

## **PROJECT PEMOGRAMAN BASIS DATA**

### **STUDI KASUS NOTA TRANSAKSI**

*Laporan ini dibuat guna memenuhi Ujian Tengah Semester kuliah Pemograman Basis Data yang diampu oleh:*

*Ridwan Dwi Irawan, S.Kom, M.Kom*



**UNIVERSITAS  
DUTA BANGSA  
SURAKARTA**

*Disusun oleh :*

Riski Dewanti 240103176

Salsa Sabila 240103177

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS DUTA BANGSA**

**2026**

## KATA PENGANTAR

Puji syukur kehadirat Allah Swt. atas rahmat dan hidayah-Nya, sehingga penulis dapat menyelesaikan tugas yang berjudul “Project Pemrograman Basis Data: Nota Coffee Shop” tepat pada waktunya.

Adapun tujuan dari penyusunan laporan ini adalah untuk memenuhi salah satu tugas pada mata kuliah Pemrograman Basis Data. Selain itu, laporan ini juga bertujuan untuk menambah pengetahuan dan pemahaman penulis mengenai penerapan konsep basis data dalam pembuatan sistem nota penjualan pada sebuah coffee shop.

Penulis mengucapkan terima kasih kepada Bapak Ridwan Dwi Irawan S. Kom, M. Kom. Selaku dosen pengampu mata kuliah Pemrograman Basis Data yang telah memberikan bimbingan dan tugas ini, sehingga penulis dapat memperdalam pemahaman mengenai perancangan dan implementasi basis data.

Ucapan terima kasih juga penulis sampaikan kepada semua pihak yang telah membantu dalam penyusunan laporan ini, baik secara langsung maupun tidak langsung.

Penulis menyadari bahwa laporan ini masih jauh dari sempurna. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang bersifat membangun untuk menyempurnakan laporan ini di masa mendatang.

Surakarta, Januari 2026

Penulis

## DAFTAR ISI

<b>KATA PENGANTAR.....</b>	2
<b>BAB I.....</b>	5
<b>PENDAHULUAN .....</b>	5
<b>1.1 Latar Belakang .....</b>	5
<b>1.2 Tujuan .....</b>	6
<b>1.3 Ruang Lingkup.....</b>	6
<b>BAB II .....</b>	7
<b>LANDASAN TEORI.....</b>	7
<b>2.1 Basis Data.....</b>	7
<b>2.2 Database Management System (DBMS) .....</b>	7
<b>2.3 Relasi Tabel .....</b>	7
<b>2.4 Entity Relationship Diagram.....</b>	7
<b>2.5 Normalisasi Basis Data .....</b>	8
<b>2.6 Relasi Basis Data dan Derajat Kardinalitas .....</b>	8
<b>2.7 Penerapan ERD ke DBMS .....</b>	9
<b>2.8 MySQL Workbench .....</b>	9
<b>2.9 Structured Query Language (SQL).....</b>	9
<b>2.10 Data Definition Language (DDL).....</b>	9
<b>2.11 Data Manipulation Language (DML) .....</b>	9
<b>2.12 Transaction Control Language (TCL).....</b>	10
<b>2.13 Agregasi dan Having.....</b>	10
<b>2.14 Group By.....</b>	10
<b>2.15 Join .....</b>	10

<b>BAB III.....</b>	11
<b>PERANCANGAN DAN IMPLEMENTASI .....</b>	11
<b>3.1     Analisis Kebutuhan Sistem.....</b>	11
<b>3.2     Perancangan Basis Data .....</b>	12
<b>3.3     Implementasi Basis Data .....</b>	14
<b>3.4     Implementasi Query SQL.....</b>	18
<b>BAB IV .....</b>	22
<b>PENUTUP.....</b>	22
<b>4.1     Ringkasan Query dan Hasil Pengujian .....</b>	22
<b>4.2     Kendala dan Perbaikan .....</b>	22
<b>4.3     Kesimpulan .....</b>	23
<b>4.4     Saran Pengembangan .....</b>	23
<b>LAMPIRAN.....</b>	24
<b>A.     Link Repository Github: <a href="https://github.com/Riskidewanti/riskidewanti">https://github.com/Riskidewanti/riskidewanti</a> .....</b>	24
<b>B.     Entity Relationship Diagram (ERD) .....</b>	24
<b>C.     Relasi Antar Tabel.....</b>	24
<b>D.     Implementasi DDL.....</b>	25
<b>E.     Implementasi DML.....</b>	27
<b>F.     Implementasi TCL .....</b>	28
<b>G.     Query Agregasi dan Having.....</b>	29
<b>H.     Query Group By .....</b>	30
<b>I.     Query Join .....</b>	30
<b>DAFTAR PUSTAKA .....</b>	33

## **BAB I**

### **PENDAHULUAN**

#### **1.1 Latar Belakang**

Perkembangan teknologi informasi saat ini telah membawa perubahan yang signifikan dalam pengelolaan data di berbagai bidang, termasuk sektor bisnis dan pelayanan. Salah satu bentuk penerapan teknologi informasi dalam dunia bisnis dapat dilihat pada sistem penjualan di coffee shop. Dalam kegiatan operasionalnya, coffee shop membutuhkan sistem yang mampu mencatat setiap transaksi penjualan secara akurat, cepat, dan terstruktur agar proses bisnis dapat berjalan secara efisien.

Pada praktiknya, pencatatan transaksi penjualan masih sering dilakukan secara manual atau menggunakan aplikasi sederhana yang belum terintegrasi dengan baik. Kondisi tersebut berpotensi menimbulkan berbagai permasalahan, seperti kesalahan pencatatan data, duplikasi informasi, serta kesulitan dalam melakukan rekap dan analisis laporan penjualan. Oleh karena itu, diperlukan suatu sistem yang berbasis basis data untuk mengelola data transaksi secara terpusat dan terstruktur.

Melalui mata kuliah Pemrograman Basis Data, mahasiswa ditugaskan untuk merancang dan mengimplementasikan sebuah sistem nota penjualan berbasis database. Sistem ini dirancang untuk menyimpan dan mengelola data transaksi, data barang, data layanan, serta data pelayan (waitress) secara terintegrasi menggunakan Database Management System (DBMS). Dalam pengelolaan basis data tersebut, digunakan bahasa SQL (Structured Query Language) yang mencakup perintah DDL, DML, dan TCL sebagai sarana untuk mendefinisikan, memanipulasi, serta mengendalikan data.

Salah satu tahapan penting dalam perancangan sistem basis data adalah proses normalisasi, yaitu proses penyusunan struktur tabel agar data lebih terorganisir, mengurangi terjadinya redundansi, serta menjaga konsistensi data. Selain itu, penerapan query seperti JOIN, GROUP BY, dan HAVING digunakan untuk menampilkan dan mengolah data transaksi secara informatif sesuai kebutuhan sistem.

Dengan adanya penerapan normalisasi dan implementasi SQL dalam perancangan basis data Nota Coffee Shop, diharapkan sistem yang dibangun mampu menghasilkan penyimpanan data yang optimal, mempercepat proses transaksi, serta meminimalkan kesalahan dalam pengelolaan informasi.

## **1.2 Tujuan**

Adapun tujuan dari penyusunan laporan proyek “Nota Coffee Shop” ini adalah sebagai berikut:

1. Menerapkan konsep dasar basis data dalam perancangan sistem informasi penjualan dengan menggunakan pendekatan normalisasi.
2. Mendesain struktur tabel yang efisien dan terorganisir sehingga data yang disimpan terhindar dari pengulangan (redundansi) dan mudah untuk diolah.
3. Membangun hubungan antar entitas (relasi) dalam sistem penjualan yang mencakup data transaksi, barang, layanan, dan pelayan.
4. Meningkatkan pemahaman mahasiswa mengenai implementasi teori Pemrograman Basis Data dalam penerapan kasus nyata.
5. Menghasilkan laporan yang sistematis dan terstruktur sebagai dokumentasi proses perancangan dan implementasi sistem basis data pada coffee shop.

## **1.3 Ruang Lingkup**

Ruang lingkup pada penyusunan laporan proyek “Nota Coffee Shop” ini dibatasi pada perancangan dan implementasi basis data sebagai pendukung sistem pencatatan transaksi penjualan. Adapun batasan masalah dalam proyek ini adalah sebagai berikut:

1. Sistem yang dibahas hanya berfokus pada pengelolaan basis data nota transaksi coffee shop, meliputi data barang, data layanan, data pelayan (waitress), data transaksi, dan detail transaksi.
2. Perancangan basis data dilakukan menggunakan Entity Relationship Diagram (ERD) dan proses normalisasi hingga bentuk normal ketiga (3NF).
3. Implementasi basis data menggunakan Database Management System (DBMS) MySQL.
4. Pengolahan dan manipulasi data dilakukan menggunakan bahasa SQL (Structured Query Language) yang mencakup perintah DDL, DML, dan TCL.
5. Query yang dibahas meliputi penggunaan JOIN, GROUP BY, dan HAVING untuk menampilkan dan mengolah data transaksi.
6. Sistem yang dibangun tidak membahas aspek antarmuka pengguna (user interface), keamanan basis data, maupun pengembangan aplikasi secara menyeluruh.
7. Proyek ini bertujuan sebagai sarana pembelajaran akademik dan tidak digunakan sebagai sistem operasional yang berjalan secara nyata.

## **BAB II**

### **LANDASAN TEORI**

#### **2.1 Basis Data**

Basis data merupakan kumpulan data yang saling berhubungan dan disimpan secara sistematis sehingga dapat dikelola dan diolah dengan mudah. Basis data digunakan untuk menyimpan data dalam jumlah besar agar proses pencarian, pengolahan, serta pembaruan data dapat dilakukan secara efisien dan terstruktur. Dalam sistem informasi, basis data berperan sebagai pusat penyimpanan data yang mendukung keberlangsungan proses bisnis.

#### **2.2 Database Management System (DBMS)**

Database Management System (DBMS) adalah perangkat lunak yang digunakan untuk mengelola basis data, mulai dari proses pembuatan basis data, penyimpanan data, hingga pengolahan dan pengamanan data. DBMS berfungsi sebagai perantara antara pengguna dengan basis data agar pengelolaan data dapat dilakukan secara terkontrol dan konsisten. Pada proyek Nota Coffee Shop, DBMS yang digunakan adalah MySQL, yaitu DBMS relasional yang mendukung pengelolaan data menggunakan bahasa SQL.

#### **2.3 Relasi Tabel**

Relasi tabel merupakan hubungan yang terbentuk antara satu tabel dengan tabel lainnya dalam sebuah basis data. Relasi ini dibangun menggunakan primary key sebagai identitas unik suatu tabel dan foreign key sebagai penghubung antar tabel. Dengan adanya relasi tabel, data yang tersimpan dapat saling terintegrasi dan menghindari terjadinya redundansi data.

#### **2.4 Entity Relationship Diagram**

Entity Relationship Diagram (ERD) adalah diagram yang digunakan untuk memodelkan struktur basis data secara konseptual. ERD menggambarkan entitas, atribut, serta hubungan antar entitas dalam suatu sistem. ERD berfungsi sebagai acuan awal dalam perancangan basis data sebelum diimplementasikan ke dalam DBMS.

#### **2.4.1 ERD Sistem Nota Coffee Shop**

ERD pada sistem Nota Coffee Shop terdiri dari beberapa entitas utama, yaitu Barang, Layanan, Waitress, Transaksi, dan Detail Transaksi. Setiap entitas saling berhubungan untuk mendukung proses pencatatan transaksi penjualan secara terintegrasi.

#### **2.4.2 Primary Key dan Foreign Key**

Primary Key merupakan atribut yang digunakan sebagai identitas unik pada setiap tabel, sedangkan Foreign Key digunakan untuk membangun hubungan antar tabel. Penggunaan primary key dan foreign key bertujuan untuk menjaga integritas data serta memastikan keterkaitan antar entitas dalam basis data.

### **2.5 Normalisasi Basis Data**

Normalisasi merupakan proses pengelompokan data ke dalam tabel-tabel yang lebih terstruktur dengan tujuan mengurangi redundansi data dan menjaga konsistensi data. Proses normalisasi dilakukan melalui beberapa tahap bentuk normal.

#### **2.5.1 Bentuk Normal Pertama (1NF)**

Bentuk Normal Pertama (1NF) mensyaratkan bahwa setiap atribut dalam tabel harus bernilai atomik dan tidak mengandung data berulang dalam satu kolom.

#### **2.5.2 Bentuk Normal Kedua (2NF)**

Bentuk Normal Kedua (2NF) mengharuskan setiap atribut non-primary key bergantung sepenuhnya pada primary key dan tidak bergantung sebagian.

#### **2.5.3 Bentuk Normal Ketiga (3NF)**

Bentuk Normal Ketiga (3NF) mensyaratkan bahwa setiap atribut non-primary key tidak bergantung pada atribut non-primary key lainnya sehingga struktur tabel menjadi lebih optimal dan bebas dari ketergantungan tidak langsung.

### **2.6 Relasi Basis Data dan Derajat Kardinalitas**

Relasi basis data menunjukkan keterkaitan antar entitas yang terdapat dalam basis data. Derajat kardinalitas menggambarkan jumlah hubungan antar entitas, seperti one to one, one to many, dan many to many. Pada sistem Nota Coffee Shop, relasi antar tabel digunakan untuk menghubungkan data transaksi dengan data barang, layanan, dan waitress.

## **2.7 Penerapan ERD ke DBMS**

Penerapan ERD ke dalam DBMS dilakukan dengan mengubah setiap entitas pada ERD menjadi tabel dalam basis data. Relasi yang terdapat pada ERD diterjemahkan ke dalam bentuk primary key dan foreign key sehingga struktur basis data sesuai dengan rancangan konseptual.

## **2.8 MySQL Workbench**

MySQL Workbench merupakan alat bantu yang digunakan untuk merancang, mengelola, dan mengimplementasikan basis data MySQL. MySQL Workbench mendukung pembuatan ERD, forward engineering, serta pengelolaan query SQL secara visual.

## **2.9 Structured Query Language (SQL)**

Structured Query Language (SQL) merupakan bahasa standar yang digunakan untuk mengelola basis data relasional. SQL digunakan untuk mendefinisikan struktur basis data, memanipulasi data, serta mengatur transaksi pada basis data.

## **2.10 Data Definition Language (DDL)**

Data Definition Language (DDL) adalah bagian dari SQL yang digunakan untuk mendefinisikan dan mengatur struktur basis data. Perintah DDL meliputi CREATE, ALTER, dan DROP yang digunakan untuk mengelola objek basis data.

## **2.11 Data Manipulation Language (DML)**

Data Manipulation Language (DML) merupakan bagian dari SQL yang digunakan untuk melakukan manipulasi data pada basis data. Perintah DML meliputi INSERT, UPDATE, DELETE, dan SELECT.

## **2.12 Transaction Control Language (TCL)**

Transaction Control Language (TCL) digunakan untuk mengatur transaksi pada basis data. Perintah TCL seperti COMMIT dan ROLLBACK berfungsi untuk menjaga konsistensi data selama proses transaksi berlangsung.

## **2.13 Agregasi dan Having**

Agregasi digunakan untuk melakukan perhitungan terhadap sekumpulan data, seperti perhitungan jumlah, total, dan rata-rata. Klausula HAVING digunakan untuk memberikan kondisi pada hasil pengelompokan data.

## **2.14 Group By**

GROUP BY merupakan klausula dalam SQL yang digunakan untuk mengelompokkan data berdasarkan satu atau lebih kolom tertentu sehingga data dapat dianalisis secara terstruktur.

## **2.15 Join**

JOIN merupakan perintah SQL yang digunakan untuk menggabungkan data dari dua atau lebih tabel berdasarkan relasi yang telah ditentukan. JOIN digunakan untuk menampilkan data yang saling berhubungan dalam satu hasil query.

## **BAB III**

### **PERANCANGAN DAN IMPLEMENTASI**

#### **3.1 Analisis Kebutuhan Sistem**

Analisis kebutuhan sistem dilakukan untuk mengidentifikasi kebutuhan utama dalam perancangan sistem basis data Nota Coffee Shop. Sistem ini dirancang untuk mendukung proses pencatatan transaksi penjualan secara terstruktur, sehingga data yang tersimpan dapat dikelola dengan baik dan mudah diolah.

Pada sistem Nota Coffee Shop, data yang dikelola meliputi data barang, data layanan, data pelayan (waitress), data transaksi, dan data detail transaksi. Sistem harus mampu mencatat setiap transaksi penjualan beserta rincian barang yang dibeli, layanan yang digunakan, serta pelayan yang menangani transaksi tersebut.

Sistem basis data ini dibangun menggunakan DBMS MySQL dan dikelola melalui MySQL Workbench. Pengelolaan data dilakukan menggunakan bahasa SQL, yang mencakup perintah untuk pendefinisian struktur basis data, manipulasi data, serta pengendalian transaksi. Dengan adanya sistem basis data ini, proses pencatatan transaksi diharapkan dapat dilakukan secara lebih akurat, konsisten, dan efisien.

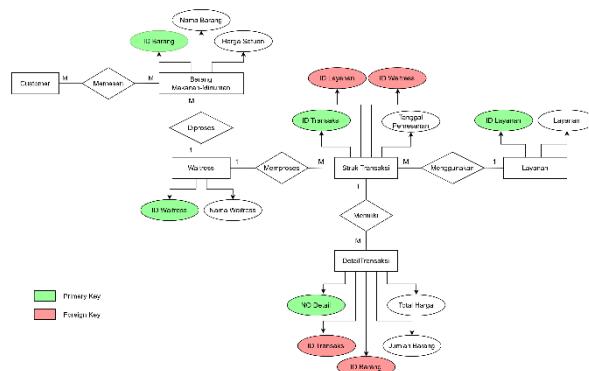
Selain itu, sistem harus mampu menampilkan informasi transaksi dalam bentuk query, seperti rekap data transaksi, perhitungan total penjualan, serta pengelompokan data berdasarkan kriteria tertentu. Oleh karena itu, sistem dirancang agar mendukung penggunaan query lanjutan seperti agregasi, GROUP BY, HAVING, dan JOIN.

Sistem Nota Coffee Shop yang dirancang pada proyek ini difokuskan sebagai sarana pembelajaran akademik dalam mata kuliah Pemrograman Basis Data. Oleh karena itu, sistem ini hanya mencakup perancangan dan implementasi basis data tanpa membahas pengembangan antarmuka pengguna maupun aspek keamanan sistem secara mendalam.

### 3.2 Perancangan Basis Data

Perancangan basis data dilakukan untuk menggambarkan struktur database yang akan digunakan pada sistem Nota Coffee Shop. Perancangan ini bertujuan agar data yang disimpan terorganisir dengan baik, saling terhubung, serta mendukung proses transaksi penjualan secara efisien. Pada tahap ini digunakan Entity Relationship Diagram (ERD) sebagai dasar perancangan database sebelum diimplementasikan ke dalam DBMS.

#### 3.2.1 ERD



#### 3.2.2 Deskripsi Entitas Antar Tabel

##### 1. Costumer

Entitas Customer digunakan untuk menggambarkan pelanggan yang melakukan pemesanan di coffee shop. Pada perancangan ini, Customer hanya digunakan untuk menunjukkan alur pemesanan secara umum dan tidak diimplementasikan secara langsung ke dalam basis data.

##### Relasi:

Melakukan pemesanan barang makanan dan minuman.

##### 2. Barang Makanan/Minuman

Entitas Barang Makanan/Minuman digunakan untuk menyimpan data makanan dan minuman yang dijual pada coffee shop.

Atribut yang dimiliki yaitu **ID\_Barang** sebagai primary key, **Nama Barang**, dan **Harga Satuan**.

**Relasi:** Dipesan oleh Customer,  
Dicatat dalam Detail Transaksi.

##### 3. Waitress

Entitas Waitress digunakan untuk menyimpan data pelayan yang menangani transaksi penjualan.

Atribut yang dimiliki adalah **ID\_Waitress** sebagai primary key dan **Nama Waitress**.

**Relasi:** Memproses transaksi penjualan.

#### **4. Layanan**

Entitas Layanan digunakan untuk menyimpan jenis layanan yang digunakan dalam transaksi, seperti dine-in atau take away.

Atribut yang dimiliki adalah ID\_Layanan sebagai primary key dan Nama Layanan.

**Relasi:** Digunakan dalam transaksi penjualan.

#### **5. Transaksi**

Entitas Transaksi digunakan untuk menyimpan data transaksi penjualan yang terjadi di coffee shop.

Atribut yang dimiliki antara lain ID\_Transaksi sebagai primary key, Tanggal Pemesanan, dan ID\_Layanan sebagai foreign key.

**Relasi:** Diproses oleh Waitress,

Menggunakan Layanan,

Memiliki Detail Transaksi

.

#### **6. Detail Transaksi**

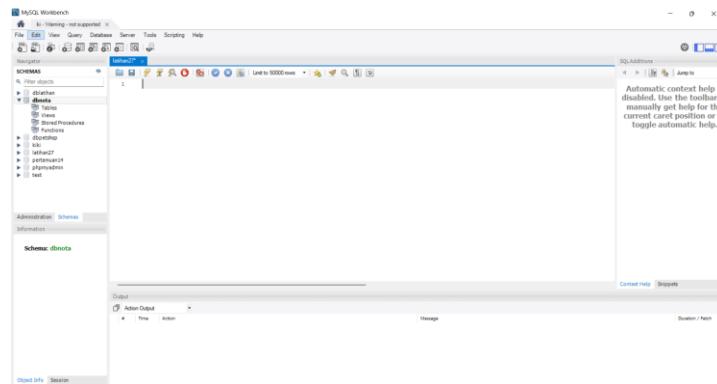
Entitas Detail Transaksi digunakan untuk menyimpan rincian barang yang dibeli pada setiap transaksi.

Atribut yang dimiliki adalah No\_Detail sebagai primary key, ID\_Transaksi dan ID\_Barang sebagai foreign key, Jumlah Barang, dan Total Harga.

**Relasi:** Menghubungkan data Transaksi dengan Barang.

### 3.3 Implementasi Basis Data

#### 3.3.1 Penerapan ERD ke DBMS

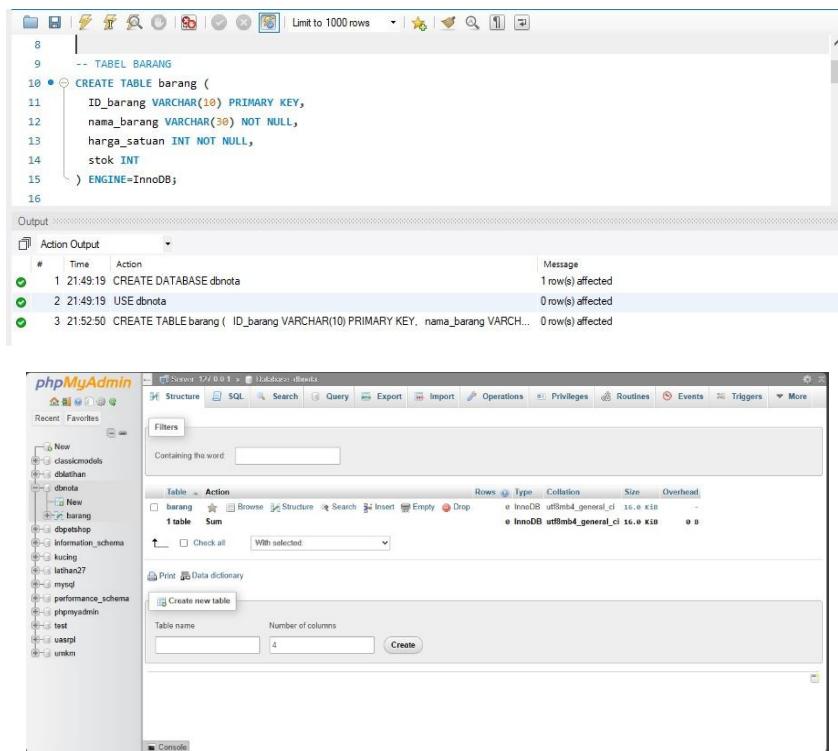


Pada tahap ini, rancangan Entity Relationship Diagram (ERD) yang telah dibuat diterapkan ke dalam Database Management System (DBMS) MySQL. Penerapan dilakukan dengan menggunakan MySQL Workbench sebagai alat untuk mengelola basis data.

Proses penerapan dimulai dengan pembuatan database dengan nama dbnota. Database ini digunakan sebagai tempat penyimpanan data pada sistem Nota Coffee Shop. Selanjutnya, setiap entitas yang terdapat pada ERD dibuat ke dalam bentuk tabel pada database, sesuai dengan atribut dan hubungan yang telah dirancang sebelumnya.

Tahap ini bertujuan agar rancangan ERD yang telah dibuat dapat digunakan secara langsung pada DBMS sebelum dilakukan proses pengisian dan pengolahan data.

### 3.3.2 Implementasi Data Definition Language (DDL)



The screenshot shows the phpMyAdmin interface. In the top-left pane, there is a code editor window containing the following SQL code:

```
8
9  -- TABEL BARANG
10  CREATE TABLE barang (
11      ID_barang VARCHAR(10) PRIMARY KEY,
12      nama_barang VARCHAR(30) NOT NULL,
13      harga_satuan INT NOT NULL,
14      stok INT
15  ) ENGINE=InnoDB;
16
```

In the bottom-left pane, there is an "Action Output" log table:

#	Time	Action	Message
1	21:49:19	CREATE DATABASE dbnota	1 row(s) affected
2	21:49:19	USE dbnota	0 row(s) affected
3	21:52:50	CREATE TABLE barang ( ID_barang VARCHAR(10) PRIMARY KEY, nama_barang VARCHAR...	0 row(s) affected

The main right-hand pane shows the database structure. The left sidebar lists databases: New, classicmodels, dbathan, dbnota, barang, dbeshop, information\_schema, kucing, latihan27, mysql, performance\_schema, phpmadmin, test, usapl, and unskm. The "barang" database is selected. The central area displays the "Structure" tab for the "barang" table, showing one table named "barang". Below the table structure, there is a "Create new table" form with "Table name" set to "barang" and "Number of columns" set to "4".

Pada tahap ini dilakukan implementasi Data Definition Language (DDL) untuk membuat struktur tabel pada basis data Nota Coffee Shop. Pembuatan tabel dilakukan berdasarkan rancangan Entity Relationship Diagram (ERD) yang telah dibuat pada tahap sebelumnya.

Setiap entitas pada ERD diterapkan ke dalam bentuk tabel dengan menentukan nama tabel, atribut, serta primary key dan foreign key sesuai dengan relasi yang ada. Proses pembuatan tabel dilakukan menggunakan perintah CREATE TABLE pada MySQL.

Sebagai contoh implementasi DDL, ditampilkan pembuatan tabel barang. Tabel ini digunakan untuk menyimpan data makanan dan minuman yang dijual pada coffee shop. Tabel barang memiliki ID\_Barang sebagai primary key, serta atribut nama barang, harga satuan, dan stok. Tabel lain pada sistem Nota Coffee Shop dibuat dengan cara yang sama dan disesuaikan dengan rancangan ERD.

### 3.3.3 Implementasi Data Manipulation Language (DML)

```

77
78 -- DML (DATA MANIPULATION LANGUAGE)
79 -- Menambah data
80 • INSERT INTO barang (ID_barang, nama_barang, harga_satuan, stok)
VALUES ('BR006', 'Coffee Latte', 15000, 5);
82

```

Action Output

#	Time	Action	Message
1	22:29:04	INSERT INTO barang (ID_barang, nama_barang, harga_satuan, stok) VALUES ('BR006', 'Coff...')	1 row(s) affected

Showing rows 0 - 4 (total, Query took 0.0004 seconds.)

SELECT \* FROM `barang`

Profil | Edit inline | Explain SQL | Create PHP code | Refresh

ID_barang	nama_barang	harga_satuan	stok	stock
ABK007	Chicken egg sambal matah	15000	5	NULL
CBK001	Coffee Latte	15000	5	NULL
CBK002	Ice coffee milk regular	11000	10	NULL
BR006	Ice coffee milk large	13000	8	NULL

```

89
90 -- Menampilkan data
91 • SELECT*FROM barang;
92

```

Result Grid

ID_barang	nama_barang	harga_satuan	stok	stock
ABK007	Chicken egg sambal matah	15000	5	NULL
CBK001	Ice coffee milk regular	11000	10	NULL
CBK002	Ice coffee milk large	13000	8	NULL
BR006	Ice coffee milk large	13000	8	NULL

Action Output

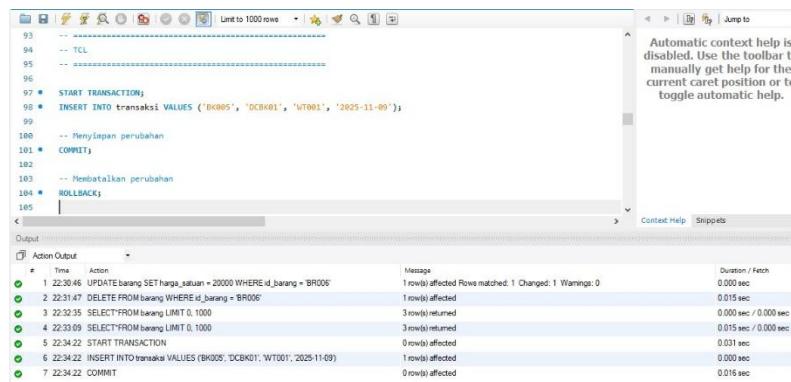
#	Time	Action	Message
1	22:30:46	UPDATE barang SET harga_satuan = 20000 WHERE id_barang = 'BR006'	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0
2	22:31:47	DELETE FROM barang WHERE id_barang = 'BR006'	1 row(s) affected
3	22:32:35	SELECT*FROM barang LIMIT 0, 1000	3 row(s) returned
4	22:33:09	SELECT*FROM barang LIMIT 0, 1000	3 row(s) returned

Pada tahap ini dilakukan implementasi Data Manipulation Language (DML) untuk mengelola data pada basis data Nota Coffee Shop. Perintah DML digunakan untuk menambahkan dan menampilkan data yang tersimpan pada tabel.

Sebagai contoh, pada tabel barang dilakukan penambahan data menggunakan perintah INSERT INTO untuk memasukkan data barang baru ke dalam database. Setelah data ditambahkan, perintah SELECT digunakan untuk menampilkan data barang yang tersimpan dan memastikan bahwa proses penyimpanan data telah berhasil dilakukan.

Implementasi DML ini menunjukkan bahwa data pada basis data dapat dikelola dan ditampilkan sesuai dengan kebutuhan sistem.

### 3.3.4 Implementasi Transaction Control Language (TCL)



The screenshot shows a MySQL Workbench interface. The SQL editor pane contains the following TCL script:

```
93 -- -----
94 -- TCL
95 -----
96
97 • START TRANSACTION;
98 • INSERT INTO transaksi VALUES ('BK005', 'DCBK01', 'WT001', '2025-11-09');
99
100 -- Menyimpan perubahan
101 • COMMIT;
102
103 -- Membatalkan perubahan
104 • ROLLBACK;
105
```

The Output pane shows the execution log:

Time	Action	Message	Duration / Fetch
1 22:30:46	UPDATE barang SET harga_satuan = 20000 WHERE id_barang = BR006	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
2 22:31:47	DELETE FROM barang WHERE id_barang = BR006	1 row(s) affected	0.015 sec
3 22:32:35	SELECT FROM barang LIMIT 0, 1000	3 row(s) returned	0.000 sec / 0.009 sec
4 22:33:09	SELECT FROM barang LIMIT 0, 1000	3 row(s) returned	0.015 sec / 0.009 sec
5 22:34:22	START TRANSACTION	0 row(s) affected	0.031 sec
6 22:34:24	INSERT INTO transaksi VALUES ('BK005', 'DCBK01', 'WT001', '2025-11-09')	1 row(s) affected	0.000 sec
7 22:34:24	COMMIT	0 row(s) affected	0.015 sec

Pada tahap ini dilakukan implementasi Transaction Control Language (TCL) untuk mengatur proses transaksi pada basis data Nota Coffee Shop. Perintah TCL digunakan untuk memastikan bahwa proses penyimpanan data transaksi dapat berjalan dengan aman dan terkontrol.

Pada implementasinya, perintah START TRANSACTION digunakan untuk memulai proses transaksi. Selanjutnya dilakukan perintah INSERT untuk menambahkan data transaksi ke dalam tabel transaksi. Setelah data berhasil dimasukkan, perintah COMMIT digunakan untuk menyimpan perubahan secara permanen ke dalam basis data.

Selain itu, perintah ROLLBACK disiapkan sebagai mekanisme pembatalan transaksi apabila terjadi kesalahan pada proses penyimpanan data. Dengan penggunaan TCL ini, ketepatan data transaksi dapat tetap terjaga.

## 3.4 Implementasi Query SQL

### 3.4.1 Query Agregasi dan Having

The screenshot shows two separate query windows in MySQL Workbench.

**Top Window (Aggregation):**

```
108 -- agregasi
109
110 • SELECT COUNT(*) FROM barang;
111 • SELECT SUM(harga_satuan) FROM barang;
112
113
114 •
115
116
117
```

**Output:**

#	Time	Action	Message	Duration / Fetch
1	22:37:11	SELECT COUNT(*) FROM barang LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
2	22:37:44	SELECT SUM(harga_satuan) FROM barang LIMIT 0, 1000	1 row(s) returned	0.016 sec / 0.000 sec

**Bottom Window (Having):**

```
113 -- HAVING
114 • SELECT nama_barang, COUNT(*)
115 FROM barang
116 GROUP BY nama_barang
117 HAVING COUNT(*) > 1;
```

**Output:**

#	Time	Action	Message	Duration / Fetch
1	22:37:11	SELECT SUM(harga_satuan) FROM barang LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
2	22:37:44	SELECT SUM(harga_satuan) FROM barang LIMIT 0, 1000	1 row(s) returned	0.016 sec / 0.000 sec
3	22:38:36	SELECT nama_barang, COUNT(*) FROM barang GROUP BY nama_barang HAVING COUNT... 0 row(s) returned		0.000 sec / 0.000 sec

Query agregasi digunakan untuk melakukan perhitungan terhadap data yang terdapat pada tabel dalam basis data. Pada sistem Nota Coffee Shop, query agregasi digunakan untuk memperoleh informasi ringkas dari data barang yang tersimpan.

Pada implementasinya, fungsi COUNT(\*) digunakan untuk menghitung jumlah data barang yang ada di dalam tabel barang, sedangkan fungsi SUM(harga\_satuan) digunakan untuk menghitung total harga satuan dari seluruh barang. Berdasarkan hasil eksekusi query, diperoleh total harga satuan barang sebesar 43000.

Selain itu, perintah HAVING digunakan untuk memberikan kondisi pada data hasil pengelompokan. Pada query ini, data barang dikelompokkan berdasarkan nama\_barang menggunakan GROUP BY, kemudian disaring dengan kondisi HAVING COUNT(\*) > 1.

Berdasarkan hasil eksekusi query, tidak terdapat data yang memenuhi kondisi HAVING. Meskipun demikian, query ini tetap menunjukkan penggunaan fungsi agregasi dan perintah HAVING dalam proses pengolahan data pada basis data.

### 3.4.2 Group By

```
118
119
120 --> GROUP BY
121
122 • SELECT nama_barang, SUM(harga_satuan)
123   FROM barang
124  GROUP BY nama_barang;
```

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

nama_barang	harga_satuan
Chicken eggs sanbeli matah	19000
[Ice coffee milk large	13000
[Ice coffee milk regular	13000

Result 12 x

Output

Action Output

Time	Action	Message	Duration / fetch
1 22:40:10	SELECT nama_barang, SUM(harga_satuan) FROM barang GROUP BY nama_barang LIMIT ...	3 rows returned	0.000 sec / 0.000 sec

Automatic context help is disabled. Use the toolbar manually get help for the current caret position or toggle automatic help.

Perintah GROUP BY digunakan untuk mengelompokkan data berdasarkan kolom tertentu sehingga data dapat ditampilkan dalam bentuk ringkas. Pada sistem Nota Coffee Shop, GROUP BY digunakan untuk mengelompokkan data barang berdasarkan nama\_barang.

Pada implementasinya, perintah GROUP BY dikombinasikan dengan fungsi agregasi SUM(harga\_satuan) untuk menghitung total harga satuan dari setiap jenis barang. Dengan menggunakan query ini, setiap barang akan ditampilkan satu kali beserta total harga satuannya.

Berdasarkan hasil eksekusi query, data barang berhasil dikelompokkan berdasarkan nama barang dan ditampilkan dalam bentuk ringkasan yang lebih terstruktur serta mudah dipahami.

### 3.4.3 Join

The screenshot displays four separate result grids from MySQL Workbench, each showing the output of a different JOIN query:

- Result 17:** Shows an INNER JOIN between layanan and waitress tables. The output is a single row: (Nia, Nia).
- Result 18:** Shows a SELECT query involving multiple tables (transaksi, layanan, detail\_transaksi, detail\_transaksi\_layanan) using various JOIN clauses. The output is a single row: (Nia, Nia, 07001, BK002).
- Result 21:** Shows a LEFT JOIN between transaksi and layanan tables. The output is two rows: (BK003, Dine In, Nia) and (BK005, Dine In, Nia).
- Result 22:** Shows a RIGHT JOIN between transaksi and layanan tables. The output is two rows: (BK003, Nia, Dine In) and (BK005, Nia, Dine In).

Perintah JOIN digunakan untuk menggabungkan data dari dua atau lebih tabel yang saling berelasi. Pada sistem Nota Coffee Shop, JOIN digunakan untuk menampilkan data transaksi secara lengkap berdasarkan hubungan antar tabel yang telah dirancang pada ERD.

a. Inner Join

Inner Join digunakan untuk menampilkan data yang memiliki kecocokan pada kedua tabel yang digabungkan. Pada implementasinya, Inner Join digunakan untuk menggabungkan tabel transaksi, layanan, dan waitress berdasarkan atribut foreign key yang sesuai. Hasil dari query ini menampilkan informasi nama layanan dan nama waitress pada setiap transaksi yang tercatat di dalam sistem.

b. Inner Join Lebih Dari Dua Tabel

Selain menggabungkan dua tabel, Inner Join juga dapat digunakan untuk menggabungkan lebih dari dua tabel sekaligus. Pada query ini, tabel transaksi, waitress, layanan, dan detail\_transaksi digabungkan untuk menampilkan data transaksi secara lebih detail. Hasil query menampilkan informasi nama waitress, nama layanan, nomor detail transaksi, dan ID transaksi. Query ini menunjukkan penerapan relasi antar tabel sesuai dengan rancangan ERD.

c. Left Join

Left Join digunakan untuk menampilkan seluruh data dari tabel sebelah kiri, meskipun tidak memiliki pasangan data pada tabel sebelah kanan. Pada implementasinya, Left Join digunakan untuk menampilkan seluruh data transaksi beserta data layanan dan waitress yang terkait. Hasil query menunjukkan bahwa semua data transaksi tetap ditampilkan, meskipun terdapat kemungkinan data pada tabel lain tidak lengkap.

d. Right Join

Right Join digunakan untuk menampilkan seluruh data dari tabel sebelah kanan, meskipun tidak memiliki pasangan data pada tabel sebelah kiri. Pada query ini, Right Join digunakan untuk menampilkan data layanan beserta data transaksi dan waitress yang terkait. Query ini digunakan untuk memahami perbedaan hasil yang ditampilkan antara Left Join dan Right Join.

## **BAB IV**

### **PENUTUP**

#### **4.1 Ringkasan Query dan Hasil Pengujian**

Pada tahap akhir ini dilakukan pengujian terhadap query-query yang telah dibuat pada sistem basis data Nota Coffee Shop. Pengujian meliputi penggunaan query Data Definition Language (DDL), Data Manipulation Language (DML), Transaction Control Language (TCL), serta query lanjutan seperti JOIN, GROUP BY, fungsi agregasi, dan HAVING.

Berdasarkan hasil pengujian, seluruh query dapat dijalankan dengan baik menggunakan MySQL Workbench dan phpMyAdmin. Query JOIN berhasil menampilkan data dari beberapa tabel yang saling berelasi sesuai dengan rancangan Entity Relationship Diagram (ERD). Selain itu, fungsi agregasi dan GROUP BY mampu menampilkan data dalam bentuk ringkasan, sedangkan perintah HAVING digunakan untuk melakukan penyaringan data hasil pengelompokan. Implementasi TCL juga berjalan dengan baik dalam mengatur proses transaksi pada basis data.

Hasil pengujian menunjukkan bahwa struktur basis data dan query yang dibuat telah sesuai dengan kebutuhan sistem Nota Coffee Shop.

#### **4.2 Kendala dan Perbaikan**

Selama proses perancangan dan implementasi basis data, terdapat beberapa kendala yang ditemui. Kendala tersebut antara lain kesalahan penulisan query SQL, hasil query yang tidak sesuai dengan yang diharapkan, serta penyesuaian relasi antar tabel agar sesuai dengan rancangan ERD.

Kendala-kendala tersebut dapat diatasi dengan melakukan pengecekan ulang struktur tabel, memastikan penggunaan primary key dan foreign key sudah benar, serta memperbaiki sintaks query SQL sesuai dengan aturan MySQL. Dengan melakukan perbaikan tersebut, sistem basis data dapat berjalan dengan lebih baik dan menghasilkan data yang sesuai.

### **4.3 Kesimpulan**

Berdasarkan hasil perancangan dan implementasi basis data Nota Coffee Shop, dapat disimpulkan bahwa sistem basis data berhasil dibuat sesuai dengan konsep basis data relasional. Proses normalisasi hingga bentuk normal ketiga (3NF) membantu dalam menjaga konsistensi data.

Penggunaan ERD mempermudah dalam perancangan struktur tabel dan relasi antar tabel. Implementasi SQL yang mencakup DDL, DML, TCL, serta query lanjutan seperti JOIN, GROUP BY, agregasi, dan HAVING memungkinkan pengelolaan data transaksi dilakukan secara terstruktur dan efisien.

### **4.4 Saran Pengembangan**

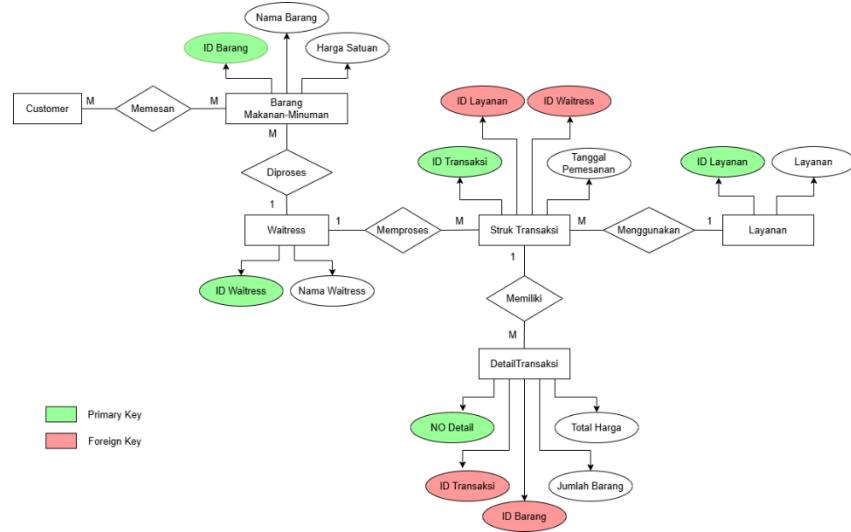
Untuk pengembangan selanjutnya, sistem basis data Nota Coffee Shop ini dapat dikembangkan dengan menambahkan antarmuka aplikasi berbasis web atau desktop agar lebih mudah digunakan oleh pengguna. Selain itu, sistem keamanan basis data dan pengelolaan hak akses pengguna juga dapat ditingkatkan agar data lebih terjamin keamanannya.

Pengembangan fitur laporan penjualan dan analisis data transaksi juga dapat ditambahkan agar sistem dapat memberikan informasi yang lebih lengkap dan bermanfaat.

## LAMPIRAN

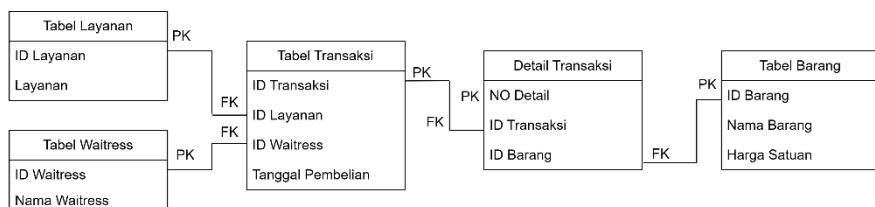
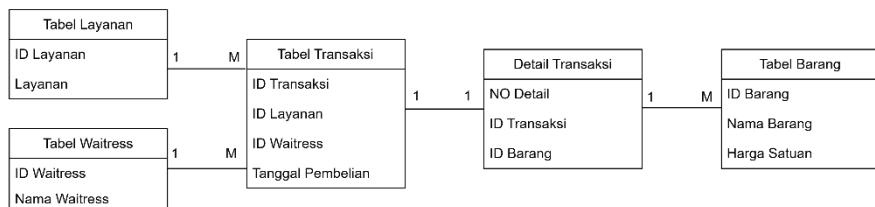
**A. Link Repository Github: <https://github.com/Riskidewanti/riskidewanti>**

**B. Entity Relationship Diagram (ERD)**

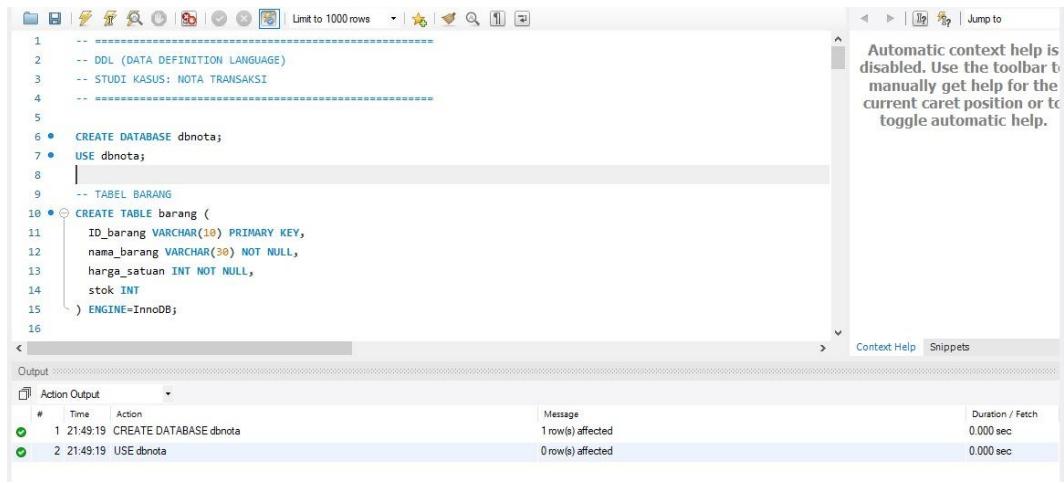


**C. Relasi Antar Tabel**

RELASI ANTAR TABEL



## D. Implementasi DDL

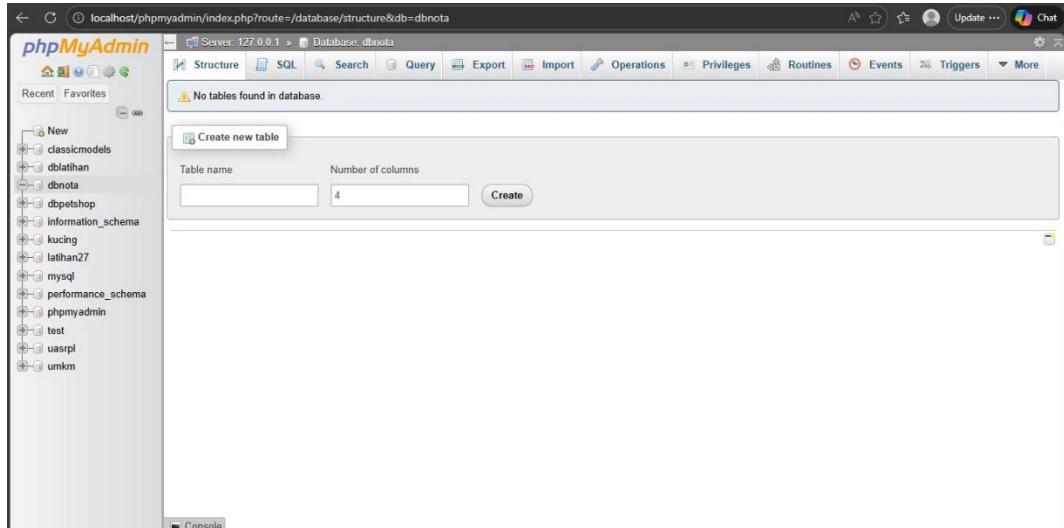


The screenshot shows the MySQL Workbench interface. In the SQL tab, the following DDL code is written:

```
1 -- =====
2 -- DDL (DATA DEFINITION LANGUAGE)
3 -- STUDI KASUS: NOTA TRANSAKSI
4 --
5
6 • CREATE DATABASE dbnota;
7 • USE dbnota;
8
9 -- TABEL BARANG
10 • CREATE TABLE barang (
11     ID_barang VARCHAR(10) PRIMARY KEY,
12     nama_barang VARCHAR(30) NOT NULL,
13     harga_satuan INT NOT NULL,
14     stok INT
15 ) ENGINE=InnoDB;
```

The Output tab shows the execution results:

#	Time	Action	Message	Duration / Fetch
1	21:49:19	CREATE DATABASE dbnota	1 row(s) affected	0.000 sec
2	21:49:19	USE dbnota	0 row(s) affected	0.000 sec

The screenshot shows the phpMyAdmin interface for the 'dbnota' database. The left sidebar lists databases: New, classicmodels, dblatihan, dbnota, dbpetshop, information\_schema, kucing, latihan27, mysql, performance\_schema, phpmyadmin, test, uaspl, and umkm. The main area shows a 'Create new table' dialog with 'Table name' and 'Number of columns' fields set to '4'. Below the dialog is a message: 'No tables found in database.'

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Context Help Snippets

```

21
22 -- TABEL LAYANAN
23 • CREATE TABLE layanan (
24     ID_layanan VARCHAR(10) PRIMARY KEY,
25     nama_layanan VARCHAR(30) NOT NULL
26 ) ENGINE=InnoDB;
27

```

**Output:**

#	Time	Action	Message	Duration / Fetch
1	21:49:19	CREATE DATABASE dbnota	1 row(s) affected	0.000 sec
2	21:49:19	USE dbnota	0 row(s) affected	0.000 sec
3	21:52:50	CREATE TABLE barang ( ID_barang VARCHAR(10) PRIMARY KEY, nama_barang VARCH... 0 row(s) affected		0.015 sec
4	21:54:56	INSERT INTO barang VALUES ('CBK001','ice coffee milk regular',11000,10), ('CBK002','ice cof... 3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0		0.015 sec
5	21:56:19	CREATE TABLE layanan ( ID_layanan VARCHAR(10) PRIMARY KEY, nama_layanan VARC... 0 row(s) affected		0.015 sec

**phpMyAdmin**

Server: 127.0.0.1 > Database: dbnota > Table: layanan

Table structure

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	ID_layanan	varchar(10)	utf8mb4_general_ci		No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>
2	nama_layanan	varchar(30)	utf8mb4_general_ci		No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>

Add to central columns Remove from central columns

Print [Propose table structure](#) [Track table](#) [Move columns](#) [Normalize](#)

Add 1 column(s) after nama\_layanan [Go](#)

Indexes

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
<a href="#">Edit</a> <a href="#">Rename</a> <a href="#">Drop</a>	PRIMARY	BTREE	Yes	No	ID_layanan	0	A	No	

Create an index on 1 columns [Go](#)

Partitions

No partitioning defined!

## E. Implementasi DML

Output

```

68
69 -- DML (DATA MANIPULATION LANGUAGE)
70
71 -- mengubah struktur tabel
72 • ALTER TABLE barang ADD stock INT
73

```

Action Output

#	Time	Action
1	22:11:25	ALTER TABLE barang ADD stock INT

Message

0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

Duration / Fetch  
0.016 sec

phpMyAdmin

Recent Favorites

Table structure

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	ID_barang	varchar(10)	utf8mb4_general_ci		No	None			Change  Drop More
2	nama_barang	varchar(30)	utf8mb4_general_ci		No	None			Change  Drop More
3	harga_satuan	int(11)			No	None			Change  Drop More
4	stok	int(11)			Yes	NULL			Change  Drop More
5	stock	int(11)			Yes	NULL			Change  Drop More

Add 1 column(s) after stock

Indexes

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit		Drop PRIMARY	BTREE	Yes	ID_barang	3	A	No	

Create an index on 1 columns Go

Output

```

77
78 -- DML (DATA MANIPULATION LANGUAGE)
79 -- Menambah data
80 • INSERT INTO barang (ID_barang, nama_barang, harga_satuan, stok)
81 VALUES ('BR006', 'Coffee Latte', 15000, 5);
82

```

Action Output

#	Time	Action
1	22:29:04	INSERT INTO barang (ID_barang, nama_barang, harga_satuan, stok) VALUES ('BR006', 'Coff... 1 row(s) affected

Message

Duration / Fetch  
0.016 sec

phpMyAdmin

Recent Favorites

Show rows 0 - 4 (5 total, Query took 0.0004 seconds)

SELECT \* FROM `barang`

Profile [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

Show all Number of rows: 25 Filter rows: Search this table Sort by key: None

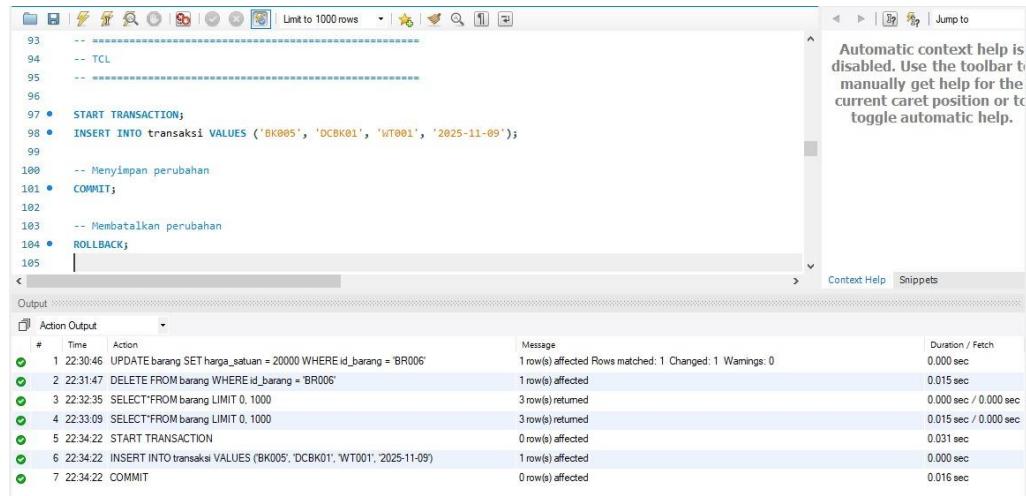
Extra options

ID_barang	nama_barang	harga_satuan	stok	stock
ABK007	Chicken egg sambal matah	19000	5	NULL
BR006	Coffee Latte	15000	5	NULL
CBK001	Ice coffee milk regular	11000	10	NULL
CBK002	Ice coffee milk large	13000	8	NULL

Query results operations

Print Copy to clipboard Export Display chart Create view

## F. Implementasi TCL



The screenshot shows a MySQL Workbench interface. The top pane displays a script editor with the following content:

```
93 -- -----
94 -- TCL
95 -----
96
97 • START TRANSACTION;
98 • INSERT INTO transaksi VALUES ('BK005', 'DCBK01', 'WT001', '2025-11-09');
99
100 -- Menyimpan perubahan
101 • COMMIT;
102
103 -- Membatalkan perubahan
104 • ROLLBACK;
105
```

The right side of the interface has a note: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or toggle automatic help." Below the script editor is an "Output" pane titled "Action Output". It contains a table with the following data:

#	Time	Action	Message	Duration / Fetch
1	22:30:46	UPDATE barang SET harga_satuan = 20000 WHERE id_barang = 'BR006'	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
2	22:31:47	DELETE FROM barang WHERE id_barang = 'BR006'	1 row(s) affected	0.015 sec
3	22:32:35	SELECT * FROM barang LIMIT 0, 1000	3 row(s) returned	0.000 sec / 0.000 sec
4	22:33:09	SELECT * FROM barang LIMIT 0, 1000	3 row(s) returned	0.015 sec / 0.000 sec
5	22:34:22	START TRANSACTION	0 row(s) affected	0.031 sec
6	22:34:22	INSERT INTO transaksi VALUES ('BK005', 'DCBK01', 'WT001', '2025-11-09')	1 row(s) affected	0.000 sec
7	22:34:22	COMMIT	0 row(s) affected	0.016 sec

## G. Query Agregasi dan Having

106 -- =====  
107 -- Agregasi & HAVING  
108 -- =====  
109 -- agregasi  
110 • SELECT COUNT(\*) FROM barang;  
111 • SELECT SUM(harga\_satuan) FROM barang;

Result Grid | Filter Rows: Export: Wrap Cell Content: Result Grid Context Help Snippets

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	22:36:12	SELECT COUNT() FROM barang LIMIT 0, 1000	1 row(s) returned	0.016 sec / 0.000 sec

108 -- =====  
109 -- agregasi  
110 • SELECT COUNT(\*) FROM barang;  
111 • SELECT SUM(harga\_satuan) FROM barang;  
112

Result Grid | Filter Rows: Export: Wrap Cell Content: Result Grid Context Help Snippets

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	22:37:11	SELECT SUM(harga_satuan) FROM barang LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
2	22:37:44	SELECT SUM(harga_satuan) FROM barang LIMIT 0, 1000	1 row(s) returned	0.016 sec / 0.000 sec

113 -- HAVING  
114 • SELECT nama\_barang, COUNT(\*)  
115 FROM barang  
116 GROUP BY nama\_barang  
117 HAVING COUNT(\*) > 1;

Result Grid | Filter Rows: Export: Wrap Cell Content: Result Grid Form Editor Context Help Snippets

Result 9 x

Action Output

#	Time	Action	Message	Duration / Fetch
1	22:37:11	SELECT SUM(harga_satuan) FROM barang LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
2	22:37:44	SELECT SUM(harga_satuan) FROM barang LIMIT 0, 1000	1 row(s) returned	0.016 sec / 0.000 sec
3	22:38:36	SELECT nama_barang, COUNT(*) FROM barang GROUP BY nama_barang HAVING COUNT... 0 row(s) returned		0.000 sec / 0.000 sec

## H. Query Group By

The screenshot shows a MySQL Workbench interface with a query editor and a results grid. The query is:

```

118
119 -- =====
120 -- GROUP BY
121 -- =====
122 • SELECT nama_barang, SUM(harga_satuan)
123   FROM barang
124   GROUP BY nama_barang;
  
```

The results grid displays the following data:

nama_barang	SUM(harga_satuan)
Chicken egg sambal matah	19000
Ice coffee milk large	13000
Ice coffee milk regular	11000

The output pane shows the query was executed at 22:40:10 and returned 3 rows.

## I. Query Join

The screenshot shows three separate MySQL Workbench sessions demonstrating different types of joins.

**Session 1 (Top): Natural Join**

```

125
126 -- =====
127 -- JOIN
128 -- =====
129 -- natural join
130 • SELECT layanan.nama_layanan, waitress.nama_waitress
131   FROM transaksi
132   JOIN layanan ON transaksi.ID_Layanan = layanan.ID_layanan
133   JOIN waitress ON transaksi.ID_Waitress = waitress.ID_Waitress;
  
```

The results grid shows:

nama_layanan	nama_waitress
Dine In	Nisa
Dine In	Nisa

**Session 2 (Middle): Join with ON Clause**

```

135
136 -- join dengan using
137 • SELECT layanan, nama_waitress
138   FROM layanan JOIN waitress USING(nama_waitress);
  
```

The results grid shows:

nama_layanan	nama_waitress
Dine In	Nisa
Dine In	Nisa

**Session 3 (Bottom): Join with ON Clause**

```

139
140 -- inner join... ON
141 • SELECT layanan.nama_layanan, waitress.nama_waitress
142   FROM transaksi
143   JOIN layanan
144     ON transaksi.ID_Layanan = layanan.ID_layanan
145   JOIN waitress
146     ON transaksi.ID_Waitress = waitress.ID_Waitress;
  
```

The results grid shows:

nama_layanan	nama_waitress
Dine In	Nisa
Dine In	Nisa

148 -- inner join lebih dari 2 tabel

```

149 • SELECT
150     waitress.nama_waitress,
151     layanan.nama_layanan,
152     detail_transaksi.NO_detail,
153     transaksi.ID_Transaksi
154     FROM transaksi
155     JOIN waitress
156     ON transaksi.ID_Waitress = waitress.ID_Waitress
157     JOIN layanan
158     ON transaksi.ID_Layanan = layanan.ID_layanan
159     JOIN detail_transaksi
160     ON transaksi.ID_Transaksi = detail_transaksi.ID_Transaksi;
161

```

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

nama_waitress	nama_layanan	NO_detail	ID_Transaksi
Nisa	Dine In	DT001	BK003

Action Output

#	Time	Action	Message	Duration / Fetch
1	22:49:08	SELECT waitress.nama_waitress, layanan.nama_layanan, detail_transaksi.NO_detail, ...	1 row(s) returned	0.031 sec / 0.000 sec

161
162 -- equijoin
163 • SELECT
164 transaksi.ID\_Transaksi,
165 layanan.nama\_layanan,
166 waitress.nama\_waitress
167 FROM transaksi, layanan, waitress
168 WHERE transaksi.ID\_Layanan = layanan.ID\_layanan
169 AND transaksi.ID\_Waitress = waitress.ID\_Waitress;
170

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

ID_Transaksi	nama_layanan	nama_waitress
BK003	Dine In	Nisa
BK005	Dine In	Nisa

Action Output

#	Time	Action	Message	Duration / Fetch
1	22:49:08	SELECT waitress.nama_waitress, layanan.nama_layanan, detail_transaksi.NO_detail, ...	1 row(s) returned	0.031 sec / 0.000 sec
2	22:49:48	SELECT transaksi.ID_Transaksi, layanan.nama_layanan, waitress.nama_waitress FRO...	2 row(s) returned	0.015 sec / 0.000 sec

171 -- left join
172 • SELECT
173 transaksi.ID\_Transaksi,
174 layanan.nama\_layanan,
175 waitress.nama\_waitress
176 FROM transaksi
177 LEFT JOIN layanan
178 ON transaksi.ID\_Layanan = layanan.ID\_layanan
179 LEFT JOIN waitress
180 ON transaksi.ID\_Waitress = waitress.ID\_Waitress;
181

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

ID_Transaksi	nama_layanan	nama_waitress
BK003	Dine In	Nisa
BK005	Dine In	Nisa

Action Output

#	Time	Action	Message	Duration / Fetch
1	22:50:44	SELECT transaksi.ID_Transaksi, layanan.nama_layanan, waitress.nama_waitress FRO...	2 row(s) returned	0.016 sec / 0.000 sec

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

```

182 -- right join
183 • SELECT
184     transaksi.ID_Transaksi,
185     layanan.nama_layanan,
186     waitress.nama_waitress
187     FROM transaksi
188     RIGHT JOIN layanan
189         ON transaksi.ID_Layanan = layanan.ID_layanan
190     LEFT JOIN waitress
191         ON transaksi.ID_Waitress = waitress.ID_Waitress;
192

```

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

ID_Transaksi	nama_layanan	nama_waitress
BK003	Dine In	Nisa
BK005	Dine In	Nisa

Action Output

#	Time	Action	Message	Duration / Fetch
1	22:51:28	SELECT transaksi.ID_Transaksi, layanan.nama_layanan, waitress.nama_waitress FRO...	2 row(s) returned	0.015 sec / 0.000 sec

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

```

193 -- full join
194 • SELECT
195     waitress.ID_Waitress,
196     waitress.nama_waitress,
197     transaksi.ID_Transaksi,
198     transaksi.Tanggal_pembelian
199     FROM waitress
200     LEFT JOIN transaksi
201         ON waitress.ID_Waitress = transaksi.ID_Waitress
202     WHERE waitress.nama_waitress LIKE '%a%';
203

```

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

ID_Waitress	nama_waitress	ID_Transaksi	Tanggal_pembelian
WT001	Nisa	BK003	2025-11-09
WT001	Nisa	BK005	2025-11-09

Action Output

#	Time	Action	Message	Duration / Fetch
1	22:52:01	SELECT waitress.ID_Waitress, waitress.nama_waitress, transaksi.ID_Transaksi, tra...	2 row(s) returned	0.016 sec / 0.000 sec

## **DAFTAR PUSTAKA**

1. Irawan RD. Pengantar basis data. Bahan ajar mata kuliah Pemrograman Basis Data. Surakarta: Program Studi Teknik Informatika, Universitas Duta Bangsa; 2025.
2. Irawan RD. Relasi tabel dan entity relationship diagram (ERD). Bahan ajar mata kuliah Pemrograman Basis Data. Surakarta: Program Studi Teknik Informatika, Universitas Duta Bangsa; 2025.
3. Irawan RD. Normalisasi basis data. Bahan ajar mata kuliah Pemrograman Basis Data. Surakarta: Program Studi Teknik Informatika, Universitas Duta Bangsa; 2025.
4. Irawan RD. Structured query language (SQL): DDL, DML, dan TCL. Bahan ajar mata kuliah Pemrograman Basis Data. Surakarta: Program Studi Teknik Informatika, Universitas Duta Bangsa; 2025.
5. Irawan RD. Implementasi query SQL: agregasi, group by, having, dan join. Bahan ajar mata kuliah Pemrograman Basis Data. Surakarta: Program Studi Teknik Informatika, Universitas Duta Bangsa; 2025.