

**LAPORAN AKHIR  
IMPLEMENTASI BASIS DATA**



**DISUSUN OLEH:**

**NIM : 123240051**  
**NAMA : MUH. RAMADHAN**  
**KELAS : PRAKTIKUM IMPLEMENTASI BASIS  
DATA IF – C**  
**ASISTEN : PANCA AULIA RAHMAN**  
**ESEKIEL JANFERI SITEPU**

**PROGRAM STUDI INFORMATIKA  
JURUSAN TEKNIK INFORMATIKA  
FAKULTAS TEKNIK INDUSTRI  
UNIVERSITAS PEMBANGUNAN NASIONAL “VETERAN”  
YOGYAKARTA  
2024**

# HALAMAN PENGESAHAN

## LAPORAN AKHIR PRAKTIKUM IMPLEMENTASI BASIS DATA

Disusun Oleh:

Muh. Ramadhan

123240051

Telah Diperiksa dan Disetujui oleh Asisten Praktikum Implementasi Basis  
Data

Menyetujui,  
Asisten Praktikum

Asisten Praktikum

PANCA AULIA RAHMAN

NIM. 123200099

ESEKIEL JANFERI SITEPU

NIM. 124230087

## DAFTAR ISI

<b>LAPORAN AKHIR IMPLEMENTASI BASIS DATA.....</b>	<b>i</b>
<b>HALAMAN PENGESAHAN.....</b>	<b>ii</b>
<b>DAFTAR ISI.....</b>	<b>iii</b>
<b>TUGAS 1 ERD .....</b>	<b>1</b>
1.1    Studi Kasus .....	1
1.2    ERD .....	1
1.3    Catatan Revisi .....	3
1.4    Revisi .....	3
<b>TUGAS 2 RAT .....</b>	<b>4</b>
2.1    RAT.....	4
2.2    Catatan Revisi .....	4
2.3    Revisi .....	5
<b>TUGAS 3 CREATE TABLE.....</b>	<b>6</b>
3.1    Query, Screenshot, dan Penjelasan .....	6
3.2    Catatan Revisi .....	11
3.3    Revisi .....	12
<b>TUGAS 4 PERINTAH SQL.....</b>	<b>13</b>
4.1    Query, Screenshot, dan Penjelasan .....	13
4.2    Catatan Revisi .....	19
4.3    Revisi .....	19
<b>TUGAS 5 PERINTAH SQL LANJUTAN .....</b>	<b>20</b>
5.1    Query, Screenshot, dan Penjelasan .....	20
5.2    Catatan Revisi .....	32
5.3    Revisi .....	32

# **TUGAS 1**

## **ERD**

### **1.1 Studi Kasus**

Studi kasus ini berfokus pada implementasi sistem basis data di supermarket adalah analisis yang dilakukan untuk memahami berbagai aspek operasional, manajerial, atau strategis dari sebuah supermarket. Studi kasus ini biasanya digunakan untuk mengidentifikasi masalah, mengevaluasi kinerja, atau menemukan peluang perbaikan dan inovasi.

Dengan melakukan studi kasus, supermarket dapat mengidentifikasi kebutuhan pelanggan, meningkatkan efisiensi, dan meraih keuntungan yang lebih besar. Dan Studi kasus biasanya memiliki fokus tertentu, misalnya: Masalah Operasional, Strategi Pemasaran, Kinerja Keuangan, Transformasi Digital dan Keputusan pelanggan.

Supermarket adalah jenis toko ritel besar yang menyediakan berbagai macam barang kebutuhan sehari-hari di bawah satu atap. Konsep supermarket dirancang untuk memberikan kemudahan kepada pelanggan dengan menyediakan berbagai produk yang terorganisir dalam bermacam-macam kategori-kategori.

### **1.2 ERD**

#### **Entitas dan Atribut**

#### **1. Pelanggan**

##### **Atribut:**

- id\_pelanggan: Merupakan primary key yang berfungsi sebagai pelanggan yang ingin membeli
- nama: Merupakan nama pelanggan yang ingin membeli
- alamat: Alamat lengkap seorang pelanggan yang telah membeli
- no\_tlpn: Nomor telepon pelanggan untuk keperluan kontak

#### **2. Produk**

##### **Atribut:**

- id\_produk: Merupakan primary key yang berfungsi sebagai identifikasi unik produk
- nama\_produk: yang berfungsi sebagai nama dalam sebuah produk
- harga: merupakan harga sebuah barang
- stok; apakah sebuah produk memiliki stok

#### **3. Penjualan**

**Atribut:**

- id\_penjualan: Merupakan primary key yang berfungsi sebagai identifikasi unik penjualan
- tanggal\_produk: Merupakan tanggal produk yang dijual
- id\_pelanggan: Merupakan foreign key ke entitas Pelanggan

**4. Detail\_Penjualan****Atribut:**

- id\_detail\_penjualan: primary key untuk setiap penjualan
- harga\_satuan: merupakan harga produk yang dijual dalam satuan
- jumlah: jumlah produk yang dijual
- id\_penjualan: foreign key ke entitas Penjualan
- id\_produk: foreign key ke entitas Produk

**Kardinalitas****1. Pelanggan berelasi dengan:**

tabel Penjualan (1:M) dengan relasi “memiliki”

- Satu pelanggan dapat melakukan banyak penjualan.
- Setiap penjualan hanya terkait dengan satu pelanggan.

**2. Penjualan berelasi dengan:**

Table detail\_Penjualan (1:M) dengan relasi “terdiri dari”

- Satu penjualan dapat terdiri dari banyak detail penjualan.
- Setiap detail penjualan hanya terkait dengan satu penjualan.

**3. Detail\_Penjualan berelasi dengan;**

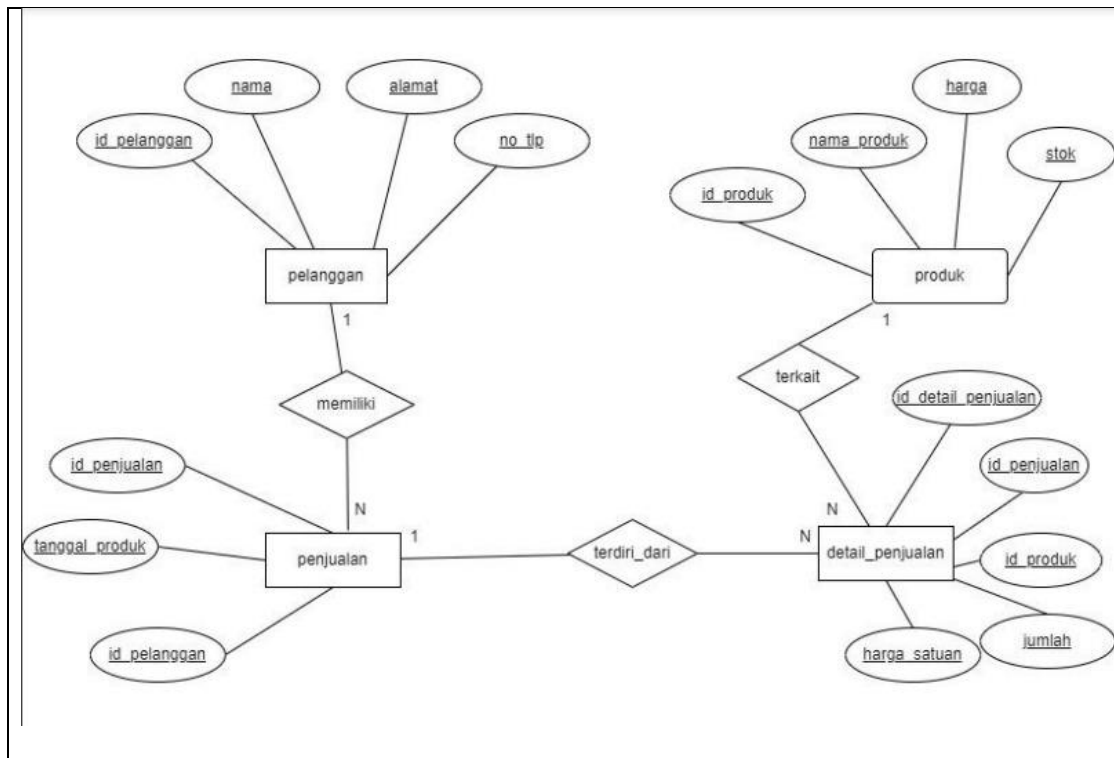
Table Produk (1:M) dengan relasi “terkait”

- Banyak detail penjualan dapat terkait dengan satu produk.
- Setiap produk hanya terkait dengan satu detail penjualan.

**4. Pelanggan berelasi dengan:**

Table Produk (1:M) dengan relasi “membeli”

- Satu pelanggan dapat membeli banyak produk.
- Satu produk dapat dibeli oleh banyak pelanggan.



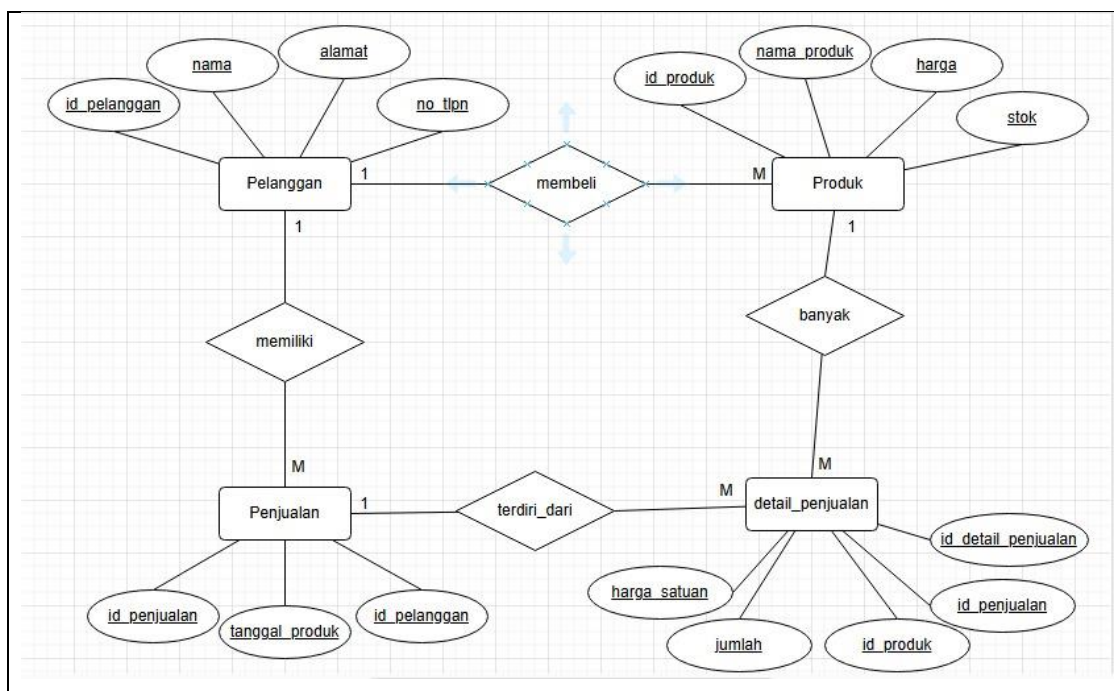
### 1.3 Catatan Revisi

Pelanggan dapat berelasi ke produk

Primary key hanya Satu

Hubungan antar entitas kurang jelas

### 1.4 Revisi



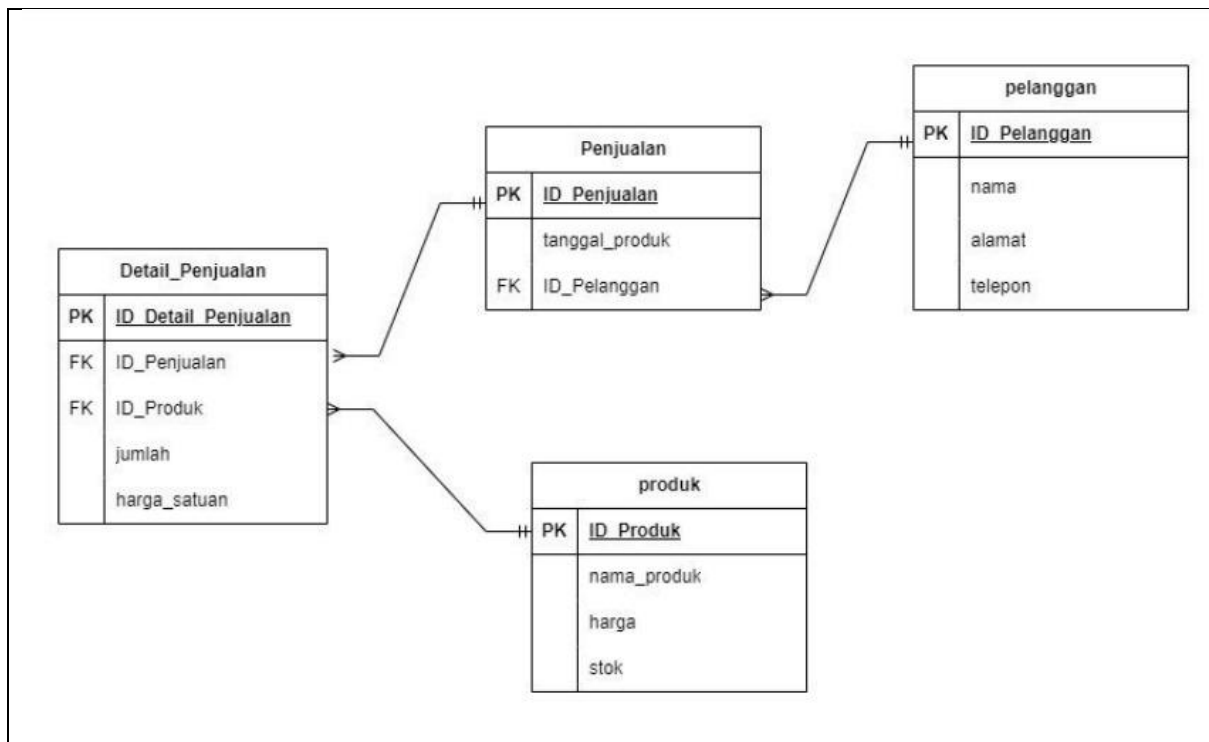
## TUGAS 2

### RELASI ANTAR TABEL (RAT)

Membuat RAT sesuai dengan pembuatan ERD sebelumnya dengan study kasus Supermarket

#### 2.1 RAT

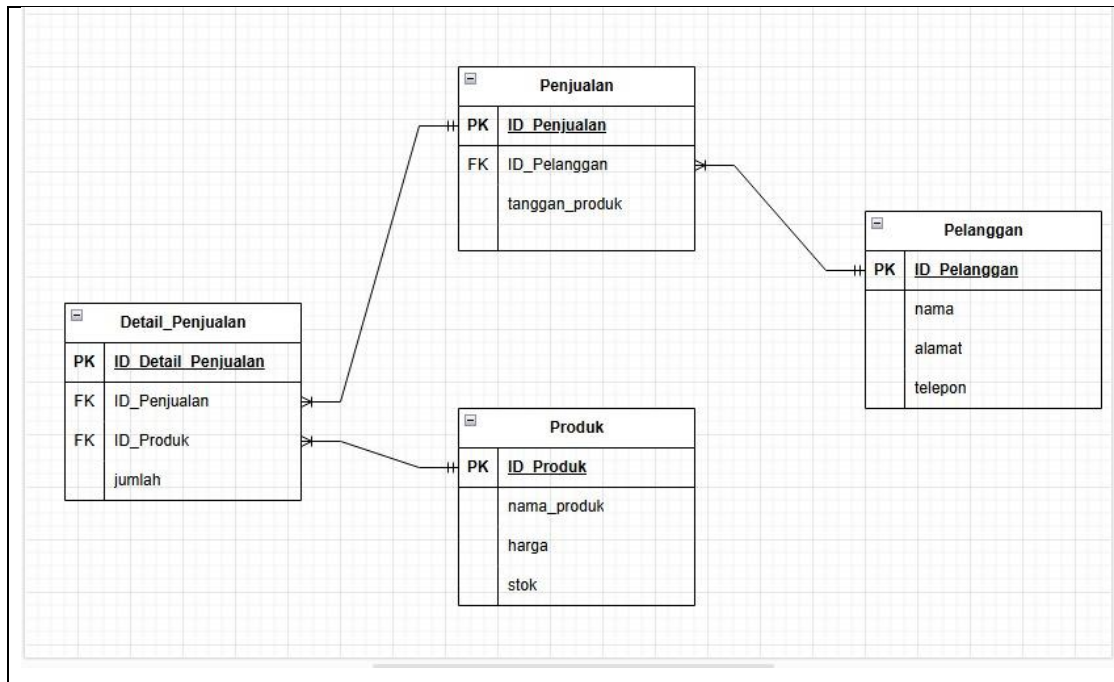
- Tabel detail\_penjualan sebagai entitas dan id\_detail\_penjualan sebagai primary key, id\_penjualan sebagai foreign key, id\_produk sebagai foreign key, serta jumlah dan harga\_satuan sebagai atribut.
- Tabel penjualan sebagai entitas dan id\_penjualan sebagai primary key, id\_pelanggan sebagai foreign key, serta tanggal\_produk sebagai atribut.
- Tabel pelanggan sebagai entitas dan id\_pelanggan sebagai primary key, serta nama, Alamat, telepon sebagai atribut.
- Tabel produk sebagai entitas dan id\_produk sebagai primary key serta nama\_produk, harga, stok.



#### 2.2 Catatan Revisi

FK sebaiknya jangan ditaruh di bawah

## 2.3 Revisi





## TUGAS 3

### NAMA TUGAS JIKA ADA

Database ini dirancang untuk mengelola operasional supermarket dengan efisien. Terdapat empat entitas utama yang akan diimplementasikan dalam bentuk tabel, yaitu Pelanggan, Penjualan, Produk, Detail\_Penjualan dan Jadwal pelanggan. Setiap tabel memiliki atribut yang relevan untuk mendukung fungsi sistem.

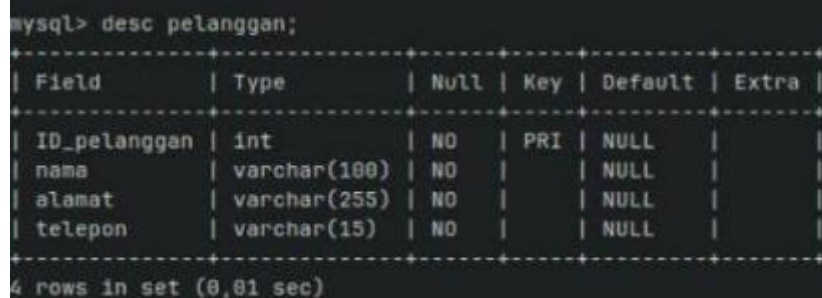
#### 3.1 Query, Screenshot, dan Penjelasan

##### 1. Tabel pelanggan

- Query

```
Create table pelanggan (  
ID_Pelanggan INT NOT NULL PRIMARY KEY,  
Nama varchar(100),  
Alamat varchar(255),  
Telepon varchar(15)  
);
```

- Hasil Query



```
mysql> desc pelanggan;  
+-----+-----+-----+-----+-----+-----+  
| Field          | Type          | Null | Key | Default | Extra |  
+-----+-----+-----+-----+-----+-----+  
| ID_pelanggan  | int           | NO   | PRI | NULL    |       |  
| nama          | varchar(100)  | NO   |     | NULL    |       |  
| alamat        | varchar(255)  | NO   |     | NULL    |       |  
| telepon       | varchar(15)   | NO   |     | NULL    |       |  
+-----+-----+-----+-----+-----+-----+  
4 rows in set (0.01 sec)
```

- Untuk menampilkan struktur tabel Pelanggan

## 2. Table produk

- Query

```
Create table produk (  
ID_Produk int(11) NOT NULL PRIMARY KEY,  
Nama_produk varchar(100),  
harga varchar(11),  
stok varchar(11)  
);
```

- Hasil Query

```
MariaDB [supermarket2]> desc produk;  
+-----+-----+-----+-----+-----+-----+  
| Field      | Type          | Null | Key | Default | Extra |  
+-----+-----+-----+-----+-----+-----+  
| id_produk  | int(11)       | NO   | PRI | NULL    |       |  
| nama_produk| varchar(100)  | NO   |     | NULL    |       |  
| harga      | int(11)       | NO   |     | NULL    |       |  
| stok       | int(11)       | NO   |     | NULL    |       |  
+-----+-----+-----+-----+-----+-----+  
4 rows in set (0.029 sec)
```

- Untuk menampilkan struktur tabel Produk

## 3. Table penjualan

- Query

```
Create table penjualan (  
ID_penjualan int(11) NOT NULL PRIMARY KEY,  
ID_Produk varchar(11),  
Tanggal_produk date not null primary key  
);
```

- Hasil Query

```
MariaDB [supermarket2]> desc penjualan;
```

Field	Type	Null	Key	Default	Extra
ID_penjualan	int(11)	NO	PRI	NULL	
tanggal_produk	date	NO		NULL	
ID_pelanggan	int(11)	NO	MUL	NULL	

3 rows in set (0.026 sec)

- Untuk menampilkan struktur tabel Penjualan

#### 4. Table detail\_penjualan

- Query

```
Create table penjualan (
ID_detail_penjualan INT NOT NULL PRIMARY KEY,
ID_penjualan INT NOT NULL PRIMARY KEY,
ID_Produk INT NOT NULL PRIMARY KEY,
Jumlah INT NOT NULL PRIMARY KEY,
Harga_satuan INT NOT NULL PRIMARY KEY
);
```

- Hasil Query

```
mysql> desc detail_penjualan;
```

Field	Type	Null	Key	Default	Extra
ID_detail_penjualan	int	NO	PRI	NULL	
ID_penjualan	int	NO	MUL	NULL	
ID_produk	int	NO	MUL	NULL	
jumlah	int	NO		NULL	
harga_satuan	int	NO		NULL	

5 rows in set (0.01 sec)


- Untuk menampilkan struktur tabel detail\_Penjualan

## 5. Table detail\_penjualan

- Query

```
INSERT INTO PELANGGAN VALUES
(1, 'Andi pratama', 'jalan Sudirman no, 10, jakarta',
'081234567890'),
(2, 'budi satiawan', 'jalan mangga dua no, 5, bandung',
'08987654321'),
(3, 'citra dewi', 'jalan Merdeka no, 15, surabaya',
'082112233445'),
(4, 'dewi santoso', 'jalan kebon jeruk no, 7, yogyakarta',
'083345678981'),
(5, 'eka saputra', 'jalan diponegoro no, 22, medan',
'0844556677889')
;
```

- Hasil Query



```
mysql> select * from pelanggan;
+-----+-----+-----+-----+
| ID_pelanggan | nama      | alamat                                     | telepon |
+-----+-----+-----+-----+
| 1 | Andi Pratama | Jalan Sudirman No. 10, Jakarta | 081234567890 |
| 2 | Budi Setiawan | Jalan Mangga Dua No. 5, Bandung | 081987654321 |
| 3 | Citra Dewi | Jalan Merdeka No. 15, Surabaya | 082112233445 |
| 4 | Dedi Santoso | Jalan Kebon Jeruk No. 7, Yogyakarta | 083345678981 |
| 5 | Eka Saputra | Jalan Diponegoro No. 22, Medan | 084556677889 |
+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

- Untuk Menampilkan data pada tabel pelanggan

## 6. Table detail\_penjualan

- Query

```
INSERT INTO PENJUALAN VALUES
(1, 1, 1, 5, 120000),
(2, 2, 2, 10, 180000),
(3, 3, 3, 8, 150000),
(4, 4, 4, 12, 220000),
(5, 5, 5, 6, 120000);
```

- Hasil Query

```
mysql> select * from penjualan;
```

ID_penjualan	tanggal_produk	ID_pelanggan
1	2024-09-15	1
2	2024-09-16	2
3	2024-09-17	3
4	2024-09-18	4
5	2024-09-19	5

```
5 rows in set (0,00 sec)
```

- Untuk Menampilkan data pada tabel penjualan

## 7. Table produk

- Query

```
INSERT INTO PRODUK VALUES
(1, 'Beras Premium', 120000, 100),
(2, 'Minyak Goreng', 180000, 200),
(3, 'Gula Pasir', 150000, 150),
(4, 'Telur Ayam', 220000, 80),
(5, 'Tepung Terigu', 120000, 120);
```

- Hasil Query

```
mysql> select * from produk
-> ;
```

ID_produk	nama_produk	harga	stok
1	Beras Premium	120000	100
2	Minyak Goreng	180000	200
3	Gula Pasir	150000	150
4	Telur Ayam	220000	80
5	Tepung Terigu	120000	120

```
5 rows in set (0,00 sec)
```

- Untuk Menampilkan data pada tabel produk

## 8. Table detail\_penjualan

- Query

```
INSERT INTO DETAIL_PENJUALAN VALUES
(1, 1, 1, 5, 120000),
(2, 2, 2, 10, 180000),
(3, 3, 3, 8, 150000),
(4, 4, 4, 12, 220000),
(5, 5, 5, 6, 120000);
```

- Hasil Query

```
mysql> select * from detail_penjualan;
+-----+-----+-----+-----+-----+
| ID_detail_penjualan | ID_penjualan | ID_produk | jumlah | harga_satuan |
+-----+-----+-----+-----+-----+
| 1 | 1 | 1 | 5 | 120000 |
| 2 | 2 | 2 | 10 | 180000 |
| 3 | 3 | 3 | 8 | 150000 |
| 4 | 4 | 4 | 12 | 220000 |
| 5 | 5 | 5 | 6 | 120000 |
+-----+-----+-----+-----+-----+
5 rows in set (0,00 sec)
```

- Untuk Menampilkan data pada tabel detail\_penjualan

## 3.2 Catatan Revisi

Menambah query create table produk

### 3.3 Revisi

```
MariaDB [supermarket2]> desc produk;
```

Field	Type	Null	Key	Default	Extra
id_produk	int(11)	NO	PRI	NULL	
nama_produk	varchar(100)	NO		NULL	
harga	int(11)	NO		NULL	
stok	int(11)	NO		NULL	

```
4 rows in set (0.029 sec)
```

```
mysql> select * from detail_penjualan;
```

ID_detail_penjualan	ID_penjualan	ID_produk	jumlah	harga_satuan
1	1	1	5	120000
2	2	2	10	18000
3	3	3	8	15000
4	4	4	12	22000
5	5	5	6	12000

```
5 rows in set (0.00 sec)
```

## TUGAS 4

### PERINTAH SQL

Membuat perintah SQL dengan tabel dan study kasus yang telah di buat sebelumnya yaitu “**supermarket**” query perintah nya adalah sebagai berikut:

**SELECT:** Digunakan untuk memilih dan menampilkan data dari tabel.

**WHERE:** Digunakan untuk menyaring data berdasarkan kondisi tertentu.

**ORDER BY:** Digunakan untuk mengurutkan hasil query.

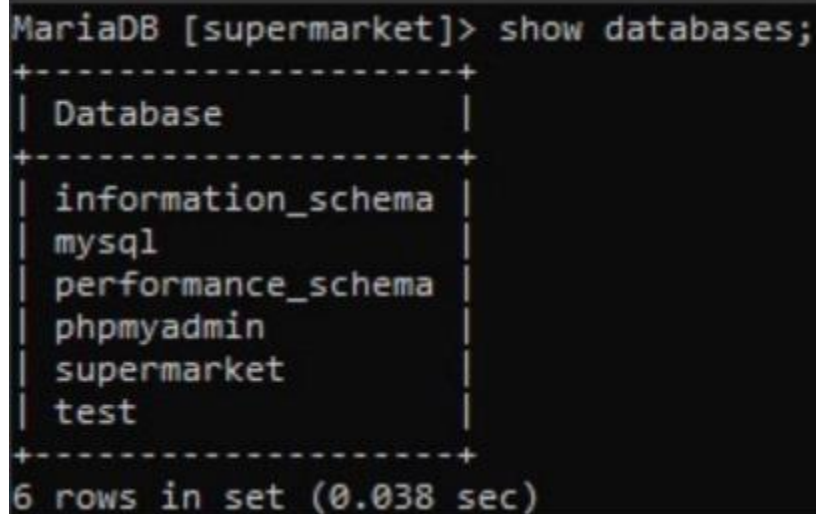
#### 4.1 Query, Screenshot, dan Penjelasan

##### ❖ Membuat databases

- Query

```
Show databases;
```

- Hasil query



```
MariaDB [supermarket]> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql          |
| performance_schema |
| phpmyadmin     |
| supermarket    |
| test           |
+-----+
6 rows in set (0.038 sec)
```

- Query tersebut untuk menampilkan semua databses

##### ❖ Membuat table pelanggan, penjualan, produk dan detail\_penjualan menggunakan describe

- Query

```
DESCRIBE PELANGGAN, PENJUALAN, PRODUK, DETAIL_PENJUALAN;
```



- Hasil query

```
MariaDB [supermarket]> DESCRIBE pelanggan;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Id_pelanggan | int(11)   | NO   | PRI | NULL    |       |
| nama        | varchar(100) | NO   |     | NULL    |       |
| alamat      | varchar(255) | NO   |     | NULL    |       |
| telepon     | varchar(15) | NO   |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.023 sec)
```

```
MariaDB [supermarket]> DESCRIBE penjualan;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id_penjualan | int(11)   | NO   | PRI | NULL    |       |
| tanggal_penjualan | date      | NO   |     | NULL    |       |
| id_pelanggan | int(11)   | NO   | MUL | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.021 sec)
```

```
MariaDB [supermarket]> DESCRIBE produk;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id_produk   | int(11)   | NO   | PRI | NULL    |       |
| tanggal_produk | date      | NO   |     | NULL    |       |
| nama_produk | varchar(100) | NO   |     | NULL    |       |
| harga_produk | decimal(10,2) | NO   |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.032 sec)
```

```
MariaDB [supermarket]> DESCRIBE detail_penjualan;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| id_detail_penjualan | int(11) | NO   | PRI | NULL    |       |
| id_penjualan | int(11)   | NO   | MUL | NULL    |       |
| id_produk    | int(11)   | NO   | MUL | NULL    |       |
| jumlah       | int(11)   | NO   |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.032 sec)
```

- Query tersebut untuk menampilkan semua describe pelanggan, penjualan, produk, detail\_penjualan.

❖ Membuat table pelanggan dan penjualan dengan menggunakan select \* from

- Query

```
SELECT * FROM PELANGGAN, SELECT * FROM PENJUALAN;
```

- Hasil query

```
MariaDB [supermarket]> SELECT * FROM pelanggan;
+-----+-----+-----+-----+
| Id_pelanggan | nama      | alamat  | telepon |
+-----+-----+-----+-----+
| 12300        | Ahmad     | Jakarta | 08123456789 |
| 12301        | Zainuddin | Jogja   | 08123456788 |
| 12302        | Hasan     | Sleman  | 08123456787 |
| 12303        | Hikmah    | Bantul  | 08123456786 |
| 12304        | Zidan     | Kaliurang | 08123456785 |
+-----+-----+-----+-----+
5 rows in set (0.001 sec)

MariaDB [supermarket]> SELECT * FROM penjualan;
+-----+-----+-----+
| id_penjualan | tanggal_penjualan | id_pelanggan |
+-----+-----+-----+
| 20011        | 2023-10-01         | 12300        |
| 20012        | 2023-10-02         | 12301        |
| 20013        | 2023-10-03         | 12302        |
| 20014        | 2023-10-04         | 12303        |
| 20015        | 2023-10-05         | 12304        |
+-----+-----+-----+
5 rows in set (0.001 sec)
```

- Query tersebut untuk menampilkan semua select \* from pelanggan dan select from penjualan.

#### ❖ Membuat table Alamat dan nomor hp

- Query

```
SELECT ALAMAT AS ALAMAT, TELEPON AS "NOMOR HP" FROM
PELANGGAN;
```

- Hasil query

```
MariaDB [supermarket]> SELECT alamat as alamat, telepon as "Nomor Hp" from
pelanggan;
+-----+-----+
| alamat | Nomor Hp |
+-----+-----+
| Jakarta | 08123456789 |
| Jogja   | 08123456788 |
| Sleman  | 08123456787 |
| Bantul  | 08123456786 |
| Kaliurang | 08123456785 |
+-----+-----+
5 rows in set (0.001 sec)
```

- Query tersebut untuk menampilkan select Alamat as Alamat, telepon as "nomor hp" from pelanggan

- ❖ Membuat table pelanggan menggunakan select \* from dan where untuk menampilkan 5 id\_pelanggan, 5 nama, 5 alamat, dan 5 telepon

- Query

```
SELECT * FROM PELANGGAN WHERE ID_PELANGGAN =1200;
```

- Hasil query

```
MariaDB [supermarket]> SELECT * FROM pelanggan WHERE ID_pelanggan =1200;
```

Id_pelanggan	nama	alamat	telepon
12300	Ahmad	Jakarta	08123456789
12301	Zainuddin	Jogja	08123456788
12302	Hasan	Sleman	08123456787
12303	Hikmah	Bantul	08123456786
12304	Zidan	Kaliurang	08123456785

```
5 rows in set (0.001 sec)
```

- Query tersebut untuk menampilkan select \* from pelanggan where id\_pelanggan =1200

- ❖ Membuat table pelanggan menggunakan select \* from dan where untuk menampilkan 1 id\_pelanggan, 1 nama, 1 alamat dan 1 telepon

- Query

```
SELECT * FROM PELANGGAN WHERE ID_PELANGGAN =12300;
```

- Hasil query

```
MariaDB [supermarket]> SELECT * FROM pelanggan WHERE id_pelanggan =12300;
```

Id_pelanggan	nama	alamat	telepon
12300	Ahmad	Jakarta	08123456789

```
1 row in set (0.001 sec)
```

- Query tersebut untuk menampilkan select \* from pelanggan where id\_pelanggan =12300

- ❖ Membuat table pelanggan menggunakan desc

- Query

```
DESC PELANGGAN;
```

- Hasil query

```
MariaDB [supermarket]> desc pelanggan;
```

Field	Type	Null	Key	Default	Extra
Id_pelanggan	int(11)	NO	PRI	NULL	
nama	varchar(100)	NO		NULL	
alamat	varchar(255)	NO		NULL	
telepon	varchar(15)	NO		NULL	

4 rows in set (0.030 sec)

- Query tersebut untuk menampilkan desc pelanggan

- ❖ Membuat table pelanggan menggunakan select \* from, where, dan like untuk menampilkan 2 id\_pelanggan, 2 nama, 2 alamat, 2 telepon

- Query

```
SELECT * FROM PELANGGAN WHERE ALAMAT LIKE 'J%';
```

- Hasil query

```
MariaDB [supermarket]> SELECT * FROM pelanggan WHERE alamat LIKE 'j%';
```

Id_pelanggan	nama	alamat	telepon
12300	Ahmad	Jakarta	08123456789
12301	Zainuddin	Jogja	08123456788

2 rows in set (0.001 sec)

- Query tersebut untuk menampilkan select \* from pelanggan where Alamat like 'j%'

- ❖ Membuat table pelanggan menggunakan select \* from, where Alamat untuk menampilkan 1 id\_pelanggan, 1 nama, 1 alamat, 1 telepon

- Query

```
SELECT * FROM PELANGGAN WHERE ALAMAT 'JAKARTA';
```

- Hasil query

```
MariaDB [supermarket]> SELECT * FROM pelanggan WHERE alamat = 'jakarta';
```

Id_pelanggan	nama	alamat	telepon
12300	Ahmad	Jakarta	08123456789

1 row in set (0.001 sec)

- Query tersebut untuk menampilkan select \* from pelanggan where Alamat 'jakarta'

- ❖ Membuat table menggunakan select telepon as “telepon”, nama, id\_pelanggan, from pelanggan, Where telepon between, order by telepon

- Query

```
SELECT TELEPON S “TELEPON”, NAMA, ID_PELANGGAN  
FROM PELANGGAN  
WHERE TELEPON BETWEEN ‘08123456789’  
ORDER BY TELEPON DESC;
```

- Hasil query

```
MariaDB [supermarket]> SELECT telepon AS "telepon", nama, id_pelanggan  
-> FROM pelanggan  
-> WHERE telepon BETWEEN '08123456785' AND '08123456789'  
-> ORDER BY telepon DESC;  
+-----+-----+-----+  
| telepon | nama | id_pelanggan |  
+-----+-----+-----+  
| 08123456789 | Ahmad | 12300 |  
| 08123456788 | Zainuddin | 12301 |  
| 08123456787 | Hasan | 12302 |  
| 08123456786 | Hikmah | 12303 |  
| 08123456785 | Zidan | 12304 |  
+-----+-----+-----+  
5 rows in set (0.001 sec)  
MariaDB [supermarket]>
```

- Query tersebut untuk menampilkan select telepon as “telepon”, nama, id\_pelanggan  
From pelanggan  
Where telepon between ‘08123456789’  
Order by telepon desc

## 4.2 Catatan Revisi

Membuat databases, Membuat table pelanggan, penjualan, produk dan detail\_penjualan menggunakan describe, Membuat table pelanggan dan penjualan dengan menggunakan select \* from, Membuat table Alamat dan nomor hp, Membuat table pelanggan menggunakan select \* from dan where untuk menampilkan 5 id\_pelanggan, 5 nama, 5 alamat, dan 5 telepon, Membuat table pelanggan menggunakan select \* from dan where untuk menampilkan 1 id\_pelanggan, 1 nama, 1 alamat dan 1 telepon, Membuat table pelanggan menggunakan desc, Membuat table pelanggan menggunakan select \* from, where, dan like untuk menampilkan 2 id\_pelanggan, 2 nama, 2 alamat, 2 telepon, Membuat table pelanggan menggunakan select \* from, where Alamat untuk menampilkan 1 id\_pelanggan, 1 nama, 1 alamat, 1 telepon, Membuat table menggunakan select telepon as "telepon", nama, id\_pelanggan, from pelanggan, Where telepon between, order by telepon,

## 4.3 Revisi

Tidak ada yang revisi

## TUGAS 5

### PERINTAH SQL LANJUTAN

Dalam dunia supermarket, manajemen data membutuhkan sistem basis data yang kompleks dan saling terhubung. Entitas utama dalam database csupermarket meliputi pelanggan yang mencatat nama pelanggan, Alamat pelanggan, nomor hp pelanggan, entitas ke dua yaitu penjualan yang terdiri dari id\_penjualan, tanggal\_produk, id\_pelanggan, entitas ke tiga yaitu produk yang terdiri dari id\_produk, nama\_produk, harga, stok, entitas kke empat yaitu detail\_penjualan yang terdiri dari id\_detail\_penjualan, id\_penjualan, id\_id\_produk, jumlah, dan harga\_satuan.

#### 5.1 Query, Screenshot, dan Penjelasan

❖ Membuat table pelanggan dan penjualan menggunakan select \* from

- Query

```
SELECT * FROM pelanggan;  
SELECT * FROM penjualan;
```

- Hasil Query

```
MariaDB [supermarket]> SELECT * FROM penjualan;
```

id_penjualan	tanggal_penjualan	id_pelanggan
20011	2023-10-01	12300
20012	2023-10-02	12301
20013	2023-10-03	12302
20014	2023-10-04	12303
20015	2023-10-05	12304

```
5 rows in set (0.000 sec)
```

```
MariaDB [supermarket]> SELECT * FROM pelanggan;
```

Id_pelanggan	nama	alamat	telepon
12300	Ahmad	Jakarta	08123456789
12301	Zainuddin	Jogja	08123456788
12302	Hasan	Sleman	08123456787
12303	Hikmah	Bantul	08123456786
12304	Zidan	Kaliurang	08123456785

```
5 rows in set (0.000 sec)
```

- Query tersebut berfungsi Untuk menampilkan data pelanggan dan data penjualan

- ❖ Membuat table pelanggan dan penjualan menggunakan select \* from dan cross join

- Query

```
SELECT * FROM pelanggan CROSS JOIN penjualan;
```

- Hasil Query

```

MariaDB [supermarket]> SELECT * FROM pelanggan CROSS JOIN penjualan;
+-----+-----+-----+-----+-----+-----+-----+
| Id_pelanggan | nama      | alamat  | telepon | id_penjualan | tanggal_penjualan | id_pelanggan |
+-----+-----+-----+-----+-----+-----+-----+
| 12300 | Ahmad    | Jakarta | 08123456789 | 20011 | 2023-10-01 | 12300 |
| 12301 | Zainuddin | Jogja   | 08123456788 | 20011 | 2023-10-01 | 12300 |
| 12302 | Hasan    | Sleman  | 08123456787 | 20011 | 2023-10-01 | 12300 |
| 12303 | Hikmah   | Bantul  | 08123456786 | 20011 | 2023-10-01 | 12300 |
| 12304 | Zidan    | Kaliurang | 08123456785 | 20011 | 2023-10-01 | 12300 |
| 12300 | Ahmad    | Jakarta | 08123456789 | 20012 | 2023-10-02 | 12301 |
| 12301 | Zainuddin | Jogja   | 08123456788 | 20012 | 2023-10-02 | 12301 |
| 12302 | Hasan    | Sleman  | 08123456787 | 20012 | 2023-10-02 | 12301 |
| 12303 | Hikmah   | Bantul  | 08123456786 | 20012 | 2023-10-02 | 12301 |
| 12304 | Zidan    | Kaliurang | 08123456785 | 20012 | 2023-10-02 | 12301 |
| 12300 | Ahmad    | Jakarta | 08123456789 | 20013 | 2023-10-03 | 12302 |
| 12301 | Zainuddin | Jogja   | 08123456788 | 20013 | 2023-10-03 | 12302 |
| 12302 | Hasan    | Sleman  | 08123456787 | 20013 | 2023-10-03 | 12302 |
| 12303 | Hikmah   | Bantul  | 08123456786 | 20013 | 2023-10-03 | 12302 |
| 12304 | Zidan    | Kaliurang | 08123456785 | 20013 | 2023-10-03 | 12302 |
| 12300 | Ahmad    | Jakarta | 08123456789 | 20014 | 2023-10-04 | 12303 |
| 12301 | Zainuddin | Jogja   | 08123456788 | 20014 | 2023-10-04 | 12303 |
| 12302 | Hasan    | Sleman  | 08123456787 | 20014 | 2023-10-04 | 12303 |
| 12303 | Hikmah   | Bantul  | 08123456786 | 20014 | 2023-10-04 | 12303 |
| 12304 | Zidan    | Kaliurang | 08123456785 | 20014 | 2023-10-04 | 12303 |
| 12300 | Ahmad    | Jakarta | 08123456789 | 20015 | 2023-10-05 | 12304 |
| 12301 | Zainuddin | Jogja   | 08123456788 | 20015 | 2023-10-05 | 12304 |
| 12302 | Hasan    | Sleman  | 08123456787 | 20015 | 2023-10-05 | 12304 |
| 12303 | Hikmah   | Bantul  | 08123456786 | 20015 | 2023-10-05 | 12304 |
| 12304 | Zidan    | Kaliurang | 08123456785 | 20015 | 2023-10-05 | 12304 |
+-----+-----+-----+-----+-----+-----+-----+
15 rows in set (0.000 sec)

```

- Menggunakan query CROSS JOIN untuk menggabungkan baris dari tabel pelanggan dan semua baris dari tabel penjualan

- ❖ Membuat table menggunakan Menggunakan INNER JOIN Query ini menggabungkan data dari tabel pelanggan atau(b) dan tabel penjualan (p), Menampilkan kolom:ID\_pelanggan,nama,Alamat,telepon dari tabel pelanggan serta id\_penjualan dan tanggal\_penjualan dari table penjualan dan hasil nya menunjukan 5 baris data yang memiliki kecocokan di kedua tabel tersebut

- Query

```

SELECT pelanggan.id_pelanggan, pelanggan.nama,
pelanggan.alamat, pelanggan.telepon,
penjualan.id_penjualan, penjualan.tanggal_penjualan
From pelanggan
INNER JOIN penjualan ON pelanggan.id_pelanggan =
penjualan.id_pelanggan;

```



- Hasil Query

```

MariaDB [supermarket]> SELECT pelanggan.id_pelanggan, pelanggan.nama, pelanggan.alamat, pelanggan.telepon, penjualan.id_penjualan, penjualan.tanggal_penjualan
-> FROM pelanggan
-> INNER JOIN penjualan ON pelanggan.id_pelanggan = penjualan.id_pelanggan;

```

id_pelanggan	nama	alamat	telepon	id_penjualan	tanggal_penjualan
12300	Ahmad	Jakarta	08123456789	20011	2023-10-01
12301	Zainuddin	Jogja	08123456788	20012	2023-10-02
12302	Hasan	Slaman	08123456787	20013	2023-10-03
12303	Hikmah	Bantul	08123456786	20014	2023-10-04
12304	Zidan	Kaliurang	08123456785	20015	2023-10-05

5 rows in set (0.003 sec)

- Menggunakan INNER JOIN Query ini menggabungkan data dari tabel pelanggan atau(b) dan tabel penjualan (p), Menampilkan kolom:ID\_pelanggan,nama,Alamat,telepon dari tabel pelanggan serta id\_penjualan dan tanggal\_penjualan dari table penjualan dan hasil nya menunjukan 5 baris data yang memiliki kecocokan di kedua tabel tersebut

❖ Membuat table menggunakan RAIGH JOIN untuk Mengembalikan semua baris dari tabel kiri (pelanggan), dan baris yang cocok dari tabel kanan (penjualan). Jika tidak ada kecocokan, maka nilai NULL akan ditampilkan untuk kolom dari table kanan.

- Query

```

SELECT pelanggan.id_pelanggan, pelangan.nama,
pelanggan.alamat, pelanggan.telepon,
penjualan.id_penjualan, penjualan.tanggal_penjualan
From pelanggan
RAIGH JOIN penjualan ON pelanggan.id_pelanggan =
penjualan.id_pelanggan;

```

- Hasil Query

```

MariaDB [supermarket]> SELECT pelanggan.id_pelanggan, pelanggan.nama, pelanggan.alamat, pelanggan.telepon, penjualan.id_penjualan, penjualan.tanggal_penjualan
-> FROM pelanggan
-> RIGHT JOIN penjualan ON pelanggan.id_pelanggan = penjualan.id_pelanggan;

```

id_pelanggan	nama	alamat	telepon	id_penjualan	tanggal_penjualan
12300	Ahmad	Jakarta	08123456789	20011	2023-10-01
12301	Zainuddin	Jogja	08123456788	20012	2023-10-02
12302	Hasan	Slaman	08123456787	20013	2023-10-03
12303	Hikmah	Bantul	08123456786	20014	2023-10-04
12304	Zidan	Kaliurang	08123456785	20015	2023-10-05

5 rows in set (0.001 sec)

- RAIGH JOIN Mengembalikan semua baris dari tabel kiri (pelanggan), dan baris yang cocok dari tabel kanan (penjualan). Jika tidak ada kecocokan, maka nilai NULL akan ditampilkan untuk kolom dari table kanan.

- ❖ Membuat table menggunakan LEFT JOIN untuk Mengembalikan semua baris dari tabel kanan (penjualan), dan baris yang cocok dari tabel kiri (pelanggan). Jika tidak ada kecocokan, maka nilai NULL akan ditampilkan untuk kolom dari tabel kiri.

- Query

```
SELECT pelanggan.id_pelanggan, pelanggan.nama,
pelanggan.alamat, pelanggan.telepon,
penjualan.id_penjualan, penjualan.tanggal_penjualan
FROM pelanggan
LEFT JOIN penjualan ON pelanggan.id_pelanggan =
penjualan.id_pelanggan
UNION
SELECT pelanggan.id_pelanggan, pelanggan.nama,
pelanggan.alamat, pelanggan.telepon,
penjualan.id_penjualan, penjualan.tanggal_penjualan
FROM pelanggan
RIGHT JOIN penjualan ON pelanggan.id_pelanggan =
penjualan.id_pelanggan
```

- Hasil Query

```
ariaDB [supermarket]> SELECT pelanggan.id_pelanggan, pelanggan.nama, pelanggan.alamat, pelanggan.telepon, penjualan.id_penjualan, penjualan.tanggal_penjualan
-> FROM pelanggan
-> LEFT JOIN penjualan ON pelanggan.id_pelanggan = penjualan.id_pelanggan
-> UNION
-> SELECT pelanggan.id_pelanggan, pelanggan.nama, pelanggan.alamat, pelanggan.telepon, penjualan.id_penjualan, penjualan.tanggal_penjualan
-> FROM pelanggan
-> RIGHT JOIN penjualan ON pelanggan.id_pelanggan = penjualan.id_pelanggan;
```

id_pelanggan	nama	alamat	telepon	id_penjualan	tanggal_penjualan
12300	Ahmad	Jakarta	08123456789	20011	2023-10-01
12301	Zainuddin	Jogja	08123456788	20012	2023-10-02
12302	Hasan	Sleman	08123456787	20013	2023-10-03
12303	Hikmah	Bantul	08123456786	20014	2023-10-04
12304	Zidan	Kaliurang	08123456785	20015	2023-10-05

rows in set (0.001 sec)

- LEFT JOIN Mengembalikan semua baris dari tabel kanan (penjualan), dan baris yang cocok dari tabel kiri (pelanggan). Jika tidak ada kecocokan, maka nilai NULL akan ditampilkan untuk kolom dari tabel kiri.

❖ Membuat table menggunakan SELECT \* FROM Pelanggan

- Query

```
SELECT * FROM pelanggan;
```

- Hasil Query

```
MariaDB [supermarket]> SELECT * FROM pelanggan;
+-----+-----+-----+-----+
| Id_pelanggan | nama      | alamat  | telepon |
+-----+-----+-----+-----+
| 12300        | Ahmad    | Jakarta | 08123456789 |
| 12301        | Zainuddin | Jogja   | 08123456788 |
| 12302        | Hasan    | Sleman  | 08123456787 |
| 12303        | Hikmah   | Bantul  | 08123456786 |
| 12304        | Zidan    | Kaliurang | 08123456785 |
+-----+-----+-----+-----+
5 rows in set (0.001 sec)
```

- Untuk menampilkan data dalam tabel pesanan

❖ Membuat table menggunakan SELECT COUNT

- Query

```
SELECT COUNT(*) from pelanggan;
```

- Hasil Query

```
MariaDB [supermarket]> SELECT COUNT(*) FROM pelanggan;
+-----+
| COUNT(*) |
+-----+
| 5        |
+-----+
row in set (0.001 sec)
```

- Untuk menampilkan banyaknya total baris data dalam tabel pelanggan

❖ Membuat table menggunakan SELECT SUM

- Query

```
SELECT SUM(id_pelanggan) FROM pelanggan;
```

- Hasil Query

```
MariaDB [supermarket]> SELECT SUM(id_pelanggan) FROM pelanggan;
+-----+
| SUM(id_pelanggan) |
+-----+
|          61510 |
+-----+
1 row in set (0.001 sec)
```

- Untuk menampilkan atau menjumlahkan total dari kolom id\_pelanggan

#### ❖ Membuat table menggunakan SELECT AVG

- Query

```
SELECT AVG (id_pelanggan) FROM pelanggan;
```

- Hasil Query

```
MariaDB [supermarket]> SELECT AVG(id_pelanggan) FROM pelanggan;
+-----+
| AVG(id_pelanggan) |
+-----+
|       12302.0000 |
+-----+
1 row in set (0.001 sec)
```

- Untuk menampilkan nilai rata-rata dari kolom id\_pelanggan dalam tabel pelanggan

#### ❖ Membuat table menggunakan SELECT MAX

- Query

```
SELECT MAX(id_pelanggan) FROM pelanggan;
```

- Hasil Query

```
MariaDB [supermarket]> SELECT MAX(id_pelanggan) FROM pelanggan;
+-----+
| MAX(id_pelanggan) |
+-----+
|          12304 |
+-----+
1 row in set (0.000 sec)
```

- Untuk menampilkan nilai terbesar atau tertinggi dari kolom id\_pelanggan dalam tabel pelanggan.

#### ❖ Membuat table menggunakan SELECT MIN

- Query

```
SELECT MIN(id_pelanggan) FROM pelanggan;
```

- Hasil Query

```
MariaDB [supermarket]> SELECT MAX(id_pelanggan) FROM pelanggan;
+-----+
| MAX(id_pelanggan) |
+-----+
|             12304 |
+-----+
1 row in set (0.000 sec)
```

- Untuk mendapatkan nilai terkecil atau terendah dari kolom id\_pelanggan dalam tabel pelanggan

#### ❖ Membuat table menggunakan SELECT ROUND

- Query

```
SELECT ROUND (2);
```

- Hasil Query

```
MariaDB [supermarket]> SELECT ROUND(2);
+-----+
| ROUND(2) |
+-----+
|          2 |
+-----+
1 row in set (0.000 sec)
```

- Digunakan untuk membulatkan angka decimal

#### ❖ Membuat table menggunakan SELECT TRUNCATE

- Query

```
SELECT TRUNCATE(10.2);
```

- Hasil Query

```
MariaDB [supermarket]> SELECT TRUNCATE(10,2);
+-----+
| TRUNCATE(10,2) |
+-----+
|          10    |
+-----+
1 row in set (0.000 sec)
```

- Digunakan untuk memotong angka decimal (10.2) adalah angka yang di potong dan angka (2) adalah jumlah angka di belakang koma, hasil nya adalah 10.2 yang dipotong 2 angka decimal

#### ❖ Membuat table menggunakan SELECT MOD

- Query

```
SELECT MOD (0);
```

- Hasil Query

```
MariaDB [supermarket]> SELECT MOD(10,5);
+-----+
| MOD(10,5) |
+-----+
|          0 |
+-----+
1 row in set (0.000 sec)
```

- Digunakan untuk menghasilkan sisa hasil pembagian.

#### ❖ Membuat table menggunakan SELECT LOWER

- Query

```
SELECT LOWER (alamat) FROM pelanggan;
```

- Hasil Query

```
MariaDB [supermarket]> SELECT LOWER(alamat) FROM pelanggan;
+-----+
| LOWER(alamat) |
+-----+
| jakarta       |
| jogja         |
| sleman        |
| bantul        |
| kaliurang     |
+-----+
5 rows in set (0.001 sec)
```

- Digunakan untuk mrngubah kata yang ada di kolom alamat menjadi huruf kecil yang ada dalam tabel pelanggan

#### ❖ Membuat table menggunakan SELECT UPPER

- Query

```
SELECT UPPER(alamat) FROM pelanggan;
```

- Hasil Query

```
MariaDB [supermarket]> SELECT UPPER(alamat) FROM pelanggan;
+-----+
| UPPER(alamat) |
+-----+
| JAKARTA       |
| JOGJA         |
| SLEMAN        |
| BANTUL        |
| KALIURANG     |
+-----+
5 rows in set (0.000 sec)
```

- Digunakan untuk mrngubah kata yang ada di dalam kolom alamat menjadi huruf besar yang ada dalam tabel pelanggan

#### ❖ Membuat table menggunakan SELECT CONCAT

- Query

```
SELECT CONCAT (nama, ' - ', alamat);
```

- Hasil Query

```
MariaDB [supermarket]> SELECT CONCAT(nama, ' - ', alamat) FROM pelanggan;
+-----+
| CONCAT(nama, ' - ', alamat) |
+-----+
| Ahmad - Jakarta             |
| Zainuddin - Jogja          |
| Hasan - Sleman              |
| Hikmah - Bantul            |
| Zidan - Kaliurang          |
+-----+
5 rows in set (0.001 sec)
```

- Digunakan untuk Menggabungkan kata nama dan Alamar menjadi nama-alamat

#### ❖ Membuat table menggunakan SELECT SUBSTR

- Query

```
SELECT SUBSTR(nama, 1, 3) from pelanggan;
```

- Hasil Query

```
MariaDB [supermarket]> SELECT SUBSTR(nama, 1, 3) FROM pelanggan;
```

SUBSTR(nama, 1, 3)
Ahm
Zai
Has
Hik
Zid

```
5 rows in set (0.001 sec)
```

- Mengambil tiga karakter pertama dari kolom nama.

❖ Membuat table menggunakan SELECT LENGTH

- Query

```
SELECT LENGTH(nama);
```

- Hasil Query

```
MariaDB [supermarket]> SELECT LENGTH(nama) FROM pelanggan;
```

LENGTH(nama)
5
9
5
6
5

```
5 rows in set (0.001 sec)
```

- Menghitung panjang atau jumlah karakter pada kolom nama untuk setiap baris.

❖ Membuat table menggunakan SELECT INSTR

- Query

```
SELECT INSTR(Alamat, 'a');
```

- Hasil Query



```
MariaDB [supermarket]> SELECT INSTR(alamat, 'a') FROM pelanggan;
+-----+
| INSTR(alamat, 'a') |
+-----+
|                    |
|                    |
|                    |
|                    |
|                    |
+-----+
5 rows in set (0.000 sec)
```

- Menemukan posisi pertama kemunculan huruf 'a' dalam kolom alamat untuk setiap baris.

#### ❖ Membuat table menggunakan SELECT LPAD

- Query

```
SELECT LPAD(alamat, 10, '##');
```

- Hasil Query

```
MariaDB [supermarket]> SELECT LPAD(alamat, 10, '##') FROM pelanggan;
+-----+
| LPAD(alamat, 10, '##') |
+-----+
| ###Jakarta              |
| #####Jogja              |
| #####Sleman             |
| #####Bantul             |
| #Kaliurang              |
+-----+
5 rows in set (0.001 sec)
```

- Menambahkan karakter ## di sebelah kiri nilai alamat hingga panjang totalnya menjadi 10 karakter.

#### ❖ Membuat table menggunakan SELECT TRIM

- Query

```
SELECT TRIM(' ' from alamat);
```

- Hasil Query

```
MariaDB [supermarket]> SELECT TRIM(' ' FROM alamat) FROM pelanggan;
+-----+
| TRIM(' ' FROM alamat) |
+-----+
| Jakarta                |
| Jogja                  |
| Sleman                  |
| Bantul                  |
| Kaliurang              |
+-----+
5 rows in set (0.000 sec)
```

- Menghapus spasi kosong di awal dan akhir nilai pada kolom alamat.

#### ❖ Membuat table menggunakan SELECT REPLACE

- Query

```
SELECT REPLACE ('pelanggan', 'la', 'u');
```

- Hasil Query

```
MariaDB [supermarket]> SELECT REPLACE(alamat, 'a', 'o') FROM pelanggan;
+-----+
| REPLACE(alamat, 'a', 'o') |
+-----+
| Jokorto                    |
| Jogjo                      |
| Slemon                     |
| Bontul                     |
| Koliurong                  |
+-----+
5 rows in set (0.000 sec)
```

- Mengganti semua huruf 'a' dengan huruf 'o' pada nilai kolom alamat.

#### ❖ Membuat table menggunakan SELECT NOW()

- Query

```
SELECT now();
```

- Hasil Query

```
MariaDB [supermarket]> SELECT NOW();
+-----+
| NOW() |
+-----+
| 2024-11-12 17:05:03 |
+-----+
1 row in set (0.000 sec)
```

- Mengambil tanggal dan waktu saat ini dalam format YYYY-MM-DD HH:MM:SS.

❖ Membuat table mennggunakan SELECT CURDATE()

- Query

```
SELECT curdate();
```

- Hasil Query

```
MariaDB [supermarket]> SELECT CURDATE();
+-----+
| CURDATE() |
+-----+
| 2024-11-12 |
+-----+
1 row in set (0.000 sec)
```

- Mengambil tanggal saat ini dalam format YYYY-MM-DD.

❖ Membuat table menggunakan SELECT CURTIME()

- Query

```
SELECT curtime();
```

- Hasil Query

```
MariaDB [supermarket]> SELECT CURTIME();
+-----+
| CURTIME() |
+-----+
| 17:05:24 |
+-----+
1 row in set (0.000 sec)
```

- Mengambil waktu saat ini dalam format HH:MM:SS.

## 5.2 Catatan Revisi

Membuat table full join

## 5.3 Revisi

```
MariaDB [supermarket]> SELECT p.Id_pelanggan, p.nama, p.alamat, p.telepon, j.id_penjualan, j.tanggal_penjualan FROM pel
JOIN penjualan j ON p.Id_pelanggan = j.id_pelanggan UNION SELECT p.Id_pelanggan, p.nama, p.alamat, p.telepon, j.id_penj
gal_penjualan FROM pelanggan p RIGHT JOIN penjualan j ON p.Id_pelanggan = j.id_pelanggan;
+-----+-----+-----+-----+-----+-----+
| Id_pelanggan | nama      | alamat  | telepon | id_penjualan | tanggal_penjualan |
+-----+-----+-----+-----+-----+-----+
| 12300        | Ahmad     | Jakarta | 08123456789 | 20011        | 2023-10-01         |
| 12301        | Zainuddin | Jogja   | 08123456788 | 20012        | 2023-10-02         |
| 12302        | Hasan     | Sleman  | 08123456787 | 20013        | 2023-10-03         |
| 12303        | Hikmah    | Bantul  | 08123456786 | 20014        | 2023-10-04         |
| 12304        | Zidan     | Kaliurang | 08123456785 | 20015        | 2023-10-05         |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.001 sec)
```