**LAPORAN AKHIR PRAKTIKUM**

Mata Praktikum : SISTEM BASIS DATA 1

Kelas : 3IA11

Praktikum ke- : 5

Tanggal : 22/11/2024

Materi : QUERY EQUI

NPM : 51422161

Nama : MUHAMMAD TARMIDZI BARIQ

Ketua Asisten : SEBASTIAN DWI

Jumlah Lembar : 12



**LABORATORIUM TEKNIK INFORMATIKA**

**UNIVERSITAS GUNADARMA**

**2024**

Jelaskan kembali query – query (baris per baris) yang digunakan pada activity

CREATE TABLE BARANG(

ID\_BARANG SERIAL PRIMARY KEY NOT NULL,

NAMA\_BARANG VARCHAR (255),

STOK INT,

HARGA INT

);

Untuk membuat tabel baru bernama barang

ID\_BARANG SERIAL PRIMARY KEY NOT NULL

* Id\_barang adalah nama kolom pertama.
* Serial adalah tipe data yang secara otomatis menghasilkan nilai yang unik dan bertambah setiap kali ada data baru yang dimasukkan. Ini sering digunakan untuk kolom yang berfungsi sebagai id unik untuk setiap baris.
* Primary key memastikan bahwa kolom id\_barang tidak memiliki nilai duplikat, dan setiap nilai di dalamnya harus unik.
* Not null berarti kolom ini tidak boleh berisi nilai kosong (null). Jadi, setiap entri harus memiliki nilai untuk id\_barang.

NAMA\_BARANG VARCHAR(255)

* Nama\_barang adalah kolom kedua yang menyimpan nama barang dalam bentuk string teks.
* Varchar(255) berarti kolom ini dapat menyimpan teks dengan panjang maksimum 255 karakter.

STOK INT

* Stok adalah kolom untuk menyimpan jumlah stok barang dalam bentuk angka bulat (integer).
* Int adalah tipe data untuk menyimpan bilangan bulat.

HARGA INT

* Harga adalah kolom untuk menyimpan harga barang dalam bentuk angka bulat (integer).
* Int berarti tipe data untuk kolom ini juga adalah bilangan bulat.

INSERT INTO BARANG VALUES

(1,'BAJU',3,15000),

(2,'CELANA',5,10000),

(3,'JAKET',6,25000),

(4,'KEMEJA',2,30000),

(5,'SEPATU',3,20000);

INSERT INTO BARANG

Perintah insert into barang digunakan untuk menambahkan data ke dalam tabel barang.

VALUES

Nilai-nilai yang akan dimasukkan ke dalam tabel. Setiap set nilai dalam tanda kurung mewakili satu baris data yang akan ditambahkan.

Kolom pertama (id\_barang): nilai id barang yang unik.

Kolom kedua (nama\_barang): nama barang dalam bentuk string (misalnya 'baju').

Kolom ketiga (stok): jumlah stok barang dalam bentuk integer (misalnya 3).

Kolom keempat (harga): harga barang dalam bentuk integer (misalnya 15000).

-- SCREENSHOT 1

SELECT \* FROM BARANG;

Menampilkan semua data di tabel barang

CREATE TABLE PEMBELI(

ID\_PEMBELI SERIAL PRIMARY KEY NOT NULL,

NAMA VARCHAR (255) NOT NULL,

ALAMAT VARCHAR (255)

);

Untuk membuat tabel baru bernama pembeli

ID\_PEMBELI SERIAL PRIMARY KEY NOT NULL,

* Id\_pembelil adalah nama kolom pertama.
* Serial adalah tipe data yang secara otomatis menghasilkan nilai yang unik dan bertambah setiap kali ada data baru yang dimasukkan. Ini sering digunakan untuk kolom yang berfungsi sebagai id unik untuk setiap baris.
* Primary key memastikan bahwa kolom id\_barang tidak memiliki nilai duplikat, dan setiap nilai di dalamnya harus unik.
* Not null berarti kolom ini tidak boleh berisi nilai kosong (null). Jadi, setiap entri harus memiliki nilai untuk id\_barang.

NAMA VARCHAR (255) NOT NULL,

* Nama adalah kolom kedua yang menyimpan nama dalam bentuk string teks.
* Varchar(255) berarti kolom ini dapat menyimpan teks dengan panjang maksimum 255 karakter.
* Not null berarti kolom ini tidak boleh berisi nilai kosong (null). Jadi, setiap entri harus memiliki nilai untuk nama.

ALAMAT VARCHAR (255)

* Alamat adalah kolom kedua yang menyimpan alamat dalam bentuk string teks.
* Varchar(255) berarti kolom ini dapat menyimpan teks dengan panjang maksimum 255 karakter.

INSERT INTO PEMBELI VALUES

(1,'GERALD','JL.MARZUKI'),

(2,'FARIZI SETYA','JL.ANDAYANI'),

(3,'FREYA AZZAHRA','JL. MAHUNTEN'),

(4,'ANNISA SALSABILA','JL.ZENDANA'),

(5,'ANNE SAMANDA','JL.TINIZOA');

INSERT INTO PEMBELI

Perintah insert into pembeli digunakan untuk menambahkan data ke dalam tabel pembeli.

VALUES

Nilai-nilai yang akan dimasukkan ke dalam tabel. Setiap set nilai dalam tanda kurung mewakili satu baris data yang akan dimasukkan.

Data yang dimasukkan

Setiap set data memiliki tiga nilai yang sesuai dengan kolom dalam tabel pembeli:

* Kolom pertama (misalnya id\_pembeli): nilai id pembeli yang unik.
* Kolom kedua (misalnya nama\_pembeli): nama pembeli dalam bentuk string.
* Kolom ketiga (misalnya alamat\_pembeli): alamat pembeli dalam bentuk string.

-- SCREENSHOT 2

SELECT \* FROM PEMBELI

Menampilkan semua data di tabel pembeli

CREATE TABLE PENJUALAN(

ID\_PENJUALAN SERIAL PRIMARY KEY NOT NULL,

CABANG VARCHAR (100),

HARI\_JUAL VARCHAR(20) NOT NULL,

PENJUAL VARCHAR(100),

JUMLAH\_BELI INT NOT NULL,

ID\_BARANG INT NOT NULL,

ID\_PEMBELI INT NOT NULL,

FOREIGN KEY(ID\_BARANG) REFERENCES BARANG(ID\_BARANG),

FOREIGN KEY(ID\_PEMBELI) REFERENCES PEMBELI(ID\_PEMBELI)

);

ID\_PENJUALAN SERIAL PRIMARY KEY NOT NULL

* Id\_penjualan adalah kolom untuk menyimpan id penjualan yang unik.
* Serial berarti tipe data ini akan otomatis bertambah (auto-increment) untuk setiap baris baru.
* Primary key memastikan nilai di kolom ini adalah unik dan tidak ada duplikasi.
* Not null memastikan kolom ini tidak boleh berisi nilai kosong.

CABANG VARCHAR(100)

* Cabang adalah kolom untuk menyimpan nama cabang penjualan dalam bentuk string dengan panjang maksimum 100 karakter.

HARI\_JUAL VARCHAR(20) NOT NULL

* Hari\_jual adalah kolom untuk menyimpan hari atau tanggal penjualan dalam format string.
* Not null berarti kolom ini harus diisi, tidak boleh kosong.

PENJUAL VARCHAR(100)

* Penjual adalah kolom untuk menyimpan nama penjual (pegawai atau kasir) yang melakukan transaksi.

JUMLAH\_BELI INT NOT NULL

* Jumlah\_beli adalah kolom untuk menyimpan jumlah barang yang dibeli dalam bentuk angka bulat (integer).
* Not null memastikan kolom ini tidak boleh kosong.

ID\_BARANG INT NOT NULL

* Id\_barang adalah kolom yang menyimpan id barang yang dijual.
* Tipe data int digunakan untuk menyimpan nilai numerik id barang.
* Not null berarti kolom ini harus diisi dengan nilai id barang yang valid.

ID\_PEMBELI INT NOT NULL

* Id\_pembeli adalah kolom yang menyimpan id pembeli yang melakukan transaksi.
* Tipe data int digunakan untuk menyimpan nilai numerik id pembeli.
* Not null memastikan kolom ini harus diisi dengan id pembeli yang valid.

FOREIGN KEY(ID\_BARANG) REFERENCES BARANG(ID\_BARANG)

* Foreign key mendefinisikan hubungan antara kolom id\_barang di tabel penjualan dan kolom id\_barang di tabel barang.
* Ini memastikan bahwa hanya id barang yang valid yang dapat dimasukkan, yaitu id yang sudah ada di tabel barang.

FOREIGN KEY(ID\_PEMBELI) REFERENCES PEMBELI(ID\_PEMBELI)

* Foreign key mendefinisikan hubungan antara kolom id\_pembeli di tabel penjualan dan kolom id\_pembeli di tabel pembeli.
* Ini memastikan bahwa hanya id pembeli yang valid yang dapat dimasukkan, yaitu id yang sudah ada di tabel pembeli.

INSERT INTO PENJUALAN VALUES

(1,'DEPOK','KAMIS','CECE',2,3,4),

(2,'JAKARTA','RABU','JENNIFER',1,2,1),

(3,'CIBUBUR','JUMAT','CECE',4,3,3),

(4,'JAKARTA','KAMIS','AHMAD',2,1,2),

(5,'DEPOK','RABU','CECE',1,5,4);

Setiap baris dalam query ini memasukkan satu transaksi penjualan ke dalam tabel penjualan dengan nilai-nilai berikut:

ID\_PENJUALAN

* Merupakan id unik untuk setiap transaksi penjualan.
* Contoh: 1, 2, 3, 4, 5.

CABANG

* Nama cabang tempat transaksi dilakukan.
* Contoh: 'depok', 'jakarta', 'cibubur'.

HARI\_JUAL

* Hari transaksi dilakukan.
* Contoh: 'kamis', 'rabu', 'jumat'.

PENJUAL

* Nama penjual yang memproses transaksi.
* Contoh: 'cece', 'jennifer', 'ahmad'.

JUMLAH\_BELI

* Jumlah barang yang dibeli oleh pembeli dalam transaksi.
* Contoh: 2, 1, 4.

ID\_BARANG

* Id barang yang dijual dalam transaksi.
* Nilai ini merujuk pada id\_barang yang ada di tabel barang. Dalam query ini, id barang yang digunakan adalah 3, 2, 3, 1, dan 5.

ID\_PEMBELI

* Id pembeli yang melakukan transaksi.
* Nilai ini merujuk pada id\_pembeli yang ada di tabel pembeli. Dalam query ini, id pembeli yang digunakan adalah 4, 1, 3, 2, dan 4.

-- SCREENSHOT 3

SELECT \* FROM PENJUALAN;

Menampilkan semua data di tabel pembeli

-- SCREENSHOT 4

SELECT \* FROM PENJUALAN

JOIN PEMBELI ON PENJUALAN.ID\_PEMBELI = PEMBELI.ID\_PEMBELI

JOIN BARANG ON PENJUALAN.ID\_BARANG = BARANG.ID\_BARANG;

SELECT \* FROM PENJUALAN

* Perintah ini akan mengambil semua kolom dari tabel penjualan. Tanda \* berarti "ambil semua kolom" dari tabel yang disebutkan.

JOIN PEMBELI ON PENJUALAN.ID\_PEMBELI = PEMBELI.ID\_PEMBELI

* Ini adalah join antara tabel penjualan dan tabel pembeli.
* Penjualan.id\_pembeli = pembeli.id\_pembeli adalah kondisi yang digunakan untuk mencocokkan kolom id\_pembeli yang ada di kedua tabel tersebut.
* Dengan kata lain, query ini akan menggabungkan data transaksi dari penjualan dengan data pembeli yang sesuai dari tabel pembeli.

JOIN BARANG ON PENJUALAN.ID\_BARANG = BARANG.ID\_BARANG

* Ini adalah join antara tabel penjualan dan tabel barang.
* Penjualan.id\_barang = barang.id\_barang adalah kondisi yang digunakan untuk mencocokkan kolom id\_barang yang ada di kedua tabel tersebut.
* Dengan kata lain, query ini akan menggabungkan data transaksi dari penjualan dengan data barang yang dijual yang ada di tabel barang.

-- SCREENSHOT 5

SELECT

JUAL.CABANG,

JUAL.HARI\_JUAL,

BRG.NAMA\_BARANG,

BRG.HARGA,

JUAL.JUMLAH\_BELI,

(JUAL.JUMLAH\_BELI \* BRG.HARGA) AS TOTAL\_HARGA

FROM

PENJUALAN JUAL, BARANG BRG

WHERE JUAL.ID\_BARANG = BRG.ID\_BARANG;

SELECT

Kolom yang akan diambil dalam query ini:

* Jual.cabang: nama cabang tempat transaksi dilakukan dari tabel penjualan.
* Jual.hari\_jual: hari transaksi penjualan dilakukan dari tabel penjualan.
* Brg.nama\_barang: nama barang yang dijual dari tabel barang.
* Brg.harga: harga barang dari tabel barang.
* Jual.jumlah\_beli: jumlah barang yang dibeli dari tabel penjualan.
* Total\_harga: kolom baru yang dihitung dengan rumus (jual.jumlah\_beli \* brg.harga), yang akan menunjukkan total harga transaksi untuk barang yang dibeli. Alias total\_harga digunakan untuk memberikan nama pada kolom hasil perhitungan ini.

FROM PENJUALAN JUAL, BARANG BRG

* Query ini menggunakan format implicit join (join yang tidak eksplisit dengan kata kunci join). Tabel penjualan diberi alias jual, dan tabel barang diberi alias brg.
* Kedua tabel ini akan digabung berdasarkan kondisi yang diberikan di where.

WHERE JUAL.ID\_BARANG = BRG.ID\_BARANG

* Kondisi where ini menghubungkan tabel penjualan (jual) dan tabel barang (brg) berdasarkan kecocokan id barang (id\_barang).
* Artinya, transaksi yang tercatat di penjualan akan digabungkan dengan data barang yang sesuai di tabel barang, di mana id barang yang dijual (dari tabel penjualan) cocok dengan id barang yang ada di tabel barang.

-- SCREENSHOT 6

SELECT

BELI.NAMA AS NAMA,

JUAL.HARI\_JUAL,

JUAL.CABANG,

BRG.NAMA\_BARANG

FROM

PENJUALAN JUAL, BARANG BRG, PEMBELI BELI

WHERE JUAL.ID\_PEMBELI = BELI.ID\_PEMBELI

AND JUAL.ID\_BARANG = BRG.ID\_BARANG

SELECT

* Beli.nama as nama: mengambil nama pembeli dari tabel pembeli. Alias nama digunakan untuk memberi nama pada kolom yang akan ditampilkan.
* Jual.hari\_jual: mengambil hari transaksi dilakukan dari tabel penjualan.
* Jual.cabang: mengambil nama cabang tempat transaksi dilakukan dari tabel penjualan.
* Brg.nama\_barang: mengambil nama barang dari tabel barang.

FROM PENJUALAN JUAL, BARANG BRG, PEMBELI BELI

* Query ini menggabungkan tiga tabel: penjualan, barang, dan pembeli.
* Tabel penjualan diberi alias jual, tabel barang diberi alias brg, dan tabel pembeli diberi alias beli.

WHERE

* Jual.id\_pembeli = beli.id\_pembeli: kondisi ini menghubungkan tabel penjualan dan pembeli berdasarkan id\_pembeli. Artinya, kita ingin mengambil transaksi yang dilakukan oleh pembeli yang sesuai dengan id\_pembeli yang ada di tabel pembeli.
* Jual.id\_barang = brg.id\_barang: kondisi ini menghubungkan tabel penjualan dan barang berdasarkan id\_barang. Artinya, kita ingin mengambil transaksi yang berisi barang yang sesuai dengan id\_barang yang ada di tabel barang.