS subtotal
FROM tabel_transaksi t
JOIN tabel_pelanggan p ON t.id_pelanggan = p.id_pelanggan
JOIN tabel_detail_transaksi d ON t.no_nota = d.no_nota
JOIN tabel_barang b ON d.id_barang = b.id_barang;
```

Ini adalah kode query Join

Hasil dari query JOIN ini menampilkan informasi nomor nota, nama pelanggan, nama barang, jumlah pembelian, harga barang, serta subtotal yang diperoleh dari hasil perkalian jumlah dan harga barang. Keberhasilan eksekusi query ini menunjukkan bahwa relasi antar tabel telah diterapkan dengan benar dan sistem mampu menyajikan data transaksi secara terintegrasi untuk keperluan laporan dan analisis penjualan.

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

no_nota	nama_pelanggan	nama_barang	jumlah	harga_barang	subtotal
CS/27/251018/0090	Fatin	Nasi Ayam Geprek Lv 2	1	9500	9500
CS/27/251018/0090	Fatin	Es Teh Jumbo	1	2000	2000
CS/27/251018/0090	Fatin	si Ayam Sambal Ijo	1	12000	12000
CS/27/251018/0090	Fatin	Es Jeruk	1	3000	3000
CS/27/251018/0090	Fatin	Tahu Goreng	2	1000	2000
CS/27/251018/0090	Fatin	Tempe Goreng	2	1000	2000

Dan ini adalah hasil query Join setelah di run

3.10.4.2 Group By & Agregasi

Query GROUP BY dan agregasi digunakan untuk melakukan perhitungan data penjualan berdasarkan kelompok tertentu. Pada gambar yang ditampilkan, query menggabungkan tabel tabel_detail_transaksi dan tabel_barang, kemudian mengelompokkan data berdasarkan nama barang. Fungsi agregasi SUM() digunakan untuk menghitung total jumlah barang yang terjual dari seluruh transaksi yang tersimpan dalam basis data.

```
-- ======================
-- GROUP BY + AGREGASI
-- ======================
SELECT b.nama_barang, SUM(d.jumlah) AS total_terjual
FROM tabel_detail_transaksi d
JOIN tabel_barang b ON d.id_barang = b.id_barang
GROUP BY b.nama_barang;
```

Hasil dari query ini menampilkan nama barang beserta total jumlah penjualan masing-masing barang. Keberhasilan eksekusi query GROUP BY dan agregasi menunjukkan bahwa sistem mampu melakukan perhitungan data secara akurat, sehingga query ini dapat digunakan sebagai dasar pembuatan laporan penjualan dan analisis performa produk.

nama_barang	total_terjual
Es Jeruk	1
Es Teh Jumbo	1
Nasi Ayam Geprek Lv 2	1
Nasi Ayam Sambal Ijo	1
Tahu Goreng	2

Ini adalah output dari query Group By & Agregasi

3.10.4.3 Having

HAVING digunakan untuk menyaring hasil dari proses pengelompokan yang dilakukan oleh GROUP BY. Berbeda dengan WHERE yang menyaring data sebelum pengelompokan, HAVING menyaring data setelah proses agregasi. Dalam sistem ini, HAVING digunakan untuk menampilkan hanya layanan yang memiliki total pendapatan di atas nilai tertentu, sehingga informasi yang ditampilkan menjadi lebih spesifik.

```
-- ======================
-- HAVING
-- ======================
SELECT b.nama_barang, SUM(d.jumlah) AS total_terjual
FROM tabel_detail_transaksi d
JOIN tabel_barang b ON d.id_barang = b.id_barang
GROUP BY b.nama_barang
HAVING SUM(d.jumlah) >= 2;
```

HAVING digunakan untuk menyaring hasil dari query yang menggunakan fungsi agregasi. Pada gambar yang ditampilkan, query terlebih dahulu mengelompokkan data penjualan berdasarkan nama barang menggunakan GROUP BY, kemudian menghitung total jumlah barang terjual dengan fungsi SUM().

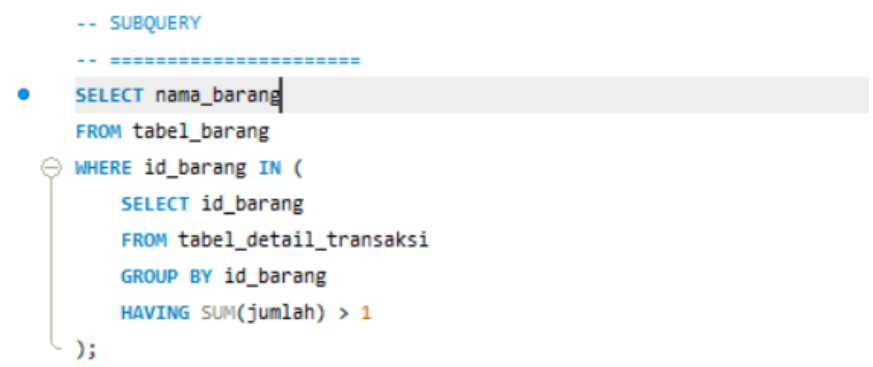
Klausa HAVING SUM(d.jumlah) >= 2 digunakan untuk menampilkan hanya barang yang memiliki jumlah penjualan dua unit atau lebih. Hasil dari query ini menunjukkan bahwa sistem mampu melakukan penyaringan data berdasarkan hasil agregasi, sehingga HAVING dapat digunakan untuk analisis penjualan, seperti mengidentifikasi produk yang paling banyak terjual.

Result Grid | Filter Rows:

nama_barang	total_terjual
Tahu Goreng	2
Tempe Goreng	2

3.10.4.4 Query Subquery

Subquery adalah query yang berada di dalam query lain. Pada sistem ini, subquery digunakan untuk menyeleksi data berdasarkan hasil perhitungan tertentu, misalnya menampilkan pelanggan yang memiliki total transaksi di atas batas tertentu. Dengan subquery, proses pencarian data menjadi lebih fleksibel dan mampu menghasilkan informasi yang lebih terarah.



```
-- SUBQUERY
-- ======================
SELECT nama_barang
FROM tabel_barang
WHERE id_barang IN (
    SELECT id_barang
    FROM tabel_detail_transaksi
    GROUP BY id_barang
    HAVING SUM(jumlah) > 1
);
```

Pada gambar yang ditampilkan, subquery digunakan untuk mencari kode barang (id_barang) yang memiliki jumlah penjualan lebih dari satu unit dengan memanfaatkan fungsi agregasi SUM(jumlah) dan klausa HAVING. Hasil subquery tersebut kemudian digunakan oleh query utama untuk menampilkan nama barang yang memenuhi kriteria tersebut. Keberhasilan eksekusi query ini menunjukkan bahwa sistem mampu melakukan pengolahan data secara bertingkat dan mendukung analisis penjualan

BAB IV

PENUTUP

4.1 Pengujian dan Hasil

Pengujian pada sistem basis data penjualan kasir Warung Makan SE dilakukan dengan mengeksekusi seluruh perintah SQL yang telah dibuat, meliputi pembuatan struktur tabel, pengisian data, serta pengujian query lanjutan. Pengujian struktur tabel dilakukan untuk memastikan bahwa seluruh tabel berhasil dibuat tanpa error, menggunakan storage engine InnoDB, serta memiliki relasi yang valid melalui penerapan primary key dan foreign key.

Selanjutnya dilakukan pengujian terhadap proses transaksi menggunakan Transaction Control Language (TCL), yaitu dengan menjalankan perintah START TRANSACTION, penyimpanan data transaksi dan detail transaksi, serta COMMIT. Hasil pengujian menunjukkan bahwa data transaksi tersimpan secara utuh dan konsisten. Pengujian query JOIN, GROUP BY, agregasi, HAVING, view, dan subquery juga menunjukkan hasil yang sesuai dengan data yang dimasukkan, sehingga dapat disimpulkan bahwa sistem basis data mampu menampilkan informasi penjualan secara akurat dan terintegrasi.

4.2 Kendala dan Perbaikan

Selama proses perancangan dan implementasi basis data, terdapat beberapa kendala yang ditemui. Salah satu kendala utama adalah terjadinya error pada penerapan foreign key constraint, khususnya pada tabel detail transaksi. Kendala ini disebabkan oleh perbedaan tipe data, panjang field, serta penggunaan storage engine yang belum sesuai, sehingga relasi antar tabel tidak dapat terbentuk dengan benar.

Perbaikan dilakukan dengan menyamakan tipe dan panjang data antara primary key dan foreign key, serta memastikan seluruh tabel menggunakan storage engine InnoDB. Selain itu, penyesuaian struktur tabel juga dilakukan agar sesuai dengan hasil normalisasi dan Entity Relationship Diagram (ERD). Setelah dilakukan perbaikan,

seluruh tabel dan relasi dapat dibuat tanpa error dan sistem dapat berjalan sesuai dengan yang diharapkan.

4.3 Kesimpulan

Berdasarkan hasil perancangan, implementasi, dan pengujian yang telah dilakukan, dapat disimpulkan bahwa sistem basis data penjualan kasir (Point of Sale) pada Warung Makan SE berhasil dirancang dan diimplementasikan dengan baik. Proses normalisasi data telah dilakukan hingga bentuk normal ketiga (3NF) sehingga struktur basis data menjadi lebih efisien dan terhindar dari redundansi data.

Implementasi basis data menggunakan MySQL telah berhasil dilakukan dengan penerapan Data Definition Language (DDL), Data Manipulation Language (DML), Transaction Control Language (TCL), serta query lanjutan seperti JOIN, agregasi, HAVING, dan subquery. Seluruh query dapat dijalankan tanpa error dan menghasilkan keluaran sesuai dengan kebutuhan sistem, sehingga basis data siap digunakan untuk mendukung proses pencatatan dan pengolahan data penjualan.

4.4 Saran

Sistem basis data yang dirancang pada proyek ini masih berfokus pada pengelolaan data transaksi. Ke depannya, sistem ini dapat dikembangkan menjadi aplikasi sederhana berbasis web atau desktop agar proses input data menjadi lebih mudah dan tidak perlu dilakukan langsung melalui perintah SQL.Selain itu, struktur transaksi dapat diperbaiki dengan memisahkan data transaksi menjadi tabel utama dan tabel detail, sehingga satu transaksi dapat memuat lebih dari satu layanan dalam satu nota.

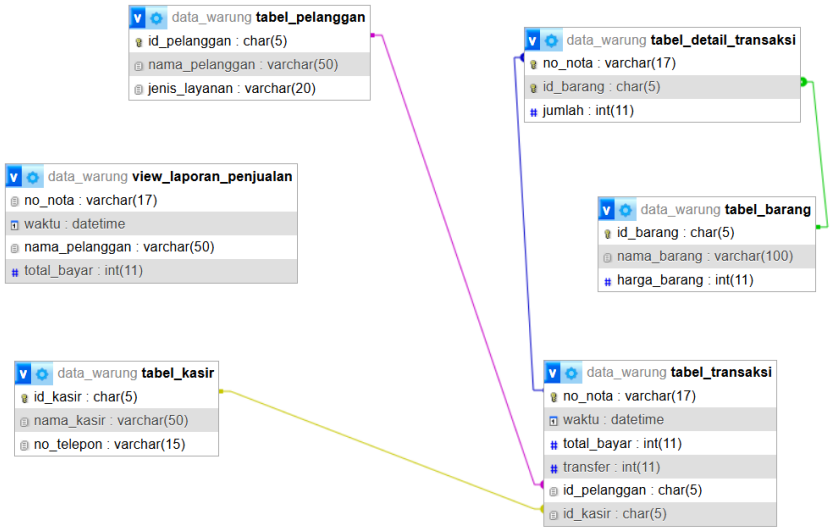
Hal ini akan membuat sistem lebih fleksibel dan mendekati kondisi nyata di lapangan. Pengembangan lain yang dapat dilakukan adalah menambahkan perhitungan otomatis untuk nilai sub_total, total_bayar, dan kurang_sisa, sehingga kasir tidak perlu menghitung secara manual. Dengan pengembangan tersebut, sistem akan menjadi lebih praktis, mengurangi kesalahan input, dan lebih siap digunakan dalam kegiatan operasional sehari-hari.

4.5 LAMPIRAN

1. Tautan GitHub Repository

   https://github.com/muhsafiq430-ai/Db_warung_makan_SE Struktur Folder

2. Screenshot ERD



Gambar ini adalah ERD final

3. Screenshot Hasil Query

a. Join

.

```
-- ======================
-- JOIN QUERY
-- ======================
SELECT t.no_nota, p.nama_pelanggan, b.nama_barang, d.jumlah,
       b.harga_barang, (d.jumlah * b.harga_barang) AS subtotal
FROM tabel_transaksi t
JOIN tabel_pelanggan p ON t.id_pelanggan = p.id_pelanggan
JOIN tabel_detail_transaksi d ON t.no_nota = d.no_nota
JOIN tabel_barang b ON d.id_barang = b.id_barang;
```

Ini adalah kode query Join

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

no_nota	nama_pelanggan	nama_barang	jumlah	harga_barang	subtotal
CS/27/251018/0090	Fatin	Nasi Ayam Geprek Lv 2	1	9500	9500
CS/27/251018/0090	Fatin	Es Teh Jumbo	1	2000	2000
CS/27/251018/0090	Fatin	si Ayam Sambal Ijo	1	12000	12000
CS/27/251018/0090	Fatin	Es Jeruk	1	3000	3000
CS/27/251018/0090	Fatin	Tahu Goreng	2	1000	2000
CS/27/251018/0090	Fatin	Tempe Goreng	2	1000	2000

Dan ini adalah hasil query Join

b. Group By & Agregasi

```
-- ======================
-- GROUP BY + AGREGASI
-- ======================
SELECT b.nama_barang, SUM(d.jumlah) AS total_terjual
FROM tabel_detail_transaksi d
JOIN tabel_barang b ON d.id_barang = b.id_barang
GROUP BY b.nama_barang;
```

nama_barang	total_terjual
Es Jeruk	1
Es Teh Jumbo	1
Nasi Ayam Geprek Lv 2	1
Nasi Ayam Sambal Ijo	1
Tahu Goreng	2

Ini adalah output dari query Group By & Agregasi

4 Having

```
-- ======================
-- HAVING
-- ======================
SELECT b.nama_barang, SUM(d.jumlah) AS total_terjual
FROM tabel_detail_transaksi d
JOIN tabel_barang b ON d.id_barang = b.id_barang
GROUP BY b.nama_barang
HAVING SUM(d.jumlah) >= 2;
```

Result Grid | Filter Rows:

	nama_barang	total_terjual
▶	Tahu Goreng	2
	Tempe Goreng	2

Gambar ini adalah hasil running dari kode Query Having

5 Query Subquery

```
-- SUBQUERY
-- ======================
SELECT nama_barang
FROM tabel_barang
WHERE id_barang IN (
    SELECT id_barang
    FROM tabel_detail_transaksi
    GROUP BY id_barang
    HAVING SUM(jumlah) > 1
);
```

Gambar ini adalah kode program dari Query Subquery

4.6 Script SQL Lengkap

```sql
CREATE DATABASE data_warung;
USE data_warung;

-- ======================
-- TABLE PELANGGAN
-- ======================
CREATE TABLE tabel_pelanggan (
    id_pelanggan CHAR(5) PRIMARY KEY,
    nama_pelanggan VARCHAR(50) NOT NULL,
    jenis_layanan VARCHAR(20) NOT NULL
);

-- ======================
-- TABLE KASIR
-- ======================
CREATE TABLE tabel_kasir (
    id_kasir CHAR(5) PRIMARY KEY,
    nama_kasir VARCHAR(50) NOT NULL,
    no_telepon VARCHAR(15)
);

-- ======================
-- TABLE BARANG
-- ======================
CREATE TABLE tabel_barang (
    id_barang CHAR(5) PRIMARY KEY,
    nama_barang VARCHAR(100) NOT NULL,
    harga_barang INT NOT NULL
);

-- ======================
```

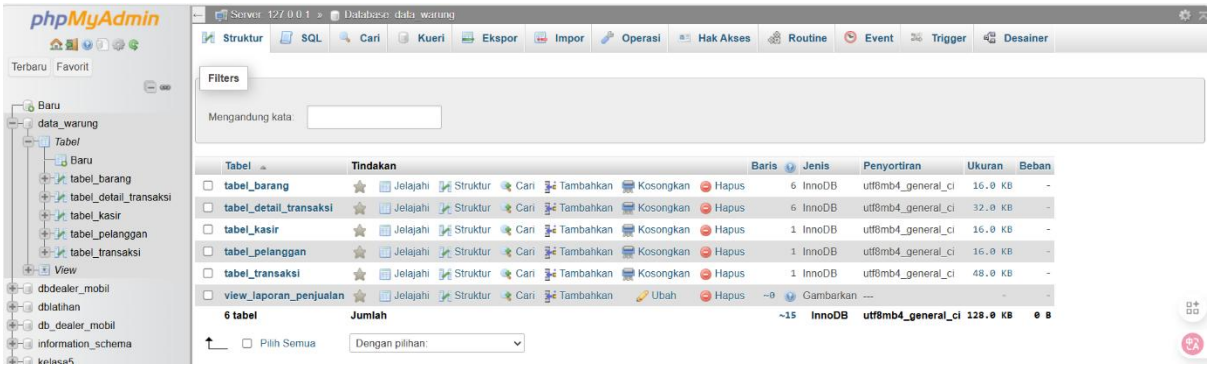
```sql
-- ======================
-- TABLE TRANSAKSI
-- ======================
CREATE TABLE tabel_transaksi (
    no_nota VARCHAR(17) PRIMARY KEY,
    waktu DATETIME NOT NULL,
    total_bayar INT NOT NULL,
    transfer INT NOT NULL,
    id_pelanggan CHAR(5) NOT NULL,
    id_kasir CHAR(5) NOT NULL,
    CONSTRAINT fk_pelanggan FOREIGN KEY (id_pelanggan)
        REFERENCES tabel_pelanggan(id_pelanggan),
    CONSTRAINT fk_kasir FOREIGN KEY (id_kasir)
        REFERENCES tabel_kasir(id_kasir)
);

-- ======================
-- TABLE DETAIL TRANSAKSI
-- ======================
CREATE TABLE tabel_detail_transaksi (
    no_nota VARCHAR(17),
    id_barang CHAR(5),
    jumlah INT NOT NULL,
    PRIMARY KEY (no_nota, id_barang),
    CONSTRAINT fk_transaksi FOREIGN KEY (no_nota)
        REFERENCES tabel_transaksi(no_nota),
    CONSTRAINT fk_barang FOREIGN KEY (id_barang)
        REFERENCES tabel_barang(id_barang)
);
```

```sql
-- ======================
-- INSERT DATA (DML)
-- ======================
INSERT INTO tabel_pelanggan VALUES
('PLG01','Fatin','Free Table');

INSERT INTO tabel_kasir VALUES
('KSR01','Yuti','08123456789');

INSERT INTO tabel_barang VALUES
('BRG01','Nasi Ayam Geprek Lv 2',9000),
('BRG02','Es Teh Jumbo',2000),
('BRG03','Nasi Ayam Sambal Ijo',12000),
('BRG04','Es Jeruk',3000),
('BRG05','Tahu Goreng',1000),
('BRG06','Tempe Goreng',1000);

-- ======================
-- TRANSACTION (TCL)
-- ======================
START TRANSACTION;

INSERT INTO tabel_transaksi VALUES
('CS/27/251018/0090','2025-10-18 12:25:00',28000,28000,'PLG01','KSR01');

INSERT INTO tabel_detail_transaksi VALUES
('CS/27/251018/0090','BRG01',1),
('CS/27/251018/0090','BRG02',1),
('CS/27/251018/0090','BRG03',1),
('CS/27/251018/0090','BRG04',1),
('CS/27/251018/0090','BRG05',2),
('CS/27/251018/0090','BRG06',2);

COMMIT;

-- ======================
```

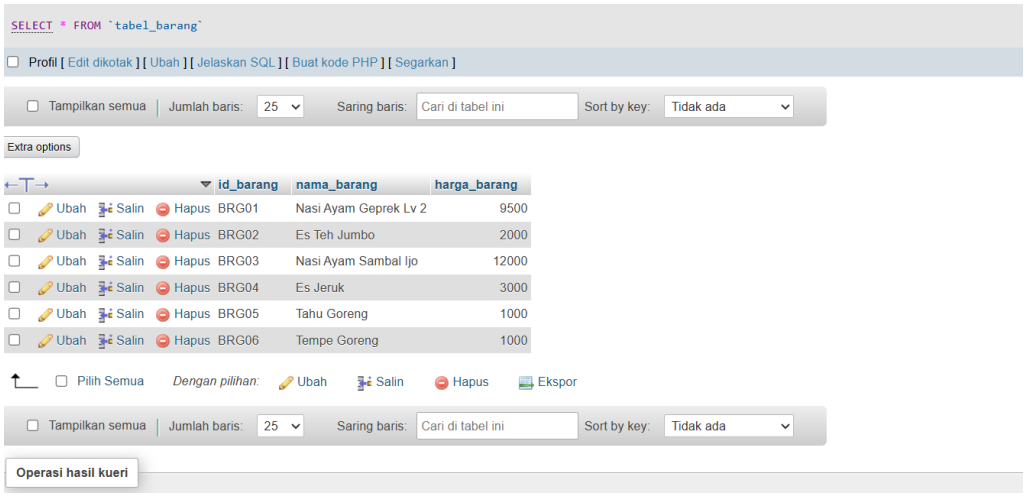
```sql
-- ======================
-- UPDATE
-- ======================
UPDATE tabel_barang
SET harga_barang = 9500
WHERE id_barang = 'BRG01';

-- ======================
-- JOIN QUERY
-- ======================
SELECT t.no_nota, p.nama_pelanggan, b.nama_barang, d.jumlah,
       b.harga_barang, (d.jumlah * b.harga_barang) AS subtotal
FROM tabel_transaksi t
JOIN tabel_pelanggan p ON t.id_pelanggan = p.id_pelanggan
JOIN tabel_detail_transaksi d ON t.no_nota = d.no_nota
JOIN tabel_barang b ON d.id_barang = b.id_barang;

-- ======================
-- GROUP BY + AGREGASI
-- ======================
SELECT b.nama_barang, SUM(d.jumlah) AS total_terjual
FROM tabel_detail_transaksi d
JOIN tabel_barang b ON d.id_barang = b.id_barang
GROUP BY b.nama_barang;

-- ======================
-- HAVING
-- ======================
SELECT b.nama_barang, SUM(d.jumlah) AS total_terjual
FROM tabel_detail_transaksi d
JOIN tabel_barang b ON d.id_barang = b.id_barang
GROUP BY b.nama_barang
HAVING SUM(d.jumlah) >= 2;

-- ======================
```

```sql
-- ========================
-- VIEW
-- ========================
CREATE VIEW view_laporan_penjualan AS
SELECT t.no_nota, t.waktu, p.nama_pelanggan, t.total_bayar
FROM tabel_transaksi t
JOIN tabel_pelanggan p ON t.id_pelanggan = p.id_pelanggan;

-- ========================
-- SUBQUERY
-- ========================
SELECT nama_barang
FROM tabel_barang
WHERE id_barang IN (
    SELECT id_barang
    FROM tabel_detail_transaksi
    GROUP BY id_barang
    HAVING SUM(jumlah) > 1
);
```

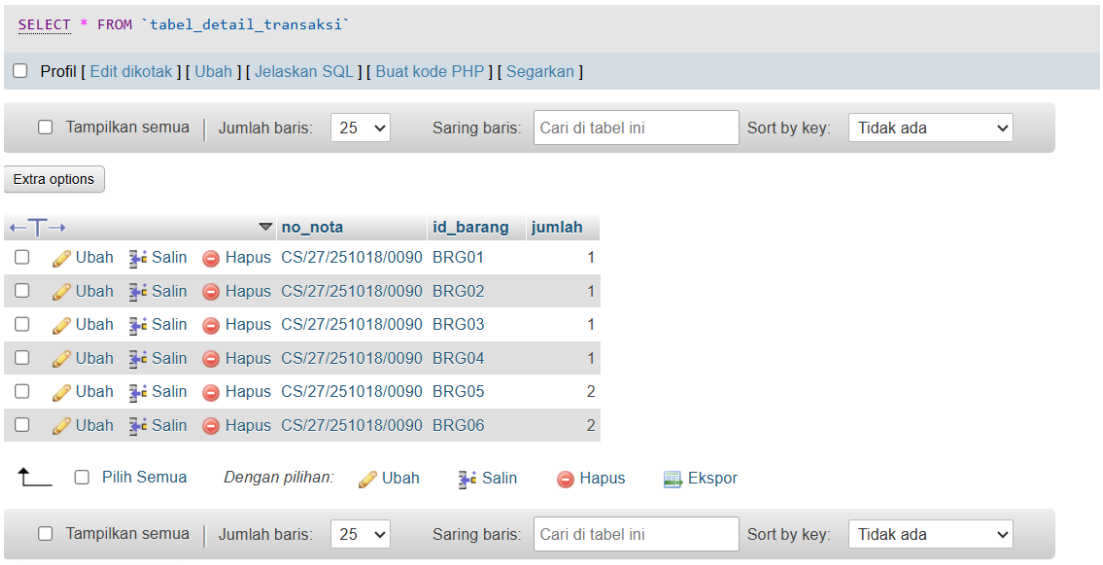
4.7 Database (PHP MyAdmin)



Gambar ini adalah database dari sistem kami



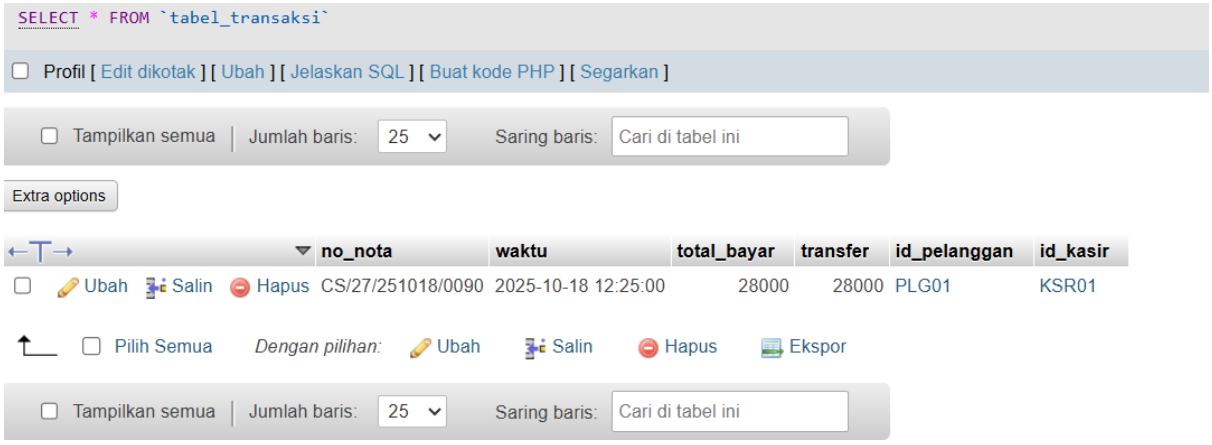
Gambar ini adalah isi dari tabel barang



Gambar ini adalah isi dari tabel transaksi

Gambar ini adalah isi dari tabel kasir

Gambar ini adalah isi dari tabel pelanggan



Gambar ini adalah ini dari tabel transaksi

DAFTAR PUSTAKA

Connolly T, Begg C. *Database Systems: A Practical Approach to Design, Implementation, and Management*. 6th ed. Boston: Pearson Education; 2015.

Elmasri R, Navathe SB. *Fundamentals of Database Systems*. 7th ed. Boston: Pearson; 2017.

Silberschatz A, Korth HF, Sudarshan S. *Database System Concepts*. 6th ed. New York: McGraw-Hill; 2019.

Coronel C, Morris S. *Database Systems: Design, Implementation, & Management*. 13th ed. Boston: Cengage Learning; 2020.

MySQL Documentation Team. *MySQL 8.0 Reference Manual* [Internet]. Oracle; 2023 [cited 2026 Jan 15]. Available from: https://dev.mysql.com/doc/

Date CJ. *An Introduction to Database Systems*. 8th ed. Boston: Addison-Wesley; 2003.

Hoffer JA, Ramesh V, Topi H. *Modern Database Management*. 13th ed. Boston: Pearson; 2019.

Widodo P, Handayani R. *Basis Data*. Jakarta: Andi Publisher; 2018.

Kadir A. *Pengenalan Sistem Informasi*. Yogyakarta: Andi; 2014.

Sutanta E. *Basis Data dalam Tinjauan Konseptual*. Yogyakarta: Andi; 2011.