“

SISTEM BASIS DATA

Laporan Project UAS

Dosen : Ridwan Dwi Irawan, M. Kom.

“

Disusun Oleh :

Nama : Aditya Firmansyah | 250119001

Adnan Muhammad Fihr | 250119002

Muhammad Nur Hasan Majid | 250119017

Universitas Duta Bangsa Surakarta

Fakultas Ilmu Komputer – Prodi Teknologi Rekayasa Perangkat Lunak

2026

DAFTAR ISI

BAB 1: PENDAHULUAN

1.1 Latar Belakang Project

Pengelolaan data barang dan transaksi pada sektor retail, khususnya perabotan rumah tangga seperti produk plastik dan wadah makanan, sering kali menghadapi kendala dalam akurasi stok dan kecepatan pelaporan. Berdasarkan studi kasus dalam **"Project UAS"**,. sistem manual yang ada saat ini menyulitkan pemantauan arus barang dari berbagai vendor ke gudang, serta pencatatan transaksi yang melibatkan banyak peran seperti kasir dan sales.

Ketidakteraturan ini berpotensi menyebabkan kerugian akibat stok mati, kehabisan atau kesalahan input data transaksi. Oleh karena itu, diperlukan sebuah sistem basis data terintegrasi yang mampu menangani manajemen inventori inii, data pelanggan, dan histori transaksi secara sistematis menggunakan teknologi DBMS (Database Management System).

1.2 Pengantar Pemrograman Basis Data

Pemrograman basis data merupakan Pondasi utama dalam pengembangan aplikasi modern yang memerlukan penyimpanan data persisten. Materi ini mencakup kemampuan untuk merancang skema yang efisien melalui teknik normalisasi, serta memanipulasi data menggunakan bahasa SQL (*Structured Query Language*).

Dalam proyek ini, pemrograman basis data diaplikasikan melalui penggunaan MySQL sebagai DBMS utama, di mana logika bisnis seperti relasi antar tabel, integritas data melalui *constraint*, dan otomasi laporan dapat diimplementasikan langsung pada level basis data ini . Penguasaan terhadap DDL, DML, hingga TCL menjadi krusial untuk memastikan sistem yang dibangun memiliki kinerja tinggi dan konsistensi data yang baik.

1.3 Tujuan Project

Berdasarkan kebutuhan teknis digitalisasi yang lebih tertata, tujuan dari proyek ini ialah tak lain dan tak bukan meliputi :

- Merancang ERD (*Entity Relationship Diagram*) yang mampu memetakan hubungan kompleks antara produk, vendor, dan transaksi penjualan.
- Mengimplementasikan teknik normalisasi 1NF hingga 3NF untuk meminimalkan redundansi data pada tabel master barang dan vendor.
- Menerapkan perintah SQL (DDL dan DML) untuk membangun struktur database dan mengelola data operasional secara efektif.
- Menghasilkan laporan informasi yang akurat melalui penggunaan query JOIN, Agregasi, dan GROUP BY untuk mendukung pengambilan keputusan manajemen.

- Menjamin keamanan dan integritas transaksi menggunakan mekanisme TCL (*Transaction Control Language*)

1.4 Gambaran Umum Sistem

Sistem basis data **"Project UAS"** bekerja dengan mengintegrasikan tujuh tabel utama yang saling berelasi. Alur sistem dimulai dari pendataan **Jenis Barang** dan **Barang** yang dipasok oleh **Vendor**. Saat terjadi transaksi, sistem akan merekam data dari tabel **Pelanggan**, **Kasir**, dan **Sales** ke dalam tabel **Transaksi**. Keunggulan sistem ini terletak pada kemampuannya menyajikan *view* laporan gabungan antara barang dan vendor secara instan, serta perlindungan data melalui sistem *rollback* jika terjadi kesalahan saat proses pembaruan stok berlangsung.

BAB 2: LANDASAN TEORI

2.1 Konsep Basis Data dan DBMS

Basis data adalah kumpulan data yang terorganisir secara logis dan saling berelasi sehingga memudahkan dalam penyimpanan, manipulasi, dan pemanggilan kembali data. Dalam proyek ini, digunakan *Database Management System* (DBMS) berupa **MySQL**. **MySQL** dipilih karena merupakan sistem manajemen basis data relasional (RDBMS) yang menggunakan *Structured Query Language* (SQL) untuk mengelola data operasional toko secara efisien dan mendukung integritas data yang tinggi.

2.2 Pengenalan Relasi Tabel dan Basis Data

Relasi tabel adalah hubungan yang terjalin antara satu tabel dengan tabel lainnya dalam sebuah basis data. Dalam sistem **"Project UAS"** ini, data tidak disimpan dalam satu tabel besar, melainkan dipecah ke dalam tabel-tabel spesifik seperti barang, jenisBarang, dan vendor. Tujuannya adalah untuk menghindari **redundansi** (pengulangan data) dan memastikan bahwa setiap entitas memiliki atribut yang sesuai dengan fungsinya.

2.3 Entity Relationship Diagram (ERD)

ERD adalah model konsep yang menggambarkan hubungan antar entitas dalam sistem. Pada proyek ini, entitas utama meliputi:

- **Entitas Master**: barang, jenisBarang, kasir, sales, pelanggan, dan vendor.

- **Entitas Transaksi**: transaksi dan detailBarang. ERD memvisualisasiikan bagaimana seorang kasir melayani banyak transaksi (*One-to-Many*) dan bagaimana satu vendor dapat memasok berbagai macam barang.

2.4 Normalisasi Data (1NF, 2NF, 3NF)

Normalisasi dilakukan untuk memastikan struktur database lebih efisien dan bebas dari anomali:

- **First Normal Form (1NF)**: Setiap kolom dalam tabel pelanggan dan kasir hanya berisi satu nilai (atomik). Tidak ada grup berulang pada nomor telepon atau nama.
- **Second Normal Form (2NF)**: Memastikan semua atribut non-key bergantung sepenuhnya pada Primary Key. Contoh: Atribut namaJenisBarang dipisahkan dari tabel barang ke tabel jenisBarang agar tidak terjadi pengulangan kategori pada setiap item produk.
- **Third Normal Form (3NF)**: Menghilangkan ketergantungan transitif. Informasi alamatVendor diletakkan di tabel vendor sehingga tidak bergantung secara tidak langsung pada idBarang di tabel utama.

2.5 Perancangan ERD Final dan Relasi Antar Tabel

Berdasarkan file SQL yang diimplementasikan, relasi final sistem adalah sebagai berikut:

1. **Tabel barang** merujuk ke **jenisBarang** melalui idjenisBarang.
2. **Tabel vendor** merujuk ke **barang** dan **jenisBarang** untuk mencatat asal pasokan produk.
3. **Tabel transaksi** menjadi pusat relasi yang menghubungkan idKasir, idSales, dan idPelanggan.
4. **Tabel detailBarang** menghubungkan idBarang dengan noTransaksi, yang memungkinkan satu nota transaksi berisi banyak item barang.

Nb : gambar ERD nya akan kami cantumkan ke dalam bab lampiran…

2.6 Cakupan Primary Key (PK) dan Foreign Key (FK)

- **Primary Key (PK)**: Atribut unik yang menjadi identitas utama sebuah baris dalam tabel.
    - *Contoh dalam project kami*: idBarang pada tabel barang dan noTransaksi pada tabel transaksi.
- **Foreign Key (FK)**: Atribut yang merujuk pada Primary Key di tabel lain untuk menciptakan hubungan/relasi.
    - *Contoh dalam project kami*: Kolom idKasir pada tabel transaksi adalah FK yang merujuk pada idKasir di tabel kasir. FK memastikan bahwa tidak ada transaksi yang dicatat tanpa ada kasir yang terdaftar secara sah di sistem (Integritas Referensial).

BAB 3: PERANCANGAN DAN IMPLEMENTASI

3.1 Kebutuhan Sistem

Sistem ini dirancang untuk mengelola data operasional pada toko retail wadah plastik. Kebutuhan utama mencakup penyimpanan data master barang, manajemen hubungan dengan vendor, serta pencatatan transaksi penjualan yang melibatkan kasir, sales, dan pelanggan secara terintegrasi.

3.2 Implementasi DDL (Data Definition Language)

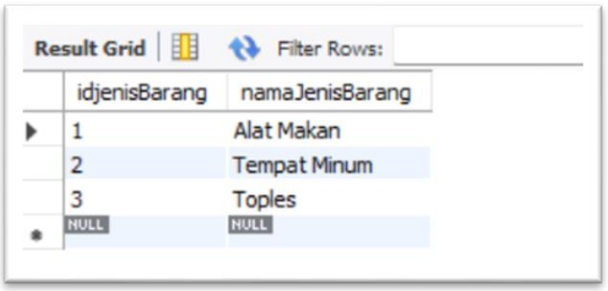
Pada tahap ini, dilakukan pembuatan struktur database dan tabel menggunakan perintah CREATE. Setiap tabel dilengkapi dengan *constraint* PRIMARY KEY dan FOREIGN KEY untuk menjaga integritas data.

- **Pembuatan Database dan Tabel Master**

Database diberi nama UAS dan tabel master dibuat untuk menyimpan entitas yang bersifat tetap.

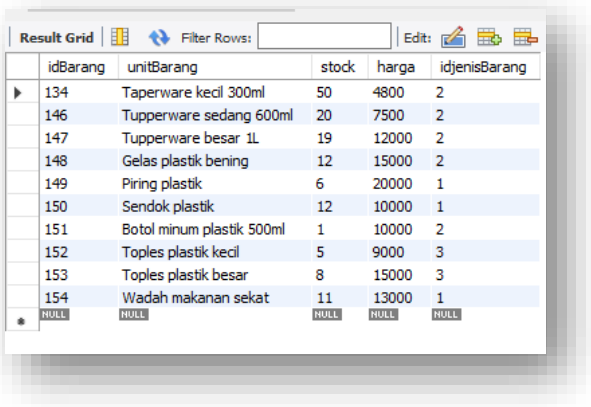
SQL

```
CREATE DATABASE IF NOT EXISTS UAS;
USE UAS;
CREATE TABLE idjenisBarang (
   idjenisBarang INT PRIMARY KEY,
   namaJenisBarang VARCHAR(50)
);
select * from idjenisBarang;
```




```
CREATE TABLE barang (
   idBarang INT PRIMARY KEY,
   unitBarang VARCHAR(100),
   stock INT(11),
   harga INT(11),
   idjenisBarang INT(11),
   FOREIGN KEY (idjenisBarang) REFERENCES jenisBarang(idjenisBarang)
);
```

select * from barang;



idBarang	unitBarang	stock	harga	idjenisBarang
134	Taperware kecil 300ml	50	4800	2
146	Tupperware sedang 600ml	20	7500	2
147	Tupperware besar 1L	19	12000	2
148	Gelas plastik bening	12	15000	2
149	Piring plastik	6	20000	1
150	Sendok plastik	12	10000	1
151	Botol minum plastik 500ml	1	10000	2
152	Toples plastik kecil	5	9000	3
153	Toples plastik besar	8	15000	3
154	Wadah makanan sekat	11	13000	1
NULL	NULL	NULL	NULL	NULL

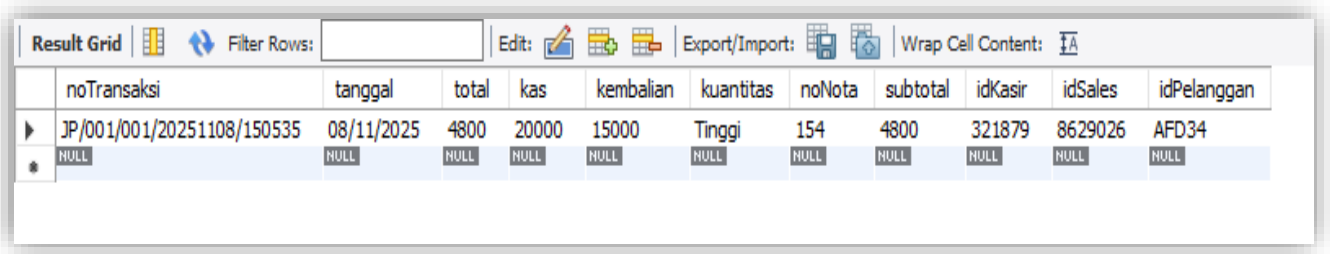
- **Penerapan Constraint dan Relasi Transaksi**

Tabel transaksi menghubungkan berbagai entitas master untuk mencatat aktivitas penjualan.

SQL

```
CREATE TABLE transaksi (
   noTransaksi VARCHAR(50) PRIMARY KEY,
   tanggal DATE,
   total INT(11),
   idKasir INT(11),
   idSales INT(11),
   idPelanggan VARCHAR(10),
   FOREIGN KEY (idKasir) REFERENCES kasir(idKasir),
   FOREIGN KEY (idSales) REFERENCES sales(idSales),
   FOREIGN KEY (idPelanggan) REFERENCES pelanggan(idPelanggan)
);
```

select * from transaksi;



noTransaksi	tanggal	total	kas	kembalian	kuantitas	noNota	subtotal	idKasir	idSales	idPelanggan
JP/001/001/20251108/150535	08/11/2025	4800	20000	15000	Tinggi	154	4800	321879	8629026	AFD34
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

3.3 Implementasi DML (Data Manipulation Language)

Implementasi DML berfokus pada pengisian data awal (*seed data*) ke dalam tabel yang telah dibuat agar sistem dapat diuji.

- **Pengisian Data Master**

Data dimasukkan ke tabel jenisBarang, barang, dan vendor untuk mensimulasikan ketersediaan stok.

- **Contoh**: Memasukkan produk "Tupperware" ke dalam kategori "Tempat Minum".
- **Contoh**: Mendata vendor "CV. JAYA PLASTIK" yang berlokasi di Bandung.

- **Manipulasi Data Transaksi**

Pencatatan transaksi dilakukan dengan menyertakan nilai total, jumlah bayar (kas), dan kembalian.

- **Input**: Mencatat nota JP/001/001/20251108/150535 dengan total Rp4.800.

3.4 Implementasi Query Penting

Bagian ini menunjukkan kemampuan sistem dalam mengolah data menjadi informasi melalui berbagai teknik query.

- **Query Agregasi dan HAVING**

Digunakan untuk menghitung total stok barang per kategori yang jumlahnya di atas 10 unit.

SQL

```
SELECT j.namaJenisBarang, SUM(b.stock) AS total_stok
FROM barang b
JOIN jenisBarang j ON b.idjenisBarang = j.idjenisBarang
GROUP BY j.namaJenisBarang
HAVING SUM(b.stock) > 10;
```

Result Grid | Filter Rows:

	namaJenisBarang	total_stok
▶	Alat Makan	29
	Tempat Minum	102
	Toples	13

- **Penggunaan JOIN**

Sistem menggunakan INNER JOIN untuk menghasilkan laporan penjualan yang informatif dengan menggabungkan tabel transaksi, kasir, dan pelanggan.

SQL

```
SELECT T.noTransaksi, T.tanggal, K.namaKasir, P.idPelanggan, T.total
FROM transaksi T
```

JOIN pelanggan P ON T.idPelanggan = P.idPelanggan;

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

noTransaksi	tanggal	namaKasir	namaSales	idPelanggan	total
JP/001/001/20251108/150535	2025-11-08	MAJU KASTALAN	DHEA	AFD34	4800

3.5 Implementasi TCL (Transaction Control Language)

TCL digunakan untuk menjamin konsistensi data, seperti saat melakukan pembaruan stok barang. Jika terjadi kesalahan, perintah ROLLBACK akan membatalkan semua perubahan dalam sesi transaksi tersebut.

SQL

START TRANSACTION;
UPDATE barang SET stock = stock - 1 WHERE idBarang = 134;
ROLLBACK; -- Membatalkan pengurangan stok jika transaksi gagal

```
START TRANSACTION;

UPDATE barang
SET stock = stock - 1
WHERE idBarang = 134;

ROLLBACK;
```

3.6 Penggunaan View

VIEW dibuat untuk mempermudah manajemen dalam melihat laporan hubungan antara barang dan vendor tanpa perlu menulis query join yang kompleks secara berulang kali.

SQL

CREATE VIEW view_laporan_barang_vendor AS
SELECT B.unitBarang, B.harga, V.namaVendor
FROM barang B
JOIN vendor V ON B.idBarang = V.idBarang;
SELECT * FROM view_laporan_barang_vendor;

Result Grid | Filter Rows: | Export:

unitBarang	harga	namaVendor
Tupperware sedang 600ml	7500	ADAFIVENDORS
Tupperware besar 1L	12000	ADAFIVENDORS
Gelas plastik bening	15000	CV. JAYA PLASTIK
Piring plastik	20000	CV. JAYA PLASTIK
Sendok plastik	10000	CV. JAYA PLASTIK
Botol minum plastik 500ml	10000	CV. JAYA PLASTIK
Toples plastik kecil	9000	TOKO ANEKA WADAH
Toples plastik besar	15000	TOKO ANEKA WADAH
Wadah makanan sekat	13000	TOKO ANEKA WADAH

BAB 4: PENUTUP

4.1 Ringkasan Hasil Uji dan Eksekusi Query

Berdasarkan hasil uji coba pada database UAS, sistem telah berhasil menjalankan berbagai fungsi logika basis data sesuai dengan materi perkuliahan. Ringkasan hasil eksekusi adalah sebagai berikut:

- **Integritas Relasi**: Tabel master dan transaksi telah terhubung dengan baik melalui *Foreign Key*, terbukti dengan keberhasilan query JOIN yang menampilkan data lengkap antara kasir, sales, dan pelanggan dalam satu nota transaksi.
- **Akurasi Agregasi**: Query GROUP BY dan HAVING berhasil mengklasifikasikan stok barang berdasarkan kategorinya, seperti pengelompokan produk Alat Makan, Tempat Minum, dan Toples.
- **Keamanan Data**: Pengujian fitur TCL menunjukkan bahwa sistem mampu melakukan pembatalan perubahan data menggunakan ROLLBACK saat terjadi simulasi kesalahan input pada pembaruan stok.
- **Efisiensi Akses**: Penggunaan VIEW mempermudah penarikan informasi inventori barang dan vendor tanpa harus menulis ulang perintah join yang kompleks.

4.2 Kendala dan Perbaikan

Selama proses perancangan dan implementasi, ditemukan beberapa kendala teknis, di antaranya:

- **Kendala Relasi**: Kesulitan awal dalam memetakan relasi antara vendor dan barang karena satu vendor dapat menyuplai banyak jenis barang.
- **Perbaikan**: Dilakukan normalisasi hingga tahap 3NF dan penambahan tabel jenisBarang untuk memastikan struktur data lebih fleksibel dan efisien.
- **Kendala Query**: Terjadi *error* saat melakukan FULL OUTER JOIN karena keterbatasan fitur standar pada MySQL.
- **Perbaikan**: Menggunakan teknik UNION yang menggabungkan LEFT JOIN dan RIGHT JOIN untuk mendapatkan hasil audit (perhitungan) data yang komprehensip (menyeluruh).

4.3 Kesimpulan

Projek final sistem basis data ini memberikan kesimpulan sebagai berikut:

Perancangan basis data yang mengikuti kaidah (aturan) normalisasi (1NF-3NF) terbukti efektif dalam mencegah redundansi data dan anomali pada sistem inventori. Implementasi SQL yang mencakup DDL, DML, DQL, hingga TCL telah berhasil menyediakan sistem penyimpanan yang aman dan terstruktur untuk operasional retail. Penggunaan relasi tabel yang tepat memudahkan proses pengambilan keputusan melalui laporan-laporan yang dihasilkan secara cepat dan akurat.

4.4 Saran Pengembangan

Untuk meningkatkan performa dan fungsionalitas sistem di masa mendatang, disarankan beberapa pengembangan berikut:

- **Otomasi Stock**: Menambahkan fitur TRIGGER untuk secara otomatis mengurangi stok pada tabel barang setiap kali terjadi transaksi baru di tabel detailBarang.
- **Keamanan Lanjutan**: Mengimplementasikan *Stored Procedure* untuk enkripsi data sensitif pelanggan dan pengaturan hak akses pengguna (DCL) yang lebih ketat.
- **Fitur Bisnis**: Menambahkan fitur perhitungan diskon otomatis bagi pelanggan tetap dan sistem *tracking* performa sales berdasarkan total transaksi yang berhasil ditangani.

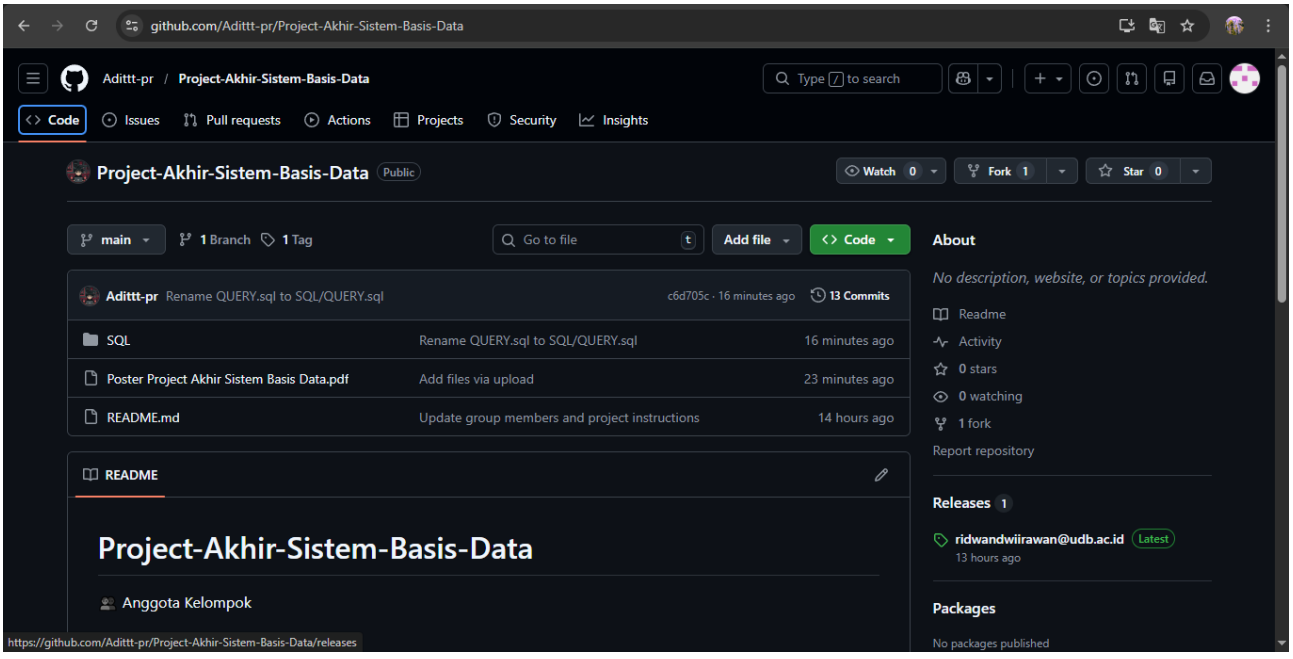
LAMPIRAN

L.1 Tautan Repository GitHub dan Ringkasan Struktur Folder

Seluruh artefak proyek, termasuk berkas SQL, poster, dan laporan ini, telah diunggah ke repository berikut sebagai bukti pengembangan yang transparan:

- **Link GitHub**: https://github.com/Adittt-pr/Project-Akhir-Sistem-Basis-Data.git
- **Struktur Folder**:
    - /SQL: Berisi file DATABASE dan QUERY.sql.
    - /dokumen: Berisi file Poster PDF dan Laporan PDF.
    - /md: Untuk isian README nya .

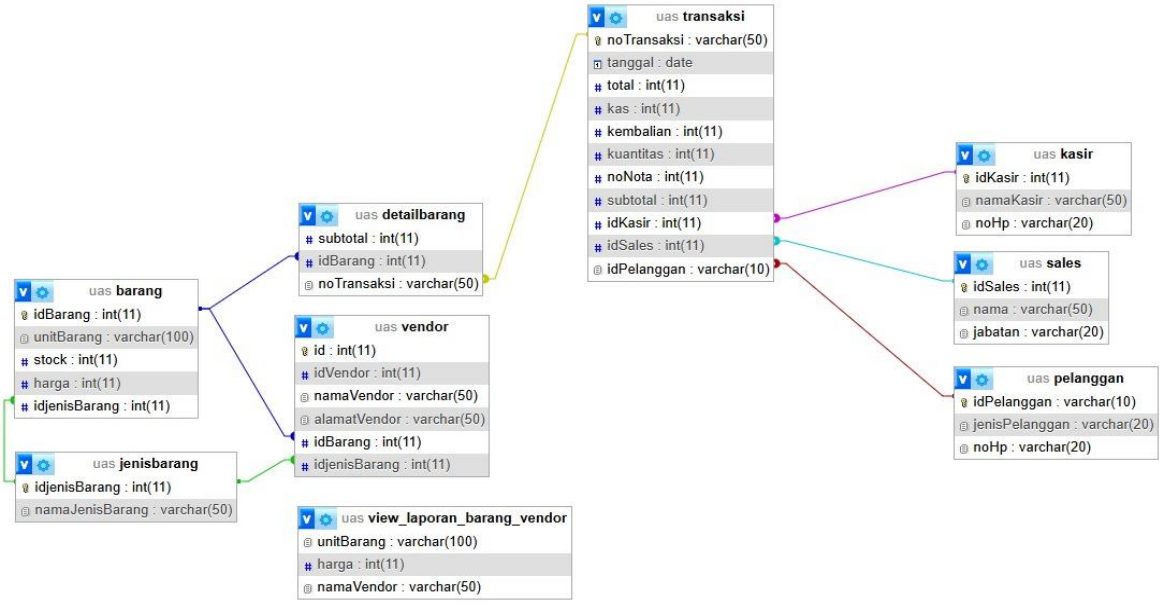
Tampilan dalam GitHub



Nb: untuk laporan belum ada karena ss nya sebelum send laporan ini ke github 😊

L.2 Screenshot ERD (MySQL Workbench)

Berikut adalah visualisasi hubungan antar entitas (ERD) yang dihasilkan melalui proses *Forward Engineering* pada MySQL Workbench atau Screenshoot Skena Database



L.3 Bukti Hasil Query dan Tampilan Program

Tangkapan layar di bawah ini menunjukkan hasil eksekusi beberapa query yang kami hasilkan guna untuk menggabungkan data transaksi dengan identitas kasir, stock barang , data sales dan serta data pelanggan, membuktikan bahwa relasi antar tabel berfungsi dengan benar.

select * from transaksi;

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

noTransaksi	tanggal	total	kas	kembalian	kuantitas	noNota	subtotal	idKasir	idSales	idPelanggan
JP/001/001/20251108/150535	08/11/2025	4800	20000	15000	Tinggi	154	4800	321879	8629026	AFD34
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

select * from barang;

Result Grid | Filter Rows: | Edit:

idBarang	unitBarang	stock	harga	idjenisBarang
134	Taperware kecil 300ml	50	4800	2
146	Tupperware sedang 600ml	20	7500	2
147	Tupperware besar 1L	19	12000	2
148	Gelas plastik bening	12	15000	2
149	Piring plastik	6	20000	1
150	Sendok plastik	12	10000	1
151	Botol minum plastik 500ml	1	10000	2
152	Toples plastik kecil	5	9000	3
153	Toples plastik besar	8	15000	3
154	Wadah makanan sekat	11	13000	1
NULL	NULL	NULL	NULL	NULL

select * from idjenisBarang;

Result Grid | Filter Rows:

idjenisBarang	namaJenisBarang
1	Alat Makan
2	Tempat Minum
3	Toples
NULL	NULL

select * from kasir;

idKasir	namaKasir	noHp
321879	MAJU KASTALAN	081225064867
NULL	NULL	NULL

select * from sales;

idSales	nama	jabatan
8629026	DHEA	Karyawan
NULL	NULL	NULL

select * from pelanggan;

Result Grid | Filter Rows: | Edit:

idPelanggan	jenisPelanggan	noHP
AFD34	Pelanggan Umum	087786549984
NULL	NULL	NULL

select * from vendor;

Result Grid | Filter Rows: | Edit: | Export/Import:

id	idVendor	namaVendor	alamatVendor	idBarang	idjenisBarang
1	123000	ADAFIVENDORS	JAKARTA	134	2
2	123000	ADAFIVENDORS	Jakarta	146	2
3	123000	ADAFIVENDORS	Jakarta	147	2
4	223000	CV. JAYA PLASTIK	Bandung	148	2
5	223000	CV. JAYA PLASTIK	Bandung	149	1
6	223000	CV. JAYA PLASTIK	Bandung	150	1
7	223000	CV. JAYA PLASTIK	Bandung	151	2
8	323000	TOKO ANEKA WADAH	Surabaya	152	3
9	323000	TOKO ANEKA WADAH	Surabaya	153	3
10	323000	TOKO ANEKA WADAH	Surabaya	154	1
NULL	NULL	NULL	NULL	NULL	NULL

1. ORDER BY QUERY

a. Query Penyaringan Stok (BETWEEN)

Query ini digunakan untuk mengidentifikasi produk yang memiliki jumlah stok dalam rentang kritis, yaitu antara 2 hingga 10 unit. Hal ini penting untuk manajemen inventori agar segera melakukan pengadaan ulang (*restock*) pada item yang hampir habis.

```
select idBarang, unitBarang, stock
from barang
where stock betwen 2 and 10;
```

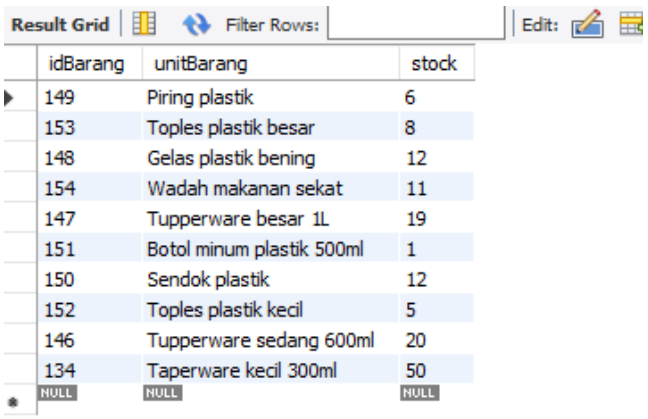
Result Grid | Filter Rows:

idBarang	unitBarang	stock
149	Piring plastik	6
152	Toples plastik kecil	5
153	Toples plastik besar	8
NULL	NULL	NULL

b. Query Pengurutan Harga (ORDER BY DESC)

Query ini bertujuan untuk menampilkan seluruh daftar produk yang terdaftar di database, namun diurutkan berdasarkan harga tertinggi ke harga terendah. Ini membantu kasir atau manajer dalam memantau nilai aset barang dari yang paling mahal.

```
select idBarang,unitBarang,stock
from barang
order by harga desc;
```



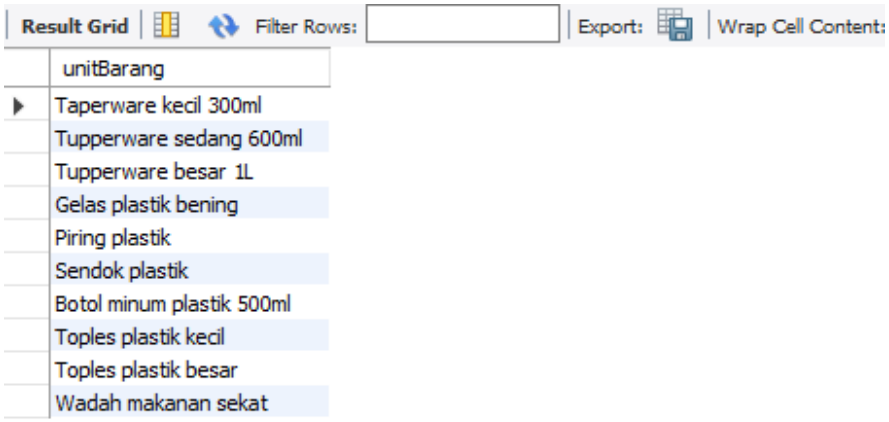
Result Grid | Filter Rows: | Edit:

idBarang	unitBarang	stock
149	Piring plastik	6
153	Toples plastik besar	8
148	Gelas plastik bening	12
154	Wadah makanan sekat	11
147	Tupperware besar 1L	19
151	Botol minum plastik 500ml	1
150	Sendok plastik	12
152	Toples plastik kecil	5
146	Tupperware sedang 600ml	20
134	Taperware kecil 300ml	50
NULL	NULL	NULL

2. DISTINCT QUERY

Penggunaan operator DISTINCT dalam query ini bertujuan untuk menampilkan data yang unik dan menghilangkan duplikasi (data ganda) dari hasil pencarian pada kolom tertentu.

```
select distinct unitBarang
from barang;
```



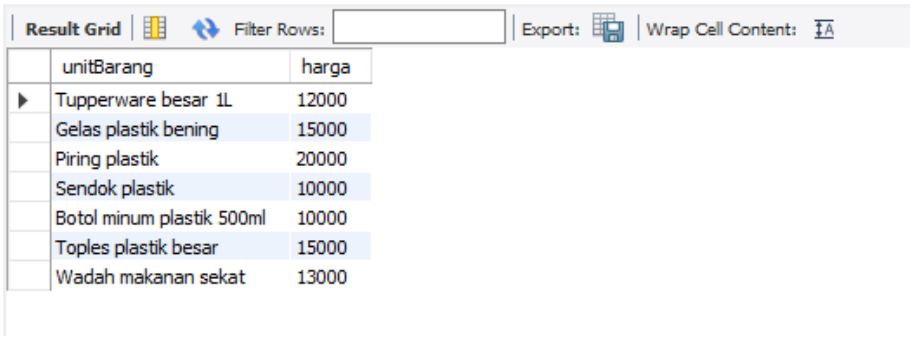
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

unitBarang
Taperware kecil 300ml
Tupperware sedang 600ml
Tupperware besar 1L
Gelas plastik bening
Piring plastik
Sendok plastik
Botol minum plastik 500ml
Toples plastik kecil
Toples plastik besar
Wadah makanan sekat

3. OPERATOR LOGIKA

Penggunaan operator logika dalam perintah SQL berfungsi untuk menyaring data berdasarkan kriteria perbandingan tertentu. Dalam skenario ini, digunakan operator >= (lebih besar atau sama dengan) untuk melakukan segmentasi harga barang.

```
select unitBarang,harga
from barang
where harga >= 10000;
```

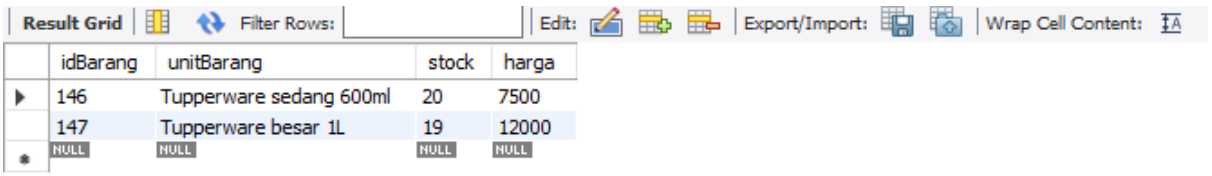


unitBarang	harga
Tupperware besar 1L	12000
Gelas plastik bening	15000
Piring plastik	20000
Sendok plastik	10000
Botol minum plastik 500ml	10000
Toples plastik besar	15000
Wadah makanan sekat	13000

4. LIKE QUERY

Penggunaan operator LIKE dalam query ini berfungsi untuk melakukan pencarian data berdasarkan pola karakter tertentu (*pattern matching*) pada kolom string. Dalam implementasinya, digunakan simbol *wildcard* % untuk mencari kata kunci di posisi mana pun dalam sebuah teks.

```
select * from barang
where unitBarang like '%Tupperware%';
```



idBarang	unitBarang	stock	harga
146	Tupperware sedang 600ml	20	7500
147	Tupperware besar 1L	19	12000
NULL	NULL	NULL	NULL

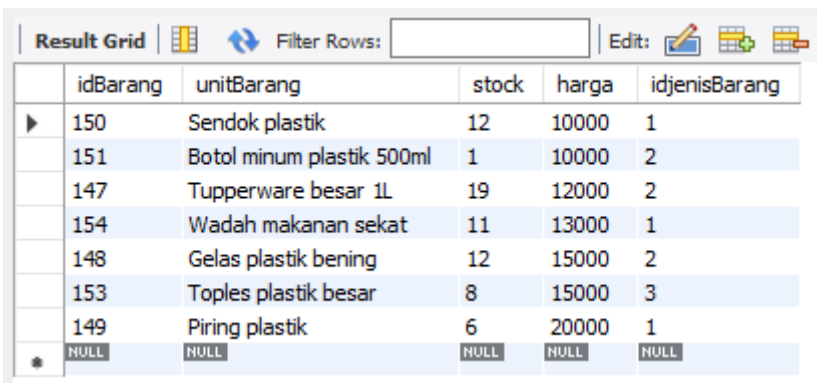
5. BETWEEN & RANGE

Penggunaan rentang nilai (*range*) dalam query bertujuan untuk menyaring data yang berada di antara batas nilai tertentu. Hal ini diimplementasikan menggunakan dua cara, yaitu operator perbandingan (>= dan <=) serta operator khusus BETWEEN.

a. Pencarian Rentang Harga (Operator Logika)

Query ini digunakan untuk menampilkan daftar barang yang memiliki harga di kisaran menengah, yaitu mulai dari Rp10.000 hingga Rp20.000. Penggunaan ORDER BY harga memastikan hasil laporan tersusun rapi dari harga terendah ke tertinggi.

```
select * from barang
where harga>=10000 and harga<=20000
order by harga;
```

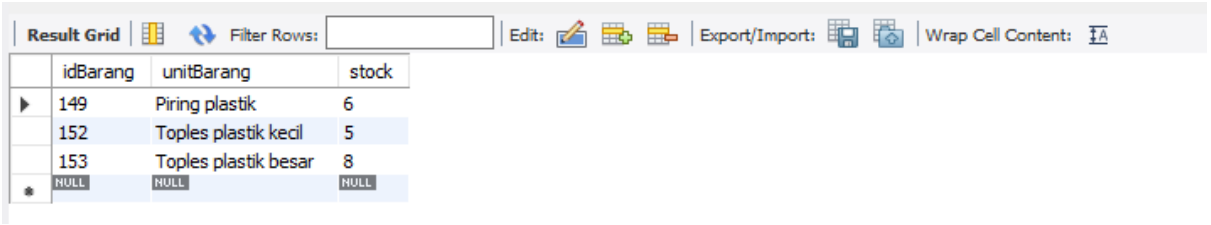


idBarang	unitBarang	stock	harga	idjenisBarang
150	Sendok plastik	12	10000	1
151	Botol minum plastik 500ml	1	10000	2
147	Tupperware besar 1L	19	12000	2
154	Wadah makanan sekat	11	13000	1
148	Gelas plastik bening	12	15000	2
153	Toples plastik besar	8	15000	3
149	Piring plastik	6	20000	1
NULL	NULL	NULL	NULL	NULL

b. Pencarian Rentang Stok (Operator BETWEEN)

Query ini menggunakan operator BETWEEN untuk menyaring jumlah stok barang yang berada dalam rentang minimal 2 hingga maksimal 10 unit.

```
select idBarang, unitBarang, stock
from barang
where stock betwen 2 AND 10;
```



idBarang	unitBarang	stock
149	Piring plastik	6
152	Toples plastik kecil	5
153	Toples plastik besar	8
NULL	NULL	NULL

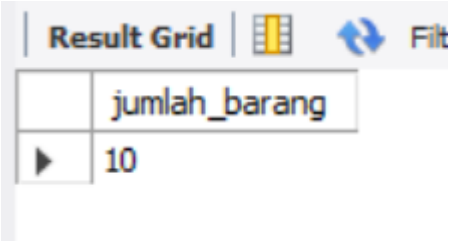
6. AGREGASI

Fungsi agregasi dalam SQL digunakan untuk melakukan perhitungan matematis pada sekumpulan nilai di dalam kolom tabel sehingga menghasilkan satu nilai ringkasan. Hal ini sangat berguna bagi manajemen untuk mendapatkan gambaran besar mengenai volume data dan total aset yang tersedia.

a. Menghitung Jumlah Record (COUNT)

Query ini berfungsi untuk menghitung total baris atau jumlah seluruh entitas barang yang terdaftar di dalam database toko.

```
SELECT COUNT(*) AS jumlah_barang
FROM barang;
```

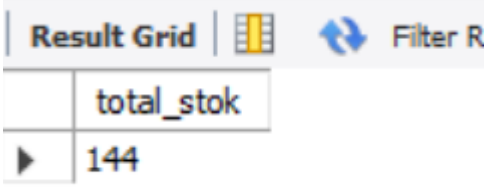


jumlah_barang
10

b. Menghitung Total Kuantitas (SUM)

Query ini digunakan untuk menjumlahkan seluruh nilai pada kolom stok barang untuk mengetahui akumulasi jumlah fisik barang yang ada di gudang.

```
select SUM(stock) AS total_stock
from barang;
```

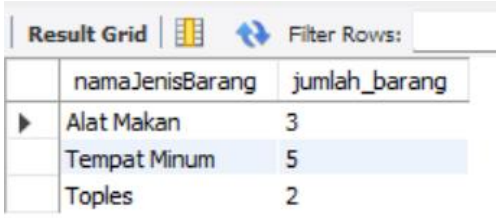


total_stok
144

7. GROUP BY

Perintah GROUP BY digunakan untuk mengelompokkan baris data yang memiliki nilai yang sama ke dalam ringkasan baris. Dalam kueri ini, data dikelompokkan berdasarkan kategori produk untuk memberikan informasi statistik per jenis barang.

```
SELECT j.namaJenisBarang, COUNT(b.idBarang) AS jumlah_barang
FROM barang b
JOIN jenisBarang j ON b.idjenisBarang = j.idjenisBarang
GROUP BY j.namaJenisBarang;
```



Result Grid | Filter Rows:

namaJenisBarang	jumlah_barang
Alat Makan	3
Tempat Minum	5
Toples	2

8. HAVING

HAVING adalah klausul yang digunakan untuk menyaring hasil kelompok data setelah perintah GROUP BY dijalankan. Perbedaan utamanya dengan WHERE adalah HAVING dapat digunakan bersama fungsi agregasi seperti SUM, COUNT, atau AVG.

```
SELECT j.namaJenisBarang, SUM(b.stock) AS total_stok
FROM barang b
JOIN jenisBarang j ON b.idjenisBarang = j.idjenisBarang
GROUP BY j.namaJenisBarang
HAVING SUM(b.stock) > 10;
```

Result Grid | Filter Rows:

namaJenisBarang	total_stok
Alat Makan	29
Tempat Minum	102
Toples	13

9. SUBQUERY

Subquery atau kueri bersarang adalah kueri di dalam kueri lainnya yang digunakan untuk melakukan perbandingan dinamis tanpa harus memasukkan nilai secara manual.

```
SELECT unitBarang, harga
FROM barang
WHERE harga > (
SELECT AVG(harga) FROM barang);
```

Result Grid | Filter Rows:

unitBarang	harga
Tupperware besar 1L	12000
Gelas plastik bening	15000
Piring plastik	20000
Toples plastik besar	15000
Wadah makanan sekat	13000

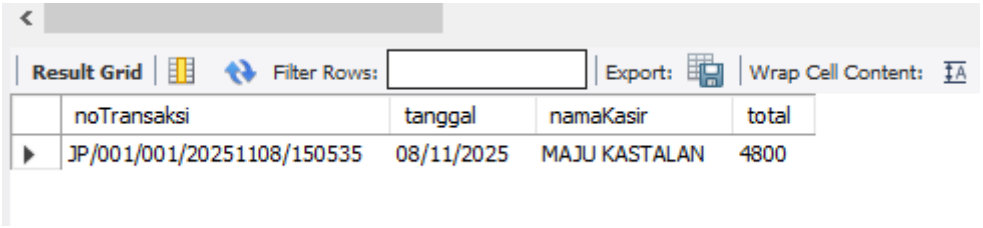
7. JOIN (Penggabungan Tabel)

Fungsi JOIN digunakan untuk menghubungkan baris-baris dari dua tabel atau lebih berdasarkan kolom yang saling terkait (Primary Key dan Foreign Key). Hal ini memungkinkan sistem untuk menyajikan data yang tersebar di berbagai tabel master menjadi satu laporan yang utuh.

a. Inner Join

Kueri ini digunakan untuk menampilkan data transaksi yang memiliki kecocokan data kasir yang valid.

```
SELECT T.noTransaksi, T.tanggal, K.namaKasir, T.total
FROM transaksi T INNER JOIN kasir K
ON T.idKasir = K.idKasir;
```

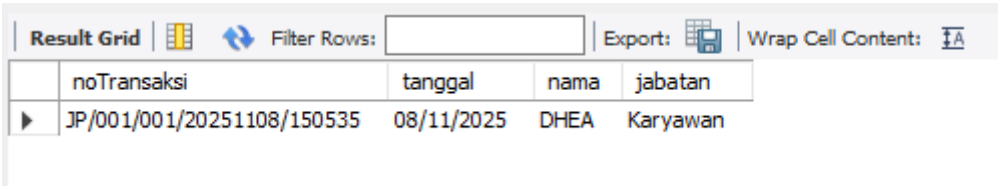


noTransaksi	tanggal	namaKasir	total
JP/001/001/20251108/150535	08/11/2025	MAJU KASTALAN	4800

b. Equi Join

Equi Join adalah bentuk penggabungan yang menggunakan operator kesamaan (=) pada klausa WHERE untuk menghubungkan tabel.

```
SELECT T.noTransaksi, T.tanggal, S.nama, S.jabatan
FROM transaksi T, sales S
WHERE T.idSales = S.idSales;
```



noTransaksi	tanggal	nama	jabatan
JP/001/001/20251108/150535	08/11/2025	DHEA	Karyawan

c. Left Join

Kueri ini memprioritaskan data dari tabel sebelah kiri (barang) agar tetap tampil semua meskipun tidak memiliki pasangan di tabel sebelah kanan (vendor).

```
SELECT B.unitBarang, B.harga, V.namaVendor, V.alamatVendor
FROM barang B LEFT JOIN vendor V
ON B.idBarang = V.idBarang;
```

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

unitBarang	harga	namaVendor	alamatVendor
Taperware kecil 300ml	4800	ADAFIVENDORS	JAKARTA
Tupperware sedang 600ml	7500	ADAFIVENDORS	Jakarta
Tupperware besar 1L	12000	ADAFIVENDORS	Jakarta
Gelas plastik bening	15000	CV. JAYA PLASTIK	Bandung
Piring plastik	20000	CV. JAYA PLASTIK	Bandung
Sendok plastik	10000	CV. JAYA PLASTIK	Bandung
Botol minum plastik 500ml	10000	CV. JAYA PLASTIK	Bandung
Toples plastik kecil	9000	TOKO ANEKA WADAH	Surabaya
Toples plastik besar	15000	TOKO ANEKA WADAH	Surabaya
Wadah makanan sekat	13000	TOKO ANEKA WADAH	Surabaya

d. Right Join

Kebalikan dari Left Join, kueri ini menampilkan seluruh data dari tabel sebelah kanan (vendor).

```
SELECT V.namaVendor, V.idjenisBarang, B.unitBarang
FROM barang B RIGHT JOIN vendor V
ON B.idBarang = V.idBarang;
```

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

namaVendor	idjenisBarang	unitBarang
ADAFIVENDORS	2	Taperware kecil 300ml
ADAFIVENDORS	2	Tupperware sedang 600ml
ADAFIVENDORS	2	Tupperware besar 1L
CV. JAYA PLASTIK	2	Gelas plastik bening
CV. JAYA PLASTIK	1	Piring plastik
CV. JAYA PLASTIK	1	Sendok plastik
CV. JAYA PLASTIK	2	Botol minum plastik 500ml
TOKO ANEKA WADAH	3	Toples plastik kecil
TOKO ANEKA WADAH	3	Toples plastik besar
TOKO ANEKA WADAH	1	Wadah makanan sekat

e. Full Join (menggunakan UNION)

Karena MySQL tidak mendukung perintah FULL OUTER JOIN secara langsung, digunakan fungsi UNION untuk menggabungkan hasil LEFT JOIN dan RIGHT JOIN.

```
SELECT B.idBarang, B.unitBarang, V.idVendor, V.namaVendor
FROM barang B
LEFT JOIN vendor V ON B.idBarang = V.idBarang
WHERE B.unitBarang LIKE '%a%'
UNION
SELECT B.idBarang, B.unitBarang, V.idVendor, V.namaVendor
FROM barang B
RIGHT JOIN vendor V ON B.idBarang = V.idBarang
WHERE B.unitBarang LIKE '%a%';
```

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

idBarang	unitBarang	idVendor	namaVendor
134	Taperware kecil 300ml	123000	ADAFIVENDORS
146	Tupperware sedang 600ml	123000	ADAFIVENDORS
147	Tupperware besar 1L	123000	ADAFIVENDORS
148	Gelas plastik bening	223000	CV. JAYA PLASTIK
149	Piring plastik	223000	CV. JAYA PLASTIK
150	Sendok plastik	223000	CV. JAYA PLASTIK
151	Botol minum plastik 500ml	223000	CV. JAYA PLASTIK
152	Toples plastik kecil	323000	TOKO ANEKA WADAH
153	Toples plastik besar	323000	TOKO ANEKA WADAH
154	Wadah makanan sekat	323000	TOKO ANEKA WADAH

8. VIEW (lebih lengkapnya)

Bagian ini menjelaskan penggunaan objek virtual untuk menyederhanakan pelaporan dan skenario query kompleks yang melibatkan banyak entitas sekaligus dalam satu output.

a. Implementasi View (Virtual Table)

VIEW digunakan untuk menyimpan kueri kompleks ke dalam sebuah tabel virtual. Hal ini mempermudah pengguna dalam memanggil laporan tanpa harus menulis ulang logika join yang panjang.

```
CREATE VIEW view_laporan_barang_vendor AS
SELECT B.unitBarang, B.harga, V.namaVendor
FROM barang B
JOIN vendor V ON B.idBarang = V.idBarang;

SELECT * FROM view_laporan_barang_vendor;
```

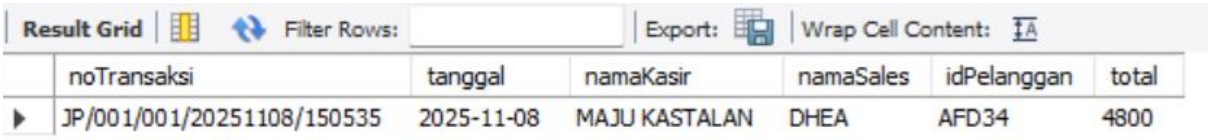
Result Grid | Filter Rows: | Export:

unitBarang	harga	namaVendor
Tupperware sedang 600ml	7500	ADAFIVENDORS
Tupperware besar 1L	12000	ADAFIVENDORS
Gelas plastik bening	15000	CV. JAYA PLASTIK
Piring plastik	20000	CV. JAYA PLASTIK
Sendok plastik	10000	CV. JAYA PLASTIK
Botol minum plastik 500ml	10000	CV. JAYA PLASTIK
Toples plastik kecil	9000	TOKO ANEKA WADAH
Toples plastik besar	15000	TOKO ANEKA WADAH
Wadah makanan sekat	13000	TOKO ANEKA WADAH

b. Skenario Transaksi Kompleks

Skenario ini merupakan implementasi kueri untuk menghasilkan data transaksi yang sangat lengkap dengan menghubungkan empat tabel sekaligus.

```
SELECT T.noTransaksi, T.tanggal,
    K.namaKasir, S.nama AS namaSales,
    P.idPelanggan, T.total
FROM transaksi T
JOIN kasir K ON T.idKasir = K.idKasir
JOIN sales S ON T.idSales = S.idSales
JOIN pelanggan P ON T.idPelanggan = P.idPelanggan;
```



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

noTransaksi	tanggal	namaKasir	namaSales	idPelanggan	total
JP/001/001/20251108/150535	2025-11-08	MAJU KASTALAN	DHEA	AFD34	4800

Nb: TCL sudah kami cantumkan dalam Bab 3.5 *Implementasi TCL (Transaction Control Language)*

L.4 Script SQL Lengkap

Berkas SQL lengkap yang digunakan dalam proyek ini meliputi pembuatan skema, pengisian data awal (*seeding*), dan skenario transaksi (TCL).

SQL

```
-- DDL
-- DATABASE

CREATE DATABASE IF NOT EXISTS UAS;
USE UAS;
-- MASTER TABLE
CREATE TABLE jenisBarang (
   idjenisBarang INT PRIMARY KEY,
   namaJenisBarang VARCHAR(50)
);
CREATE TABLE barang (
   idBarang INT PRIMARY KEY,
   unitBarang VARCHAR(100),
   stock INT,
   harga INT,
   idjenisBarang INT,
   FOREIGN KEY (idjenisBarang) REFERENCES jenisBarang(idjenisBarang)
);
CREATE TABLE kasir (
   idKasir INT PRIMARY KEY,
   namaKasir VARCHAR(50),
   noHp VARCHAR(20)
);
CREATE TABLE sales (
   idSales INT PRIMARY KEY,
   nama VARCHAR(50),
   jabatan VARCHAR(20)
);
CREATE TABLE pelanggan (
   idPelanggan VARCHAR(10) PRIMARY KEY,
   jenisPelanggan VARCHAR(20),
   noHp VARCHAR(20)
);
```

```
CREATE TABLE vendor (
    id INT AUTO_INCREMENT PRIMARY KEY,
    idVendor INT,
    namaVendor VARCHAR(50),
    alamatVendor VARCHAR(50),
    idBarang INT,
    idjenisBarang INT,
    FOREIGN KEY (idBarang) REFERENCES barang(idBarang),
    FOREIGN KEY (idjenisBarang) REFERENCES jenisBarang(idjenisBarang)
);
-- TRANSAKSI
CREATE TABLE transaksi (
    noTransaksi VARCHAR(50) PRIMARY KEY,
    tanggal DATE,
    total INT,
    kas INT,
    kembalian INT,
    kuantitas INT,
    noNota INT,
    subtotal INT,
    idKasir INT,
    idSales INT,
    idPelanggan VARCHAR(10),
    FOREIGN KEY (idKasir) REFERENCES kasir(idKasir),
    FOREIGN KEY (idSales) REFERENCES sales(idSales),
    FOREIGN KEY (idPelanggan) REFERENCES pelanggan(idPelanggan)
);
CREATE TABLE detailBarang (
    subtotal INT,
    idBarang INT,
    noTransaksi VARCHAR(50),
    FOREIGN KEY (idBarang) REFERENCES barang(idBarang),
    FOREIGN KEY (noTransaksi) REFERENCES transaksi(noTransaksi)
);
```

```
--- DML(INSERT)
-- INSERT DATA MASTER
INSERT INTO jenisBarang VALUES
(1,'Alat Makan'),
(2,'Tempat Minum'),
(3,'Toples');

INSERT INTO barang VALUES
(134,'Tupperware kecil 300ml',50,4800,2),
(146,'Tupperware sedang 600ml',20,7500,2),
(147,'Tupperware besar 1L',19,12000,2),
(148,'Gelas plastik bening',12,15000,2),
(149,'Piring plastik',6,20000,1),
(150,'Sendok plastik',12,10000,1),
(151,'Botol minum plastik 500ml',1,10000,2),
(152,'Toples plastik kecil',5,9000,3),
(153,'Toples plastik besar',8,15000,3),
(154,'Wadah makanan sekat',11,13000,1);

INSERT INTO kasir VALUES
(321879,'MAJU KASTALAN','081225064867');

INSERT INTO sales VALUES
(8629026,'DHEA','Karyawan');

INSERT INTO pelanggan VALUES
('AFD34','Pelanggan Umum','087786549984');

INSERT INTO vendor (idVendor,namaVendor,alamatVendor,idBarang,idjenisBarang) VALUES
(123000,'ADAFIVENDORS','Jakarta',146,2),
(123000,'ADAFIVENDORS','Jakarta',147,2),
(223000,'CV. JAYA PLASTIK','Bandung',148,2),
(223000,'CV. JAYA PLASTIK','Bandung',149,1),
(223000,'CV. JAYA PLASTIK','Bandung',150,1),
(223000,'CV. JAYA PLASTIK','Bandung',151,2),
(323000,'TOKO ANEKA WADAH','Surabaya',152,3),
(323000,'TOKO ANEKA WADAH','Surabaya',153,3),
```

```
(323000,'TOKO ANEKA WADAH','Surabaya',154,1);
INSERT INTO transaksi VALUES
('JP/001/001/20251108/150535','2025-11-
08',4800,20000,15000,1,154,4800,321879,8629026,'AFD34');

-- BAGIAN QUERY
-- GUNAKAN DATABASE

USE UAS;
--- DML (SELECT DASAR-JOIN)
-- 1. SELECT DASAR

SELECT * FROM barang;
SELECT * FROM vendor;
SELECT * FROM transaksi;
select * from kasir;
select * from sales;
select * from pelanggan;
select * from jenisBarang;

-- 2. ORDER BY & BETWEEN

SELECT idBarang, unitBarang, stock
FROM barang
ORDER BY harga DESC;

SELECT idBarang, unitBarang, stock
FROM barang
WHERE stock BETWEEN 2 AND 10;

-- 3. DISTINCT

SELECT DISTINCT namaVendor
FROM vendor;

-- 4. OPERATOR LOGIKA
```

```
SELECT unitBarang, harga
FROM barang
WHERE harga >= 10000;

-- 5. LIKE QUERY

SELECT * FROM barang
WHERE unitBarang LIKE '%Tupperware%';

-- 6. AGREGASI

SELECT COUNT(*) AS jumlah_barang
FROM barang;

SELECT SUM(stock) AS total_stok
FROM barang;

SELECT AVG(harga) AS rata_rata_harga
FROM barang;

-- 7. GROUP BY

SELECT j.namaJenisBarang, COUNT(b.idBarang) AS jumlah_barang
FROM barang b
JOIN jenisBarang j ON b.idjenisBarang = j.idjenisBarang
GROUP BY j.namaJenisBarang;

-- 8. HAVING

SELECT j.namaJenisBarang, SUM(b.stock) AS total_stok
FROM barang b
JOIN jenisBarang j ON b.idjenisBarang = j.idjenisBarang
GROUP BY j.namaJenisBarang
HAVING SUM(b.stock) > 10;

-- 9. SUBQUERY
```

```
SELECT unitBarang, harga
FROM barang
WHERE harga > (
   SELECT AVG(harga) FROM barang
);

-- 10. JOIN

-- INNER JOIN
SELECT T.noTransaksi, T.tanggal, K.namaKasir, T.total
FROM transaksi T
JOIN kasir K ON T.idKasir = K.idKasir;

-- EQUI JOIN
SELECT T.noTransaksi, P.idPelanggan, S.nama AS namaSales, T.total
FROM transaksi T
JOIN pelanggan P ON T.idPelanggan = P.idPelanggan
JOIN sales S ON T.idSales = S.idSales;

-- LEFT JOIN
SELECT B.unitBarang, V.namaVendor
FROM barang B
LEFT JOIN vendor V ON B.idBarang = V.idBarang;

-- RIGHT JOIN
SELECT V.namaVendor, V.idjenisBarang, B.unitBarang
FROM barang B RIGHT JOIN vendor V
ON B.idBarang = V.idBarang;

-- FULL JOIN (MySQL pakai UNION)
SELECT B.idBarang,B.unitBarang,V.idVendor,V.namaVendor
FROM barang B
LEFT JOIN vendor V ON B.idBarang = V.idBarang

UNION

SELECT B.idBarang,B.unitBarang,V.idVendor,V.namaVendor
```

```
FROM barang B
RIGHT JOIN vendor V ON B.idBarang = V.idBarang;

-- 11. view

CREATE VIEW view_laporan_barang_vendor AS
SELECT B.unitBarang, B.harga, V.namaVendor
FROM barang B
JOIN vendor V ON B.idBarang = V.idBarang;

SELECT * FROM view_laporan_barang_vendor;

-- 11.SCENARIO TRANSAKSI

SELECT T.noTransaksi, T.tanggal,
    K.namaKasir, S.nama AS namaSales,
    P.idPelanggan, T.total
FROM transaksi T
JOIN kasir K ON T.idKasir = K.idKasir
JOIN sales S ON T.idSales = S.idSales
JOIN pelanggan P ON T.idPelanggan = P.idPelanggan;

--- TCL
START TRANSACTION;
UPDATE barang
SET stock = stock - 1
WHERE idBarang = 134;
```

DAFTAR PUSTAKA

1. Irawan RD. *Modul Praktikum Pemrograman Basis Data*. Surakarta: Universitas Duta Bangsa Surakarta; 2024.
2. Silberschatz A, Korth HF, Sudarshan S. *Database System Concepts*. 7th ed. New York: McGraw-Hill; 2019.
3. Elmasri R, Navathe SB. *Fundamentals of Database Systems*. 7th ed. Boston: Pearson; 2016.
4. Universitas Duta Bangsa Surakarta. *Pedoman Penulisan Laporan Tugas Akhir Fakultas Ilmu Komputer*. Surakarta: UDB; 2023.