**LAPORAN PRAKTIKUM**

**PRAKTIKUM DATABASE PERTEMUAN 4**

Disusun untuk Memenuhi Matakuliah Praktikum {Nama Matakuliah}

Dibimbing oleh : Sulaibatul Aslamiyah, M.Kom

****

Oleh:

**RAFLI RAHMAN.EFENDY**

**1124102162**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA**

**SEKOLAH TINGGI ILMU KOMPUTER PGRI BANYUWANGI**

**2025**

# **LEMBAR PENGESAHAN**

# **LAPORAN PRAKTIKUM**

Matakuliah : Praktikum Database

Oleh : Rafli Rahman.Efendy

NIM : 1124102162

**Telah disahkan pada**

Hari : RABU

Tanggal : 16/03/2025

**Mengetahui/ Menyetujui :**

Dosen Pengampu Mata Kuliah

**Sulaibatul Aslamiyah, M.Kom**

NIDN. 0712058304

**KATA PENGANTAR**

Berisikan kata pengantar yang dituliskan oleh mahasiswa untuk laporan praktikum matakuliah yang diikuti.

**MODUL PRAKTIKUM 3**

1. **Capaian Praktikum Pertemuan 1**

PEMBUATAN DATABASE KOMPLEKS DAN RELASI TABLE dalam satu cangkupan

1. **Indikator Capaian**

* Membuat Tabel ‘db\_Kantor’,‘ customer’,’Transasksi’, dan ’detail\_Transakasi’ di buat dalam table mysql tanpa eror.
* Membuat Sruktur table yang sesui dengan yang di buat/study kasus nya(kolom tipe data, dan constraint).

1. **Landasan Teori**

Jelaskan landasan teori yang digunakan dalam penyelesaian studi kasus. pembuatan tabel - tabel didalam database, dimana antar tabel saling terkait satu sama lainnya

1. **Pelaksanaan Praktikum**

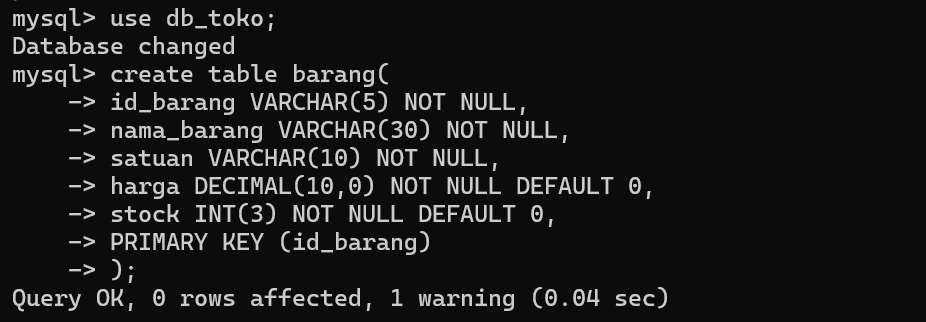
Landasan teori MySQL adalah Structured Query Language (SQL) sebagai bahasa interaktif untuk mengelola data. MySQL adalah sistem manajemen basis data relasional (RDBMS) yang menggunakan SQL untuk menjalankan fungsinya.

1. **Percobaan Pertama**

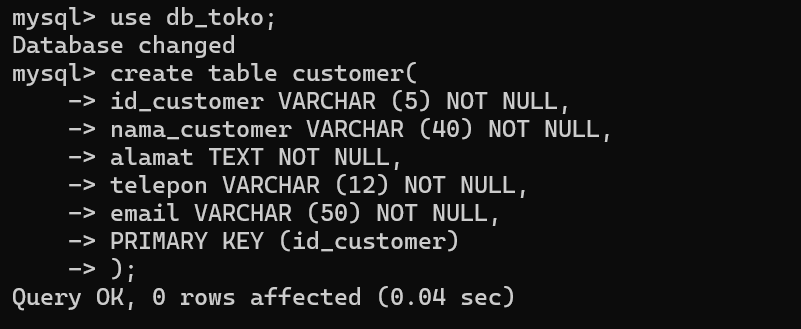
Terlebih dahulu dibuatkan sebuah database untuk menampung tabel-tabel tersebut. Database yang dibuat diberikan nama db\_toko. Selanjutnya terdapat 4 tabel yang dipergunakan antara lain tabel customer, transaksi, barang, dan detail\_transaksi.

Dengan Membuat database db\_toko: ”create database if not exists db\_toko;”

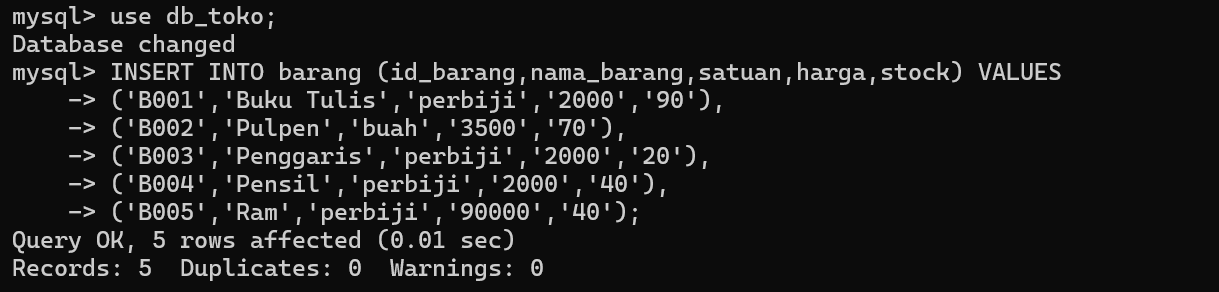
**Di buatkan Table Barang:**



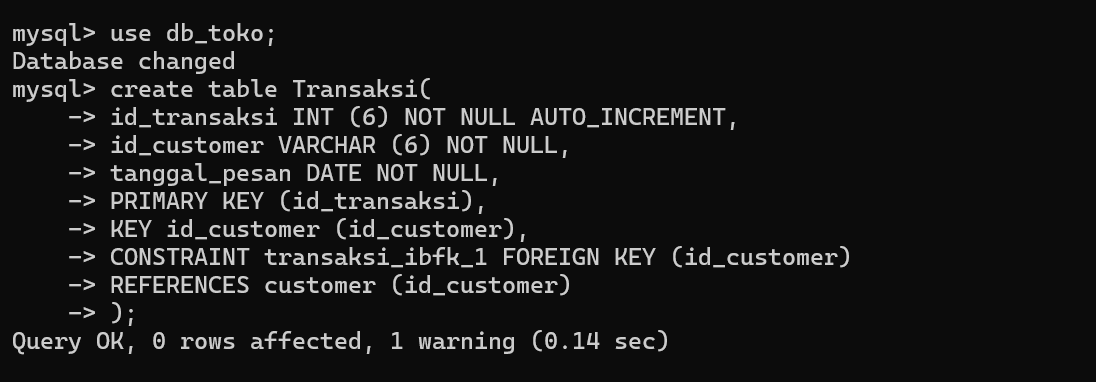
**Di buatkan Table Costomer:**

****

**Di buatkan Table Transaksi:**

****

**Di buatkan Detail\_Transaksi:**

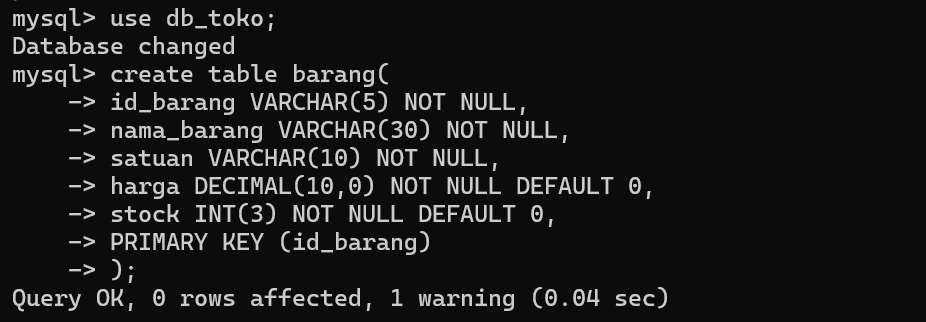
****

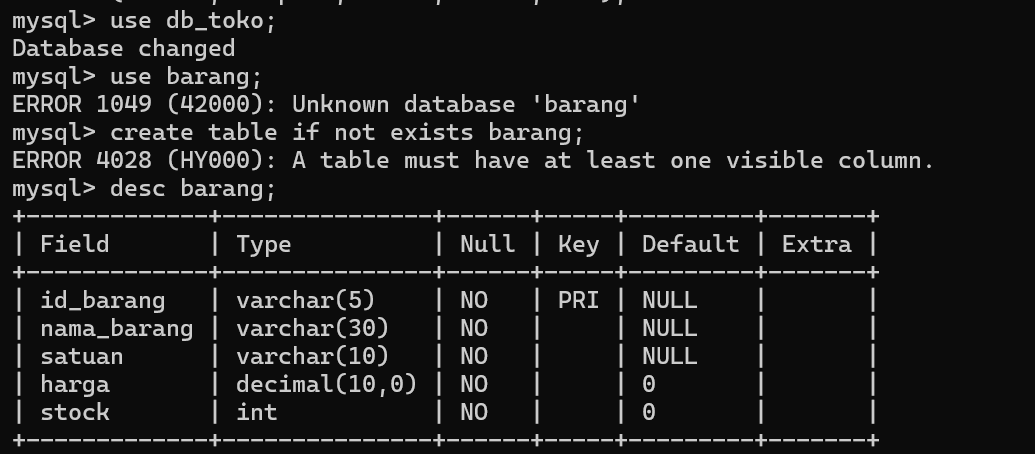
**Deskripsi :**

Untun menampilkan semua hasil yang telah di coding di Mysql membuat ‘• create database if not exists db\_toko;’ lalu terdapat 4 tabel yang dipergunakan antara lain tabel customer, transaksi, barang, dan detail\_transaksi

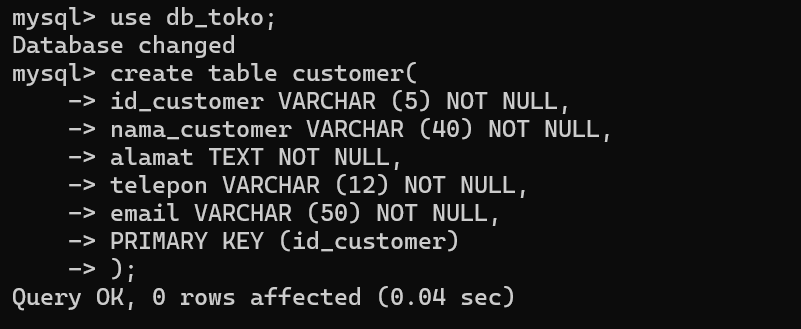
**Hasil Uji Coba :**

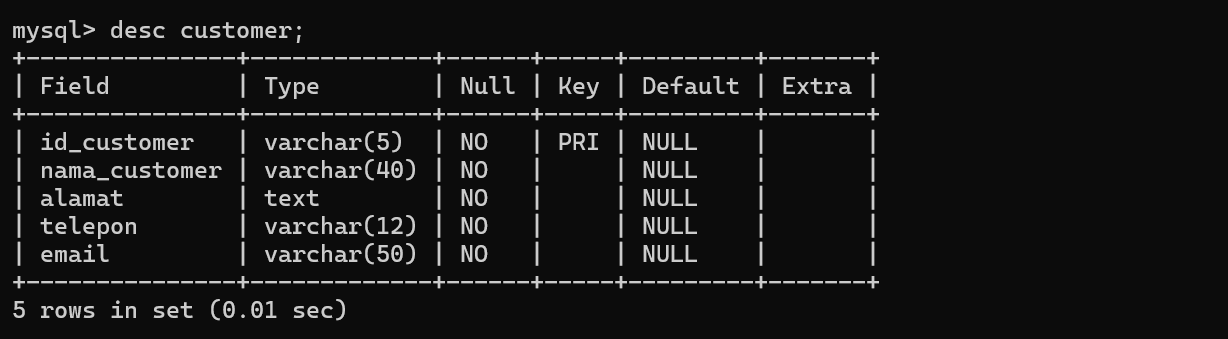
**Hasil dari Table Barang:**

****

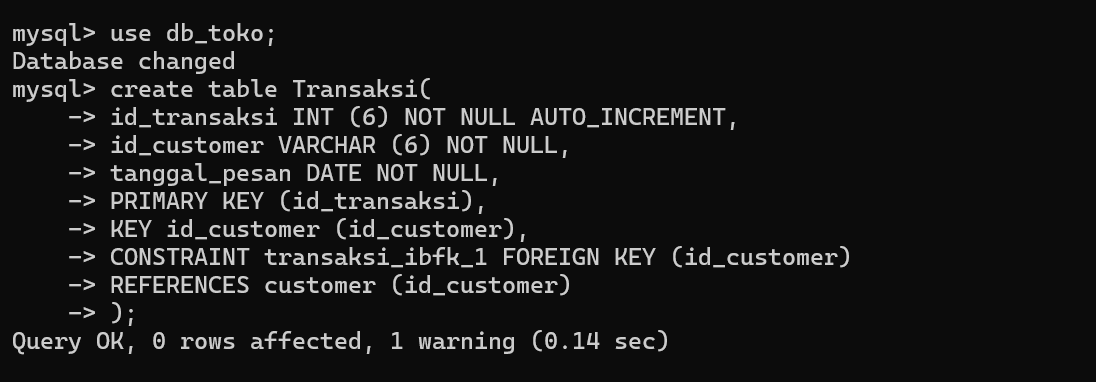
****

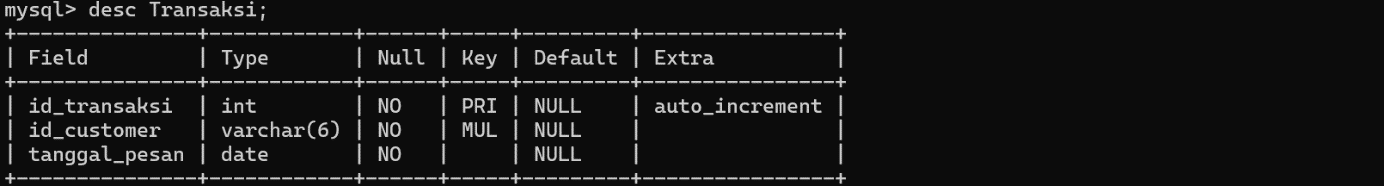
**Hasil dari Table Costomer:**

****

****

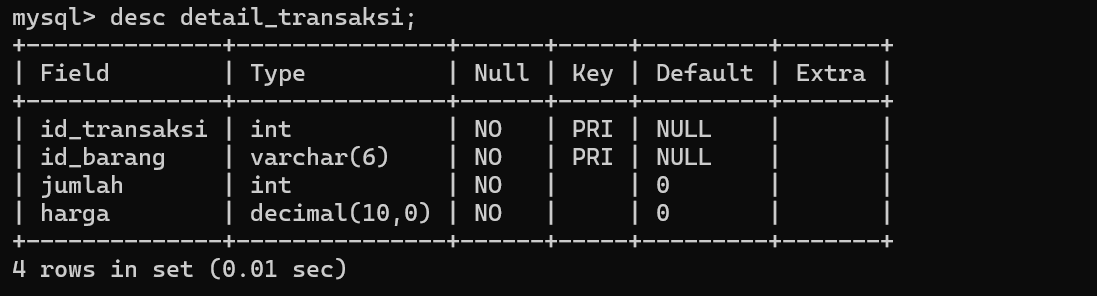
**Hasil dari Table Transaksi:**

****

****

**Hasil dari Table detail\_transaksi:**

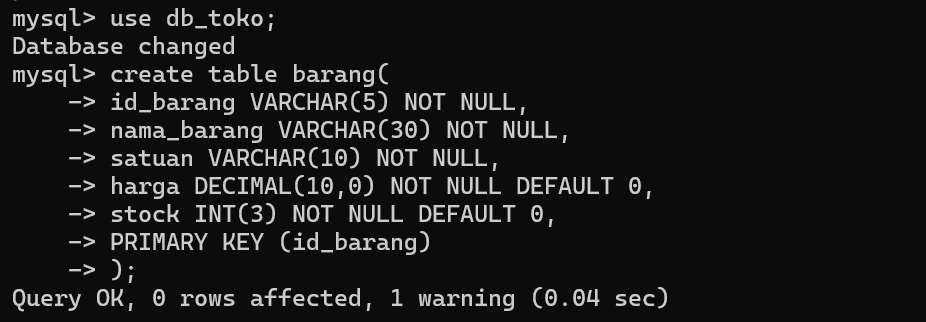
****

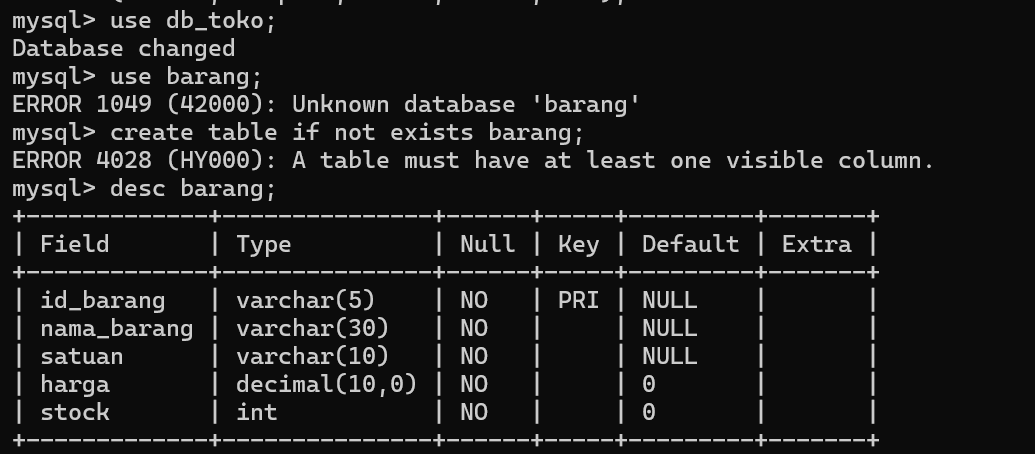
****

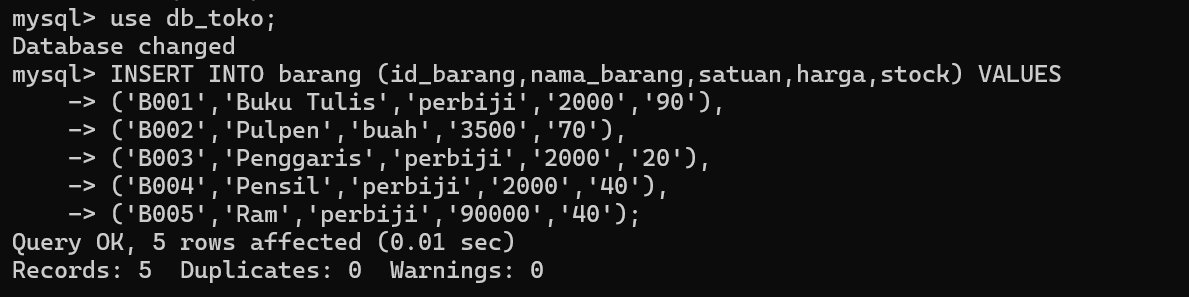
Hasil uji coba untuk menapilkan semua data tersebut tidak di buat satu persatu table melainkan di buat dilalam data “db\_toko”, dan dibuatkan table/desc table lain tabel customer, transaksi, barang, dan detail\_transaksi.

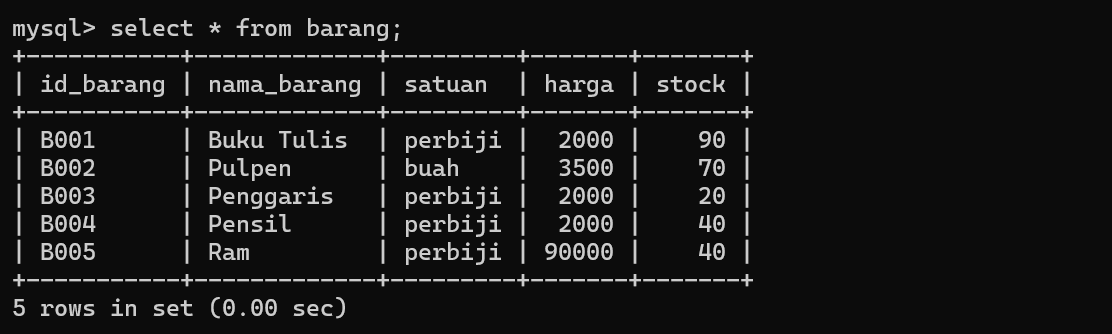
**1.4.2**

**1. Hasil Uji Coba: Membuat Table Barang beserta Isi didalam nya**

****

****

****

****

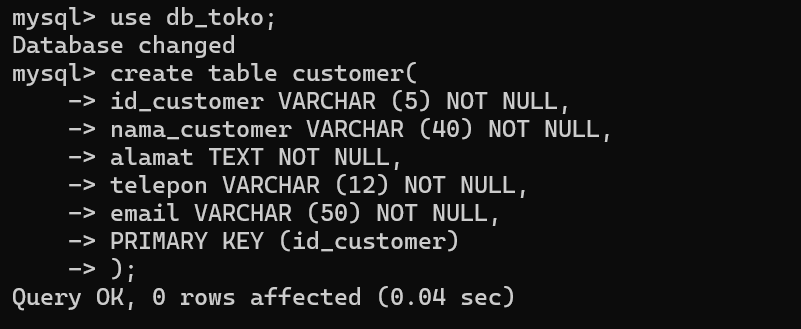
**Deskripsi :**

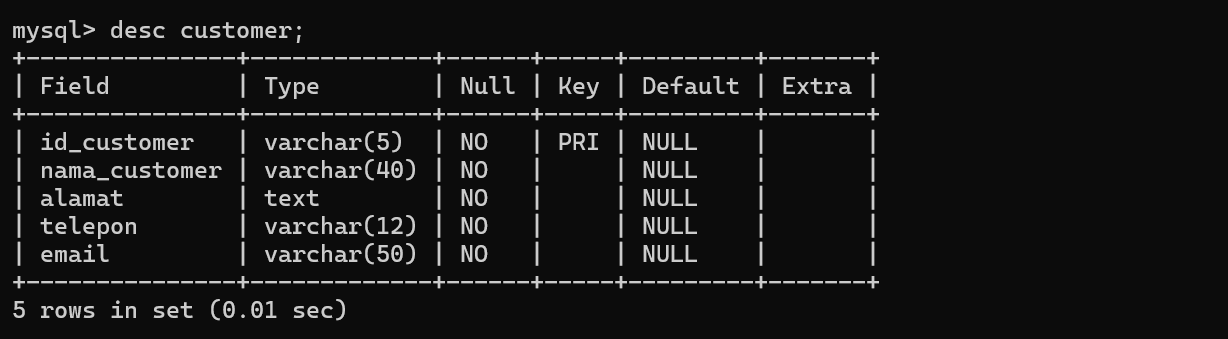
Untun menampilkan semua hasil yang telah di coding di Mysql membuat ‘create table baran’ lalu table yang telah di buat ketik ‘use db\_toko;’ agar bisa di gunakan untuk mengisi table database yang sudah di buat.

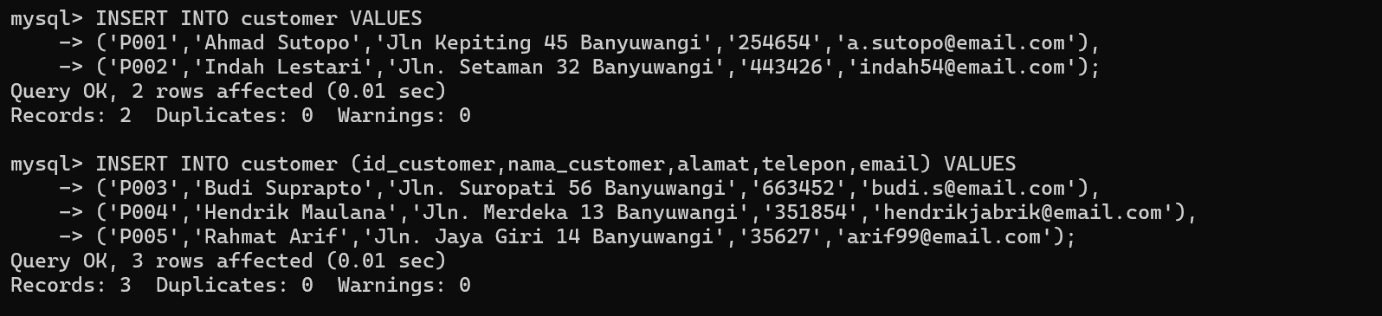
**Hasil Uji Coba :**

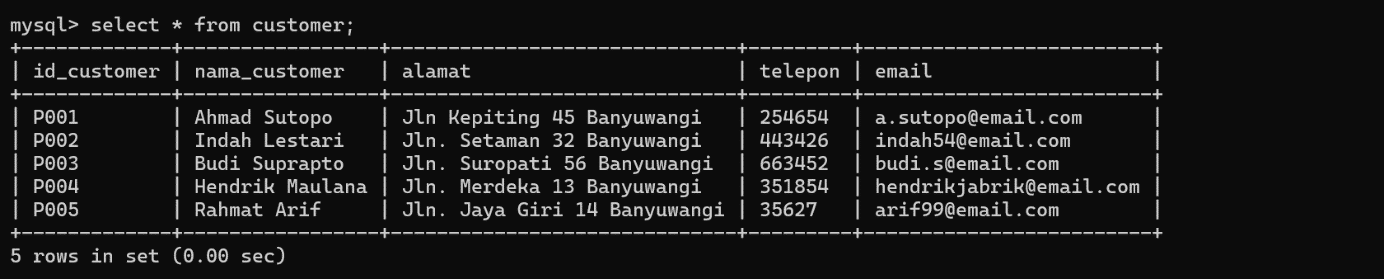
Hasil uji coba untuk menapilkan data siswa dengan menggunakan perintah ‘desc barang;’ akan menampilkan seluruh table yang telah di buat, umtuk menampilkan apa yang sudah di isi didalam table ‘select \* from barang;’

**2. Hasil Uji Coba: Membuat Table Costomer beserta Isi didalam nya**

****







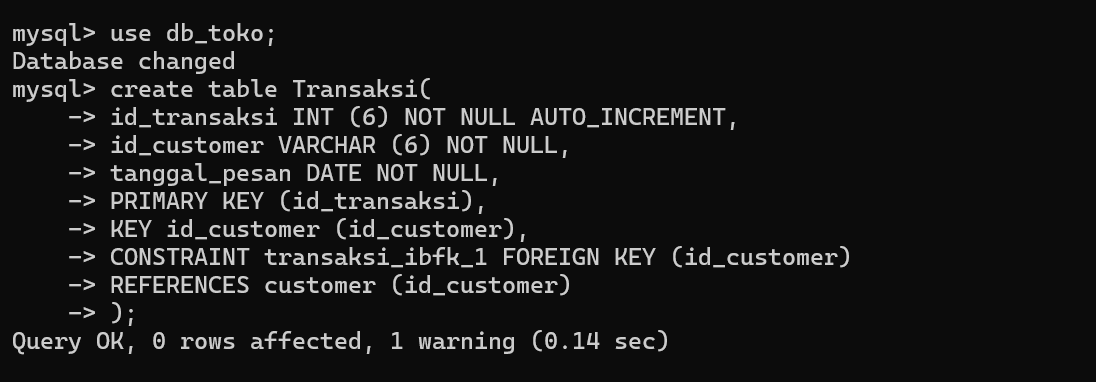
**Deskripsi :**

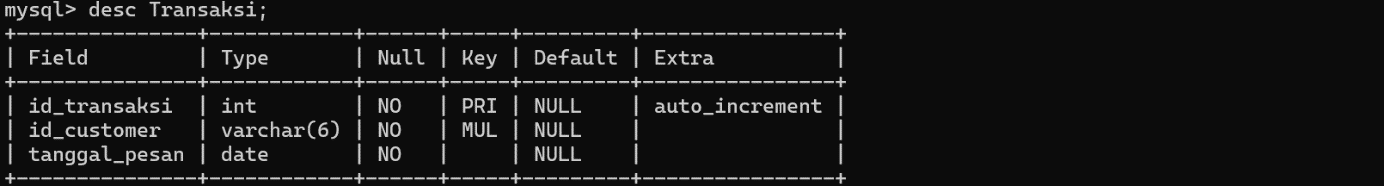
Untun menampilkan semua hasil yang telah di coding di Mysql membuat ‘create table customer’ lalu table yang telah di buat ketik ‘use db\_toko;’ agar bisa di gunakan untuk mengisi table database yang sudah di buat.

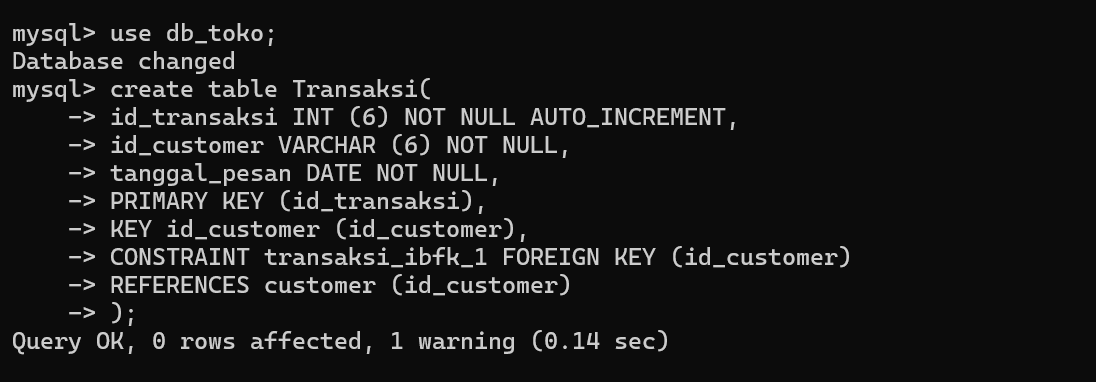
**Hasil Uji Coba :**

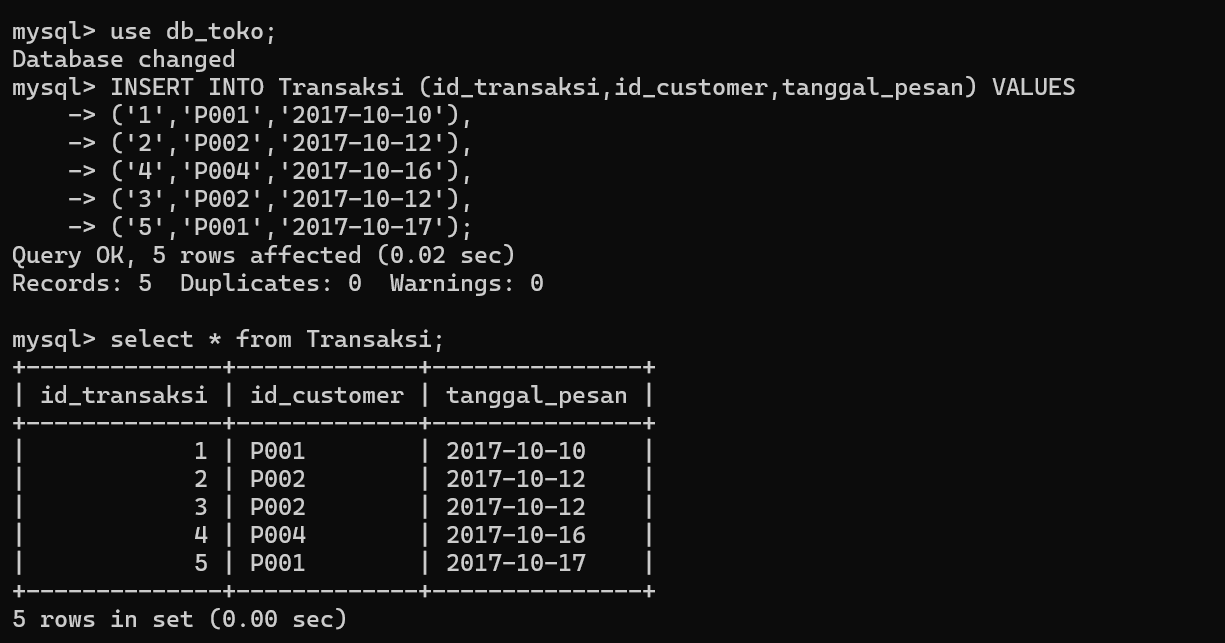
Hasil uji coba untuk menapilkan data siswa dengan menggunakan perintah ‘desc costomer;’ akan menampilkan seluruh table yang telah di buat, umtuk menampilkan apa yang sudah di isi didalam table ‘select \* from costomer;’

**3. Hasil Uji Coba: Membuat Table Transaksi beserta Isi didalam nya**

****







**Deskripsi :**

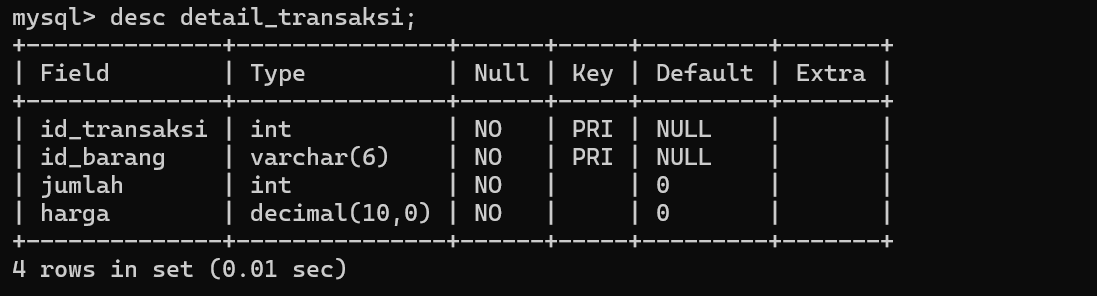
Untun menampilkan semua hasil yang telah di coding di Mysql membuat ‘create table Transaksi’ lalu table yang telah di buat ketik ‘use db\_toko;’ agar bisa di gunakan untuk mengisi table database yang sudah di buat.

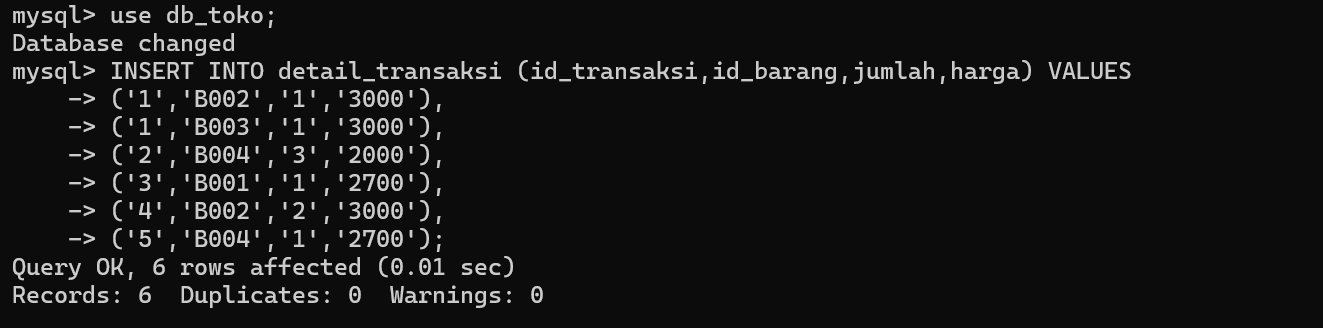
**Hasil Uji Coba :**

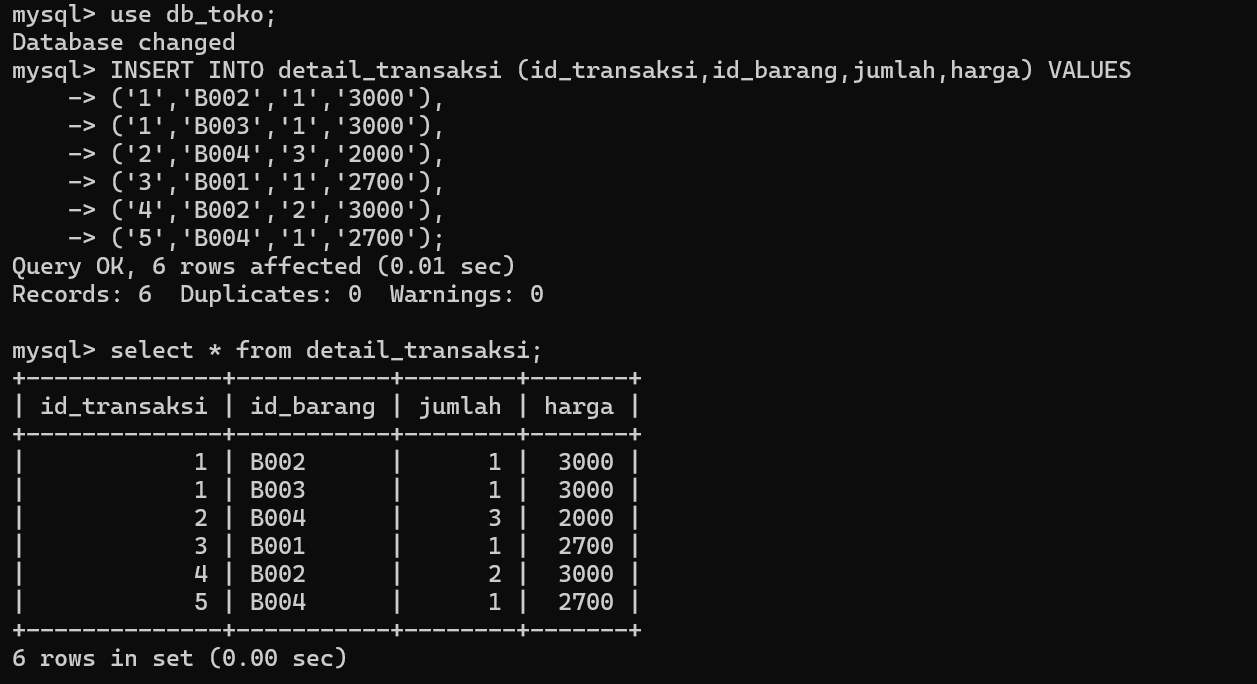
Hasil uji coba untuk menapilkan data siswa dengan menggunakan perintah ‘desc costomer;’ akan menampilkan seluruh table yang telah di buat, umtuk menampilkan apa yang sudah di isi didalam table ‘select \* from Transaksi;’

**4. Hasil Uji Coba: Membuat Table detail\_transaksi beserta Isi didalam nya**

****

****





**Deskripsi :**

Untun menampilkan semua hasil yang telah di coding di Mysql membuat ‘create table detail\_transaksi’ lalu table yang telah di buat ketik ‘use db\_toko;’ agar bisa di gunakan untuk mengisi table database yang sudah di buat.

**Hasil Uji Coba :**

Hasil uji coba untuk menapilkan data siswa dengan menggunakan perintah ‘desc costomer;’ akan menampilkan seluruh table yang telah di buat, umtuk menampilkan apa yang sudah di isi didalam table ‘select \* from detail\_transaksi;’

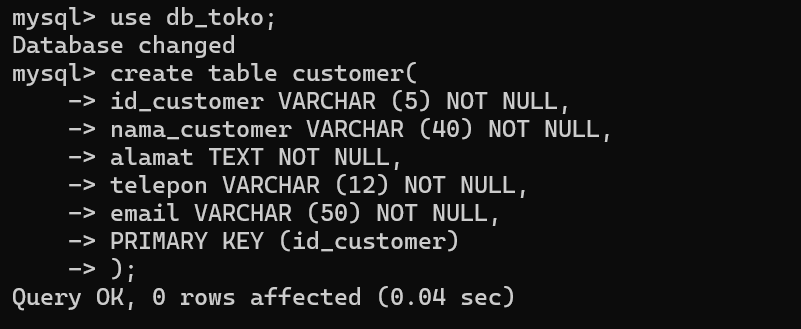
**1.4.3 Tugas Praktikum Database Pada Modul 3**

Terlebih dahulu dibuatkan sebuah database untuk menampung tabel-tabel tersebut. Database yang dibuat diberikan nama db\_toko. Selanjutnya terdapat 4 tabel yang dipergunakan antara lain tabel customer, transaksi, barang, dan detail\_transaksi. Query dalam pembuatan tabel-tabel tersebut dijabarkan sebagai berikut:

**Membuat database db\_toko**

* create database if not exists db\_toko;

**# Membuat Create table customer:**



**Membuat table customer**

* create table customer (

id\_customer VARCHAR (5) NOT NULL,

nama\_customer VARCHAR (40) NOT NULL,

alamat TEXT NOT NULL,

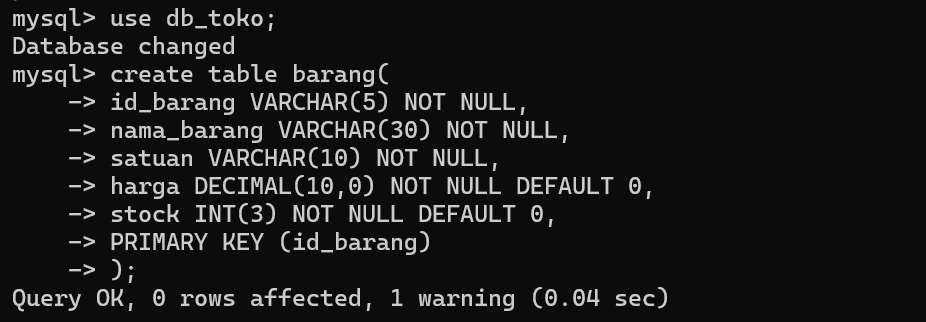
telepon VARCHAR (20) NOT NULL,

email VARCHAR (50) NOT NULL,

PRIMARY KEY (id\_customer)

)

**# Membuat Create table Barang:**

****

**Membuat table barang**

* create table barang (

id\_barang VARCHAR (5) NOT NULL,

nama\_barang VARCHAR (30) NOT NULL,

satuan VARCHAR (10) NOT NULL,

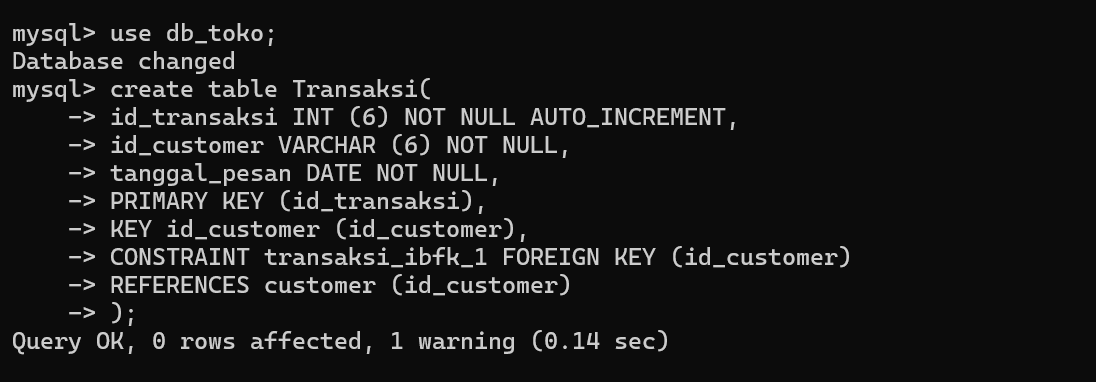
harga DECIMAL (10,0) NOT NULL DEFAULT '0',

stock INT (3) NOT NULL DEFAULT '0',

PRIMARY KEY (id\_barang)

)

**# Membuat Create table Transaksi:**

****

**Membuat table transaksi**

* create table transaksi (

id\_transaksi INT (5) NOT NULL AUTO\_INCREMENT,

id\_customer VARCHAR (5) NOT NULL,

tgl\_pesan DATE NOT NULL,

PRIMARY KEY (id\_transaksi),

KEY id\_customer (id\_customer),

CONSTRAINT transaksi\_ibfk\_1 FOREIGN KEY (id\_customer)

REFERENCES customer (id\_customer)

)

**# Membuat Create table detail\_transaksi:**

****

**Membuat table detail\_transaksi**

* create table detail\_transaksi (

id\_transaksi INT (5) NOT NULL,

id\_barang VARCHAR (5) NOT NULL,

jumlah INT (5) NOT NULL DEFAULT '0',

harga DECIMAL (10,0) NOT NULL DEFAULT '0',

PRIMARY KEY (id\_transaksi, id\_barang),

KEY FK\_detail\_transaksi (id\_barang),

KEY id\_transaksi (id\_transaksi),

CONSTRAINT FK\_detail\_transaksi FOREIGN KEY (id\_barang)

REFERENCES barang (id\_barang),

CONSTRAINT FK\_detail\_transaksi2 FOREIGN KEY (id\_transaksi)

REFERENCES transaksi (id\_transaksi)

)

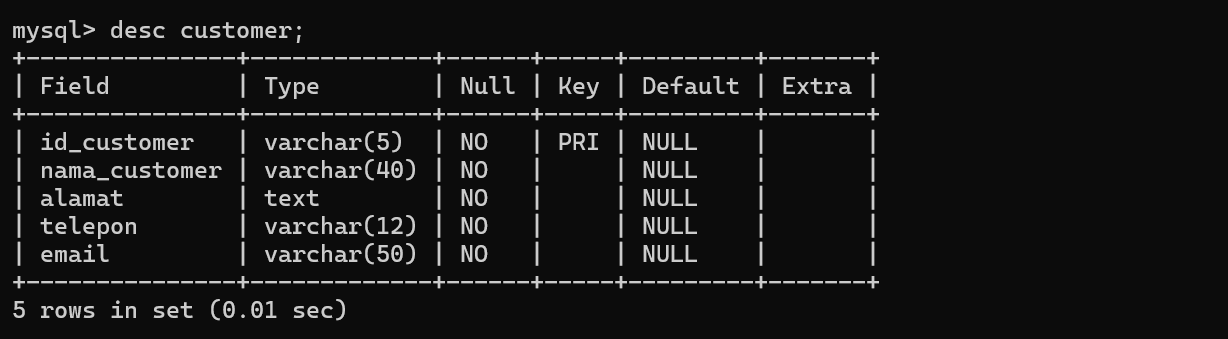
**Deskripsi :**

Perhatikan pada tabel transaksi dan tabel detail\_transaksi, dimana pada tabel transaksi terdapat satu field yang mengandung Foreign Key yaitu id\_customer yang berhubungan dengan tabel customer. Dan pada tabel detail\_transaksi terdapat 2 Foreign Key yaitu id\_barang yang terhubung dengan tabel barang dan id\_transaksi yang terhubung dengan tabel transaksi.

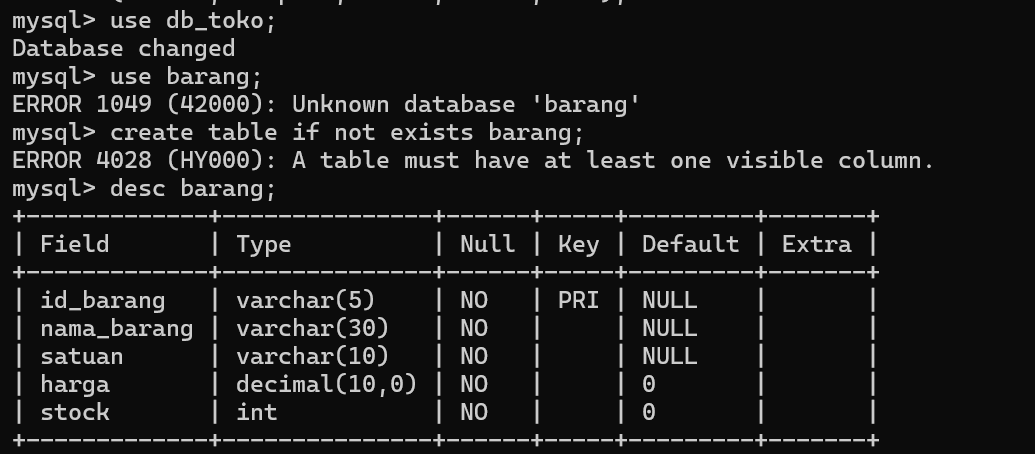
**Hasil Uji Coba :**

Meyatukan semua table yang di hubungkan dalam satu table

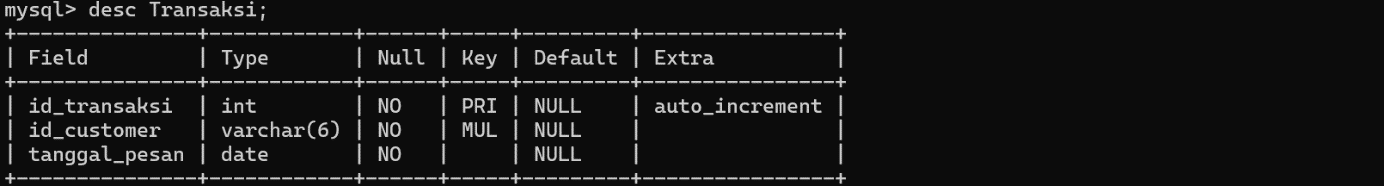
**# DESC Customer**



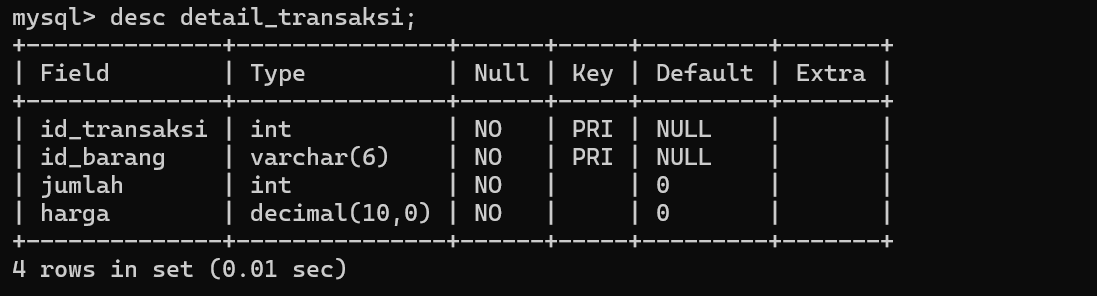
**# DESC Barang**



**# DESC Transaksi**

****

**# DESC detail\_transaksi**

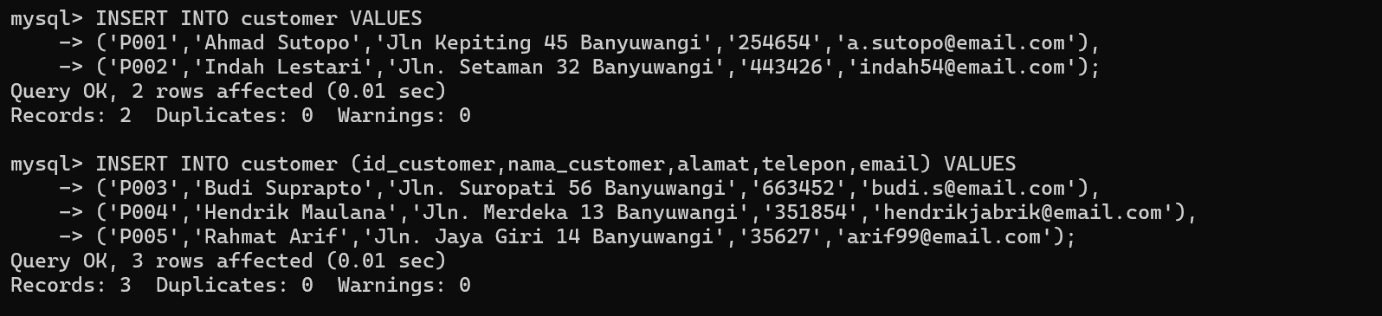
****

**Hasil Uji Coba 1 :**

kolom atau field pada suatu tabel yang berfungsi sebagai kunci tamu dari tabel lain. Foreign Key sangat berguna bila bekerja dengan banyak tabel yang saling berelasi satu sama lain.

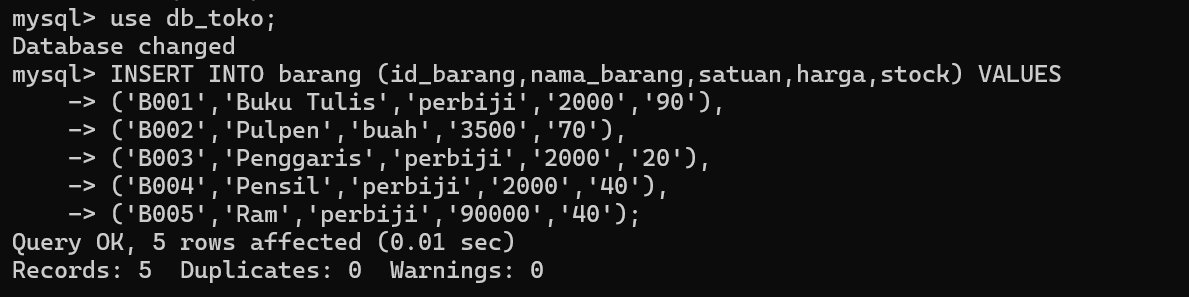
Isikan data pada setiap tabel tersebut. Query yang dipergunakan adalah sebagai berikut:

**# Membuat atau mengisi table Insert Data Customer**

****

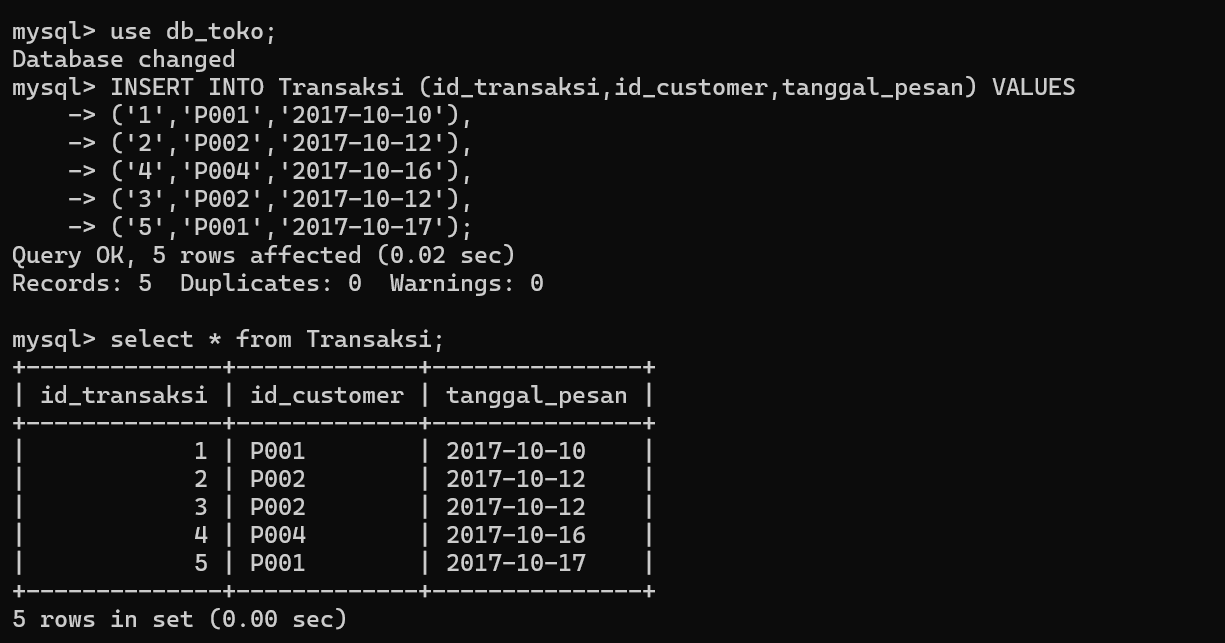
|  |
| --- |
| * INSERT INTO customer VALUES   ('P001', 'Ahmad Sutopo', 'Jln Kepiting 45 Banyuwangi', '254654', 'a.sutopo@email.com'),  ('P002', 'Indah Lestari', 'Jln. Setaman 32 Banyuwangi', '443426', 'indah54@email.com'); |

**# Membuat atau mengisi table Insert Data Barang**

****

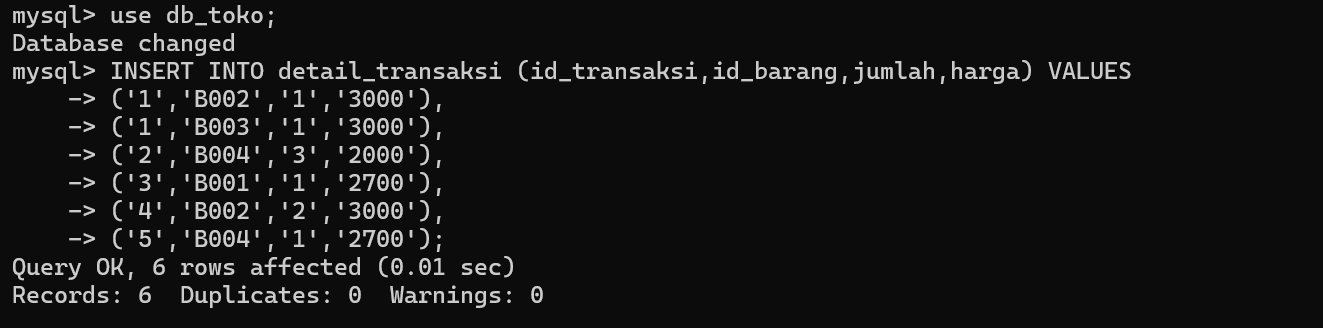
|  |
| --- |
| * INSERT INTO barang (id\_barang, nama\_barang, satuan, harga, stock) VALUES   ('B001', 'Buku Tulis', 'perbiji', '2000','90'),  ('B002', 'Pulpen', 'buah', '3500', '70'),  ('B003', 'Penggaris', 'perbiji', '2000','20'),  ('B004', 'Pensil', 'perbiji', '2000','40'),  ('B005', 'Ram', 'perbiji', '90000','40'); |

**# Membuat atau mengisi table Insert Data Transaksi**

****

|  |
| --- |
| * INSERT INTO Transaksi (id\_transaksi, id\_customer, tanggal\_pesan) VALUES   ('1','P001', '2017-10-10'),  ('2', 'P002', '2017-10-12'),  ('4','P004', '2017-10-16'),  ('3','P002', '2017-10-12'),  ('5', 'P001', '2017-10-17'); |

**# Membuat atau mengisi table Insert Data detail\_transaksi**

****

|  |
| --- |
| * INSERT INTO detail\_transaksi (id\_transaksi, id\_barang, jumlah, harga) VALUES   ('1','B002','1','3000'),  ('1','B003','1', '3000'),  ('2', 'B004', '3', '2000'),  ('3','B001','1','2700'),  ('4','B002','2', '3000'),  ('5', 'B004', '1', '2700'); |

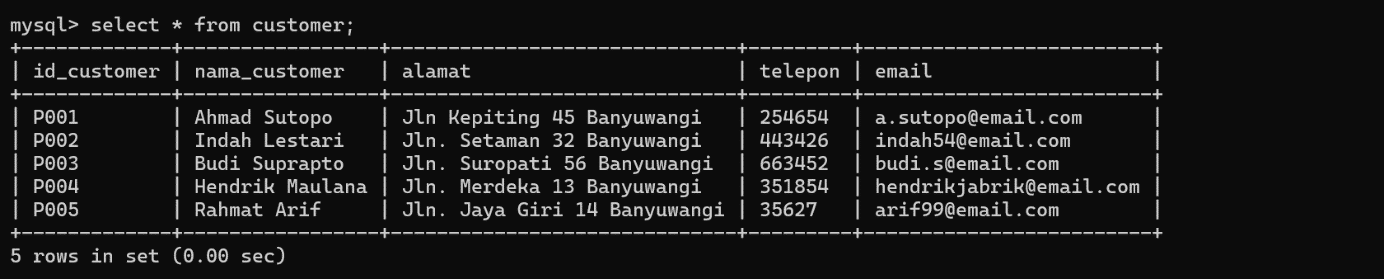
**Deskripsi :**

Perhatikan pada tabel transaksi dan tabel detail\_transaksi, dimana pada tabel transaksi terdapat satu field yang mengandung Foreign Key yaitu id\_customer yang berhubungan dengan tabel customer. Dan pada tabel detail\_transaksi terdapat 2 Foreign Key yaitu id\_barang yang terhubung dengan tabel barang dan id\_transaksi yang terhubung dengan tabel transaksi.

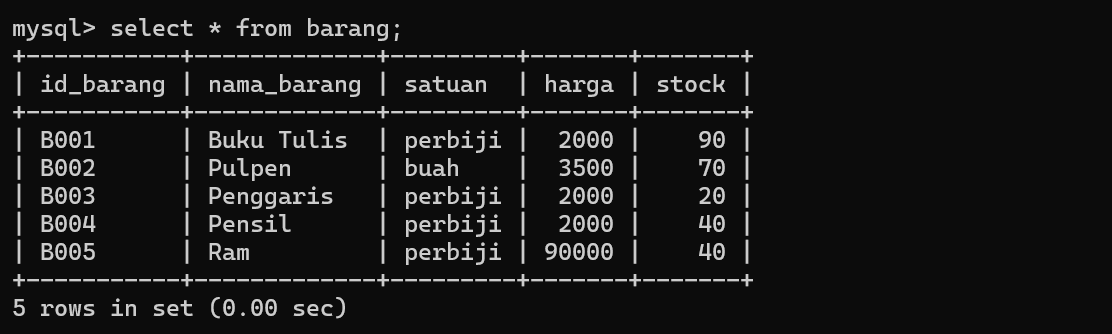
**Hasil Uji Coba :**

Meyatukan semua table yang di hubungkan dalam satu table

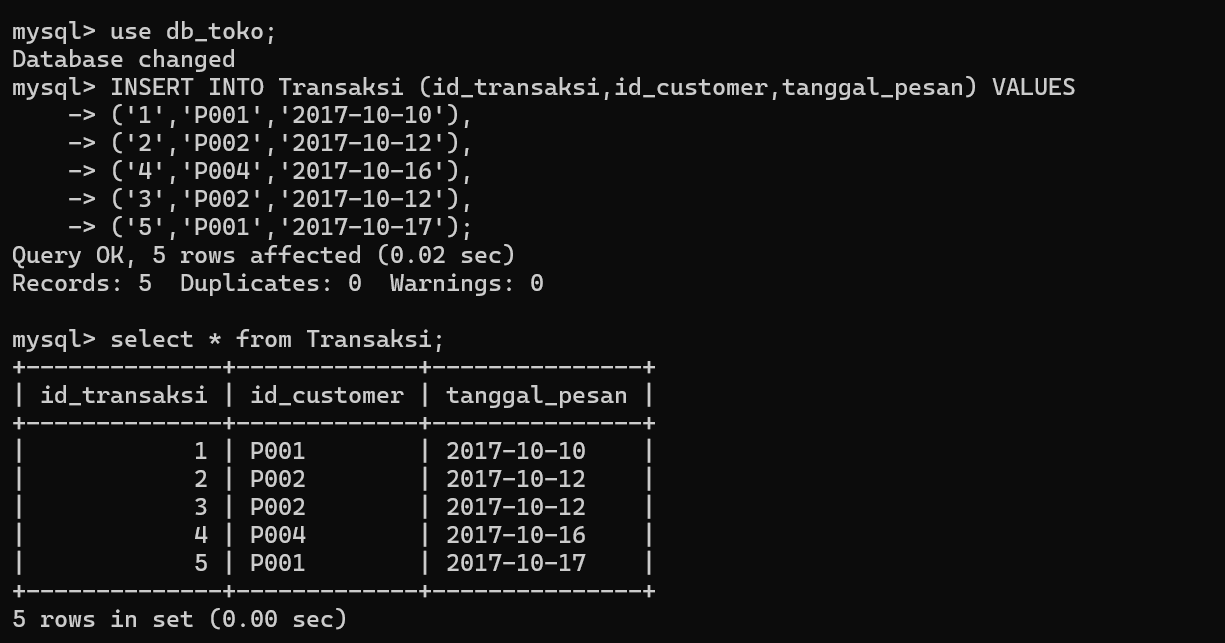
**# select \* from costomer**

****

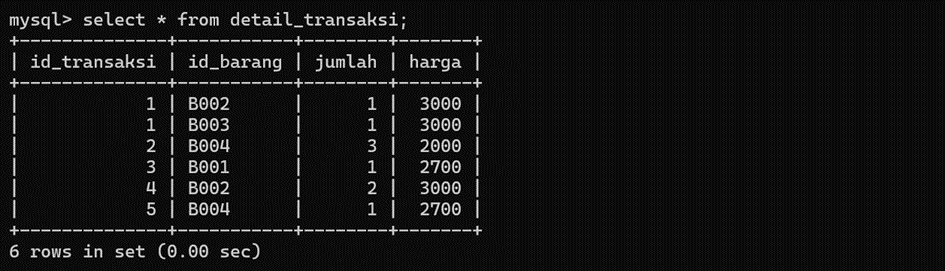
**# select \* from Barang**

****

**# select \* from Transaksi**

****

**# select \* from detail\_transaksi**

****

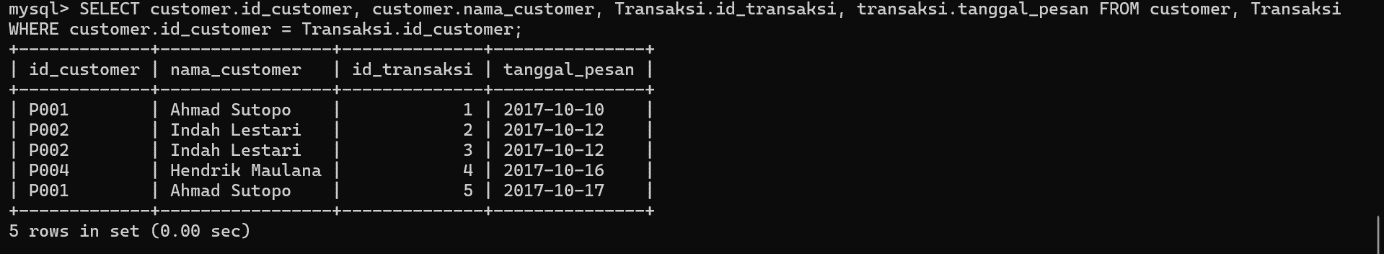
Di dalam suatu RDBMS, dalam satu database dapat terdiri dari beberapa tabel. Masing-masing tabel tersebut berhubungan satu sama lain atau dengan kata lain memiliki relasi. Relasi antartabel dapat berupa relasi 1-1, 1-M, atau M-N. Sebagai contoh terlihat pada gambar pemodelan data konseptual (class diagram) di atas. Tabel customer berhubungan dengan transaksi, transaksi dengan barang, dan sebagainya. Pada praktisnya, terkadang dibutuhkan tampilan data dari beberapa tabel sekaligus. Misal, diinginkan untuk menampilkan nama customer dengan transaksi yang pernah dilakukannya. Dari contoh tersebut, dapat dilakukan penggabungan minimal dua tabel, yaitu customer dan transaksi. Untuk menggabungkan 2 (dua) atau lebih tabel, dapat menggunakan bentuk perintah JOIN.

**1.4.4 Tugas Praktikum Database Pada Modul 3**

**Inner Join**

Dengan inner join, tabel akan digabungkan dua arah, sehingga tidak ada data yang NULL di satu sisi. Cara yang pertama dengan menggunakan where. Berikut syntak perintahnya:

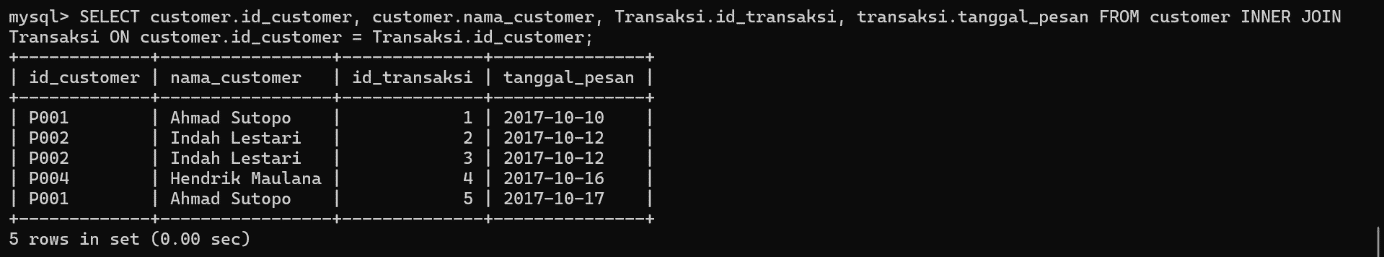
**Where**



|  |
| --- |
| * Menggabung dua tabel dengan kondisi Where:   SELECT tabel1.\*, tabel2.\* FROM tabel1, tabel2 WHERE tabel1.PK=tabel2.FK;  Atau  SELECT customer.id\_customer, customer.nama\_customer, Transaksi.id\_transaksi, transaksi.tanggal\_pesan FROM customer, Transaksi  WHERE customer.id\_customer Transaksi.id\_customer; |

**Inner Join**

Jika dengan menggunakan Inner Join, maka query perintahnya adalah sebagai berikut:

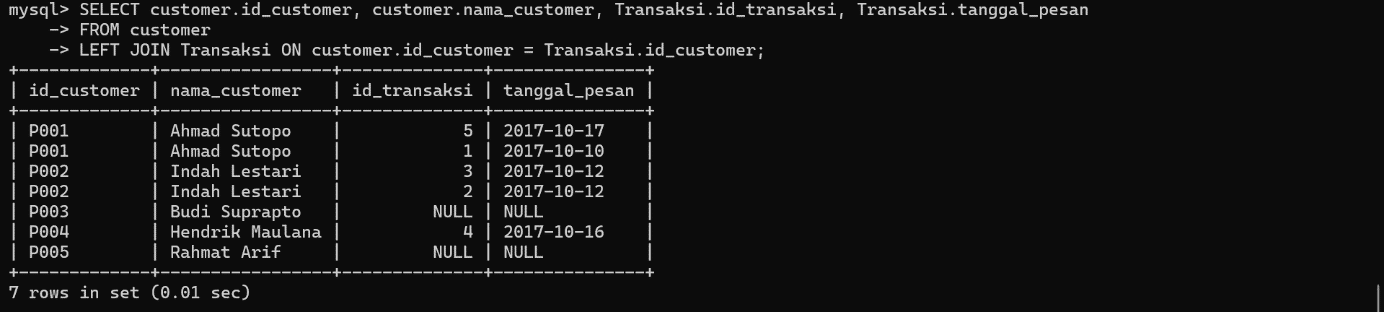


|  |
| --- |
| * Menggabung dua tabel dengan Inner Join:   SELECT tabel1.\*, tabel2.\* FROM tabel1 INNER JOIN tabel2 ON tabel1.PK=tabel2.FK;  Atau  SELECT customer.id\_customer, customer.nama\_customer, Transaksi.id\_transaksi, transaksi.tanggal pesan FROM customer INNER JOIN  Transaksi ON customer.id\_customer Transaksi.id\_customer; |

**Outer Join**

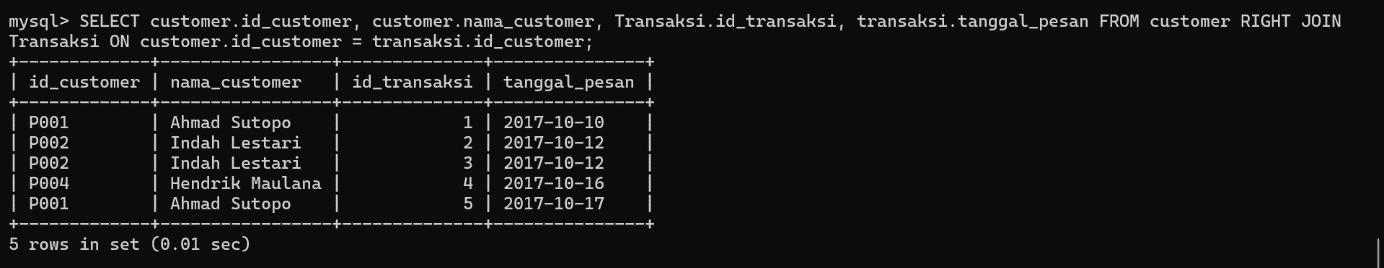
Dengan outer join, tabel akan digabungkan satu arah, sehingga memungkinkan terdapat data yang NULL (kosong) di satu sisi. Sebagai contoh semisal akan menampilkan daftar pelanggan yang pernah melakukan transaksi. Outer join terbagi menjadi dua yaitu right join dan left join. Perintah query:

**Left join**

****

|  |
| --- |
| * Menggabung tabel dengan left join:   SELECT tabel1.\*, tabel2.\* FROM tabel1 LEFT JOIN tabel2 ON tabel1.PK=tabel2.FK;  Atau  SELECT customer.id\_customer, customer.nama\_customer, Transaksi.id\_transaksi, Transaksi, tanggal\_pesan  FROM customer  LEFT JOIN Transaksi ON customer.id\_customer = Transaksi.id\_customer; |

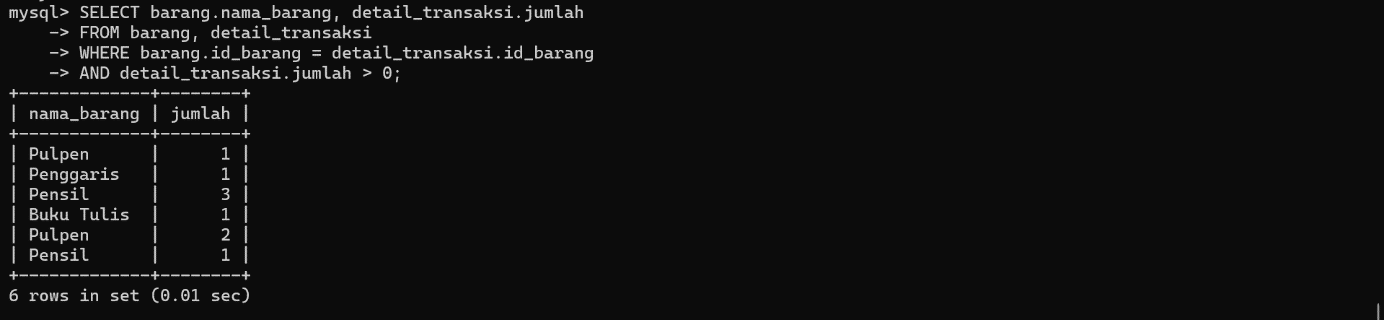
**Right Join**

****

|  |
| --- |
| * Menggabung tabel dengan right Join   SELECT tabel1.\*, tabel2.\* FROM tabel1 RIGHT JOIN tabel2 ON tabel1.PK=tabel2.FK;  Atau  SELECT customer.id\_customer, customer.nama\_customer, Transaksi.id\_transaksi, transaksi.tanggal\_pesan FROM customer RIGHT JOIN  Transaksi ON customer.id\_customer transaksi.id\_customer; |

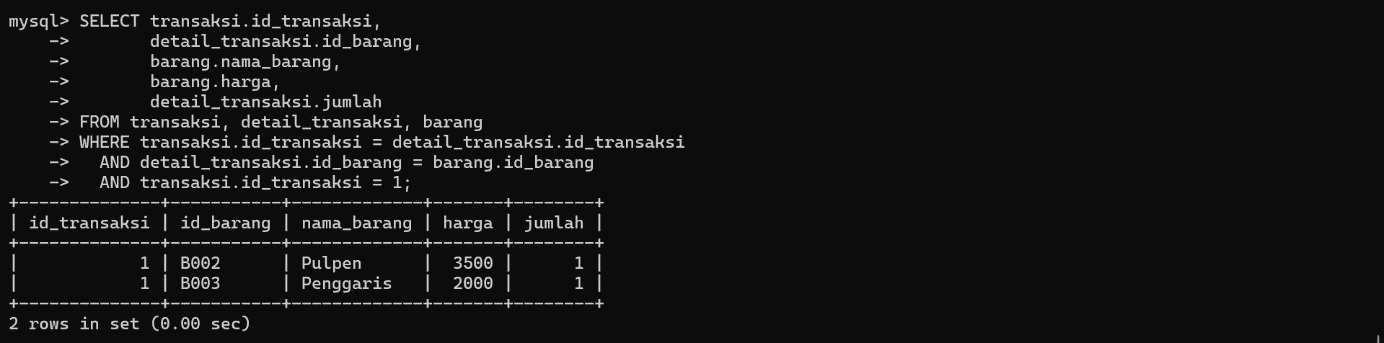
**3.3 Soal Latihan Soal Modul 3 Pertemuan 3**

Tampilkan nama barang serta jumlah barang yang telah terjual dari tabel **transaksi** dan **barang**. (menggunakan format **Where** atau Join). Hasil tabel seperti berikut:

****

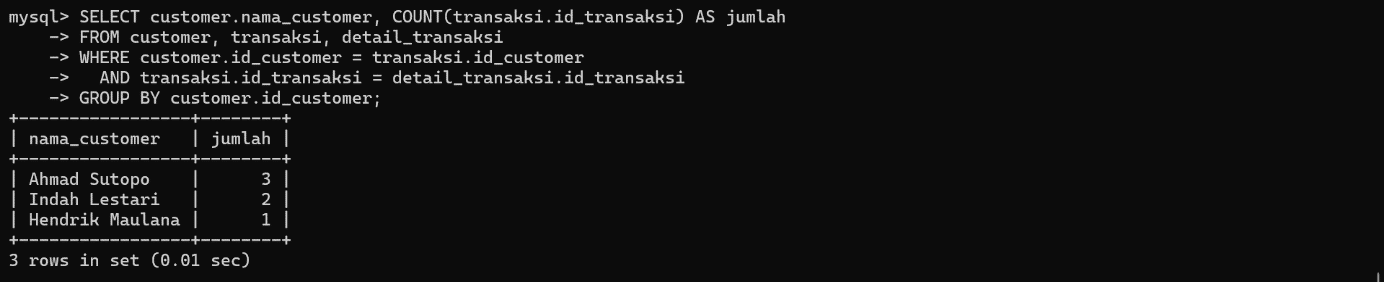
|  |
| --- |
| SELECT barang.nama\_barang, detail\_transaksi.jumlah  FROM barang, detail\_transaksi  WHERE barang.id\_barang detail = transaksi.id\_barang  AND detail\_transaksi.jumlah > 0; |

Gabungkan 3 tabel (**transaksi, detail\_transaksi,** dan **barang**) agar dapat menampilkan barang yang dipesan, nama, barang, serta harga untuk pemesanan dengan id transaksi = 1. (menggunakan format **Where** atau **Join**). Hasil tabel seperti berikut:



|  |
| --- |
| SELECT transaksi.id\_transaksi,  detail transaksi.id\_barang,  barang.nama\_barang,  barang.harga,  -> detail\_transaksi.jumlah  -> FROM transaksi, detail\_transaksi, barang  -> WHERE transaksi.id\_transaksi detail\_transaksi.id\_transaksi  -> AND detail\_transaksi.id\_barang = barang.id\_barang  AND transaksi.id\_transaksi = 1; |

Gabungkan 3 tabel (**customer, transaksi,** dan **detail\_transaksi**) agar dapat menampilkan data customer serta jumlah **transaksi** yang telah dilakukan oleh customer tersebut. (menggunakan format **Where** atau **Join**). Hasil tabel seperti berikut:



|  |
| --- |
| SELECT customer.nama\_customer, COUNT(transaksi.id\_transaksi) AS jumlah  -> FROM customer, transaksi, detail\_transaksi  -> WHERE customer.id\_customer transaksi.id\_customer  AND transaksi.id\_transaksi = detail\_transaksi.id\_transaksi  GROUP BY customer.id\_customer; |

Apa perbedaan penggabungan tabel menggunakan kondisi where dan join? Dan apa perbedaan antara left join dan right join? Jelaskan berdasarkan pengertian anda sendiri beserta query pembuatan tabel diatas. Diusahakan tidak plagiarism / kesamaan artikel.

**---JAWABAN---**

* **Perbedaan Penggabungan Tabel dengan WHERE dan JOIN**

Secara fungsional, JOIN dan penggabungan menggunakan WHERE bisa memberikan hasil yang sama, khususnya untuk INNER JOIN, tapi ada beberapa perbedaan penting:

|  |  |
| --- | --- |
| JOIN | WHERE |
| Lebih terstruktur dan eksplisit dalam menggabungkan tabel. | Kurang eksplisit, karena kondisi penggabungan ditulis di bagian WHERE bersama kondisi filter. |
| Mendukung berbagai jenis join seperti LEFT JOIN,RIGHT JOIN, dll. | Biasanya hanya digunakan untuk INNER JOIN. |
| Lebih mudah dibaca untuk query kompleks. | Bisa jadi membingungkan ketika melibatkan banyak kondisi. |

* **Perbedaan LEFT JOIN dan RIGHT JOIN**

|  |  |
| --- | --- |
| LEFT JOIN | RIGHT JOIN |
| Mengambil semua data dari tabel kiri, dan mencocokkan dengan tabel kanan. Jika tidak cocok, data dari kanan jadi NULL. | Mengambil semua data dari tabel kanan, dan mencocokkan dengan tabel kiri. Jika tidak cocok, data dari kiri jadi NULL. |
| SELECT \* FROM A LEFT JOIN B → semua dari A tetap tampil. | SELECT \* FROM A RIGHT JOIN B → semua dari B tetap tampil. |

**Link Githup Saya:**

**https://github.com/PwRafli/SP2.1-Praktikum-Database**

**BAB 1**

**PENUTUP**

**Kesimpulan**

teori MySQL adalah Structured Query Language (SQL) sebagai bahasa interaktif untuk mengelola data. MySQL adalah sistem manajemen basis data relasional (RDBMS) yang menggunakan SQL untuk menjalankan fungsinya..

**Saran**

Saran saya tidak ada perubahan pembelajaran sama seperti si Semester 1 dan masih masuk di praktikum kali kecuali pada pengisian laporan praktikum database ke word itu yang perlu di tanyakan dan cara pengisian laporan untuk kedepannya.

**DAFTAR PUSTAKA**

Tuliskan rujukan yang anda gunakan baik website maupun buku seperti contoh dibawah.

1. Tim Asisten Dosen. 2014. Modul 1 Pengenalan Sistem Operasi, Ide Visual C++, Dan Algoritma Pemrograman. Malang: Unversitas Negeri Malang.
2. Program Konversi Suhu (online)

<http://bondanoky.blogspot.com/2012/10/program-konversi-suhu-c.html>*.* Di akses 8 September.

**LAMPIRAN**

Berisikan syntax atau gambar yang dibutuhkan dalam tiap pertemuan praktikum.