Modul Praktikum

Praktikum Basis Data

Pertemuan 3

PEMBUATAN DATABASE KOMPLEKS DAN RELASI TABLE

Program Studi	Tatap Muka	Kode MK
S1 - Teknik Informatika	01	Kxxxx

Abstraksi

Pertemuan ini membahas tentang database kompleks yang terdiri dari relasi tabel, constrain, foreign key serta join tabel

Kompetensi

Mahasiswa memahami hubungan / relasi antar setiap tabel yang dibuat.
Mahasiswa dapat mengolah data berdasarkan penggabungan dua atau lebih tabel yang terdapat didalam database.
Mahasiswa mampu memahami mengenai perintah-perintah Join table.

Petunjuk Pelaksanaan Praktikum

- Pahami tujuan dan dasar teori dengan baik dan benar
- Kerjakan latihan dan tugas praktikum dengan baik, benar, mandiri, serta jujur

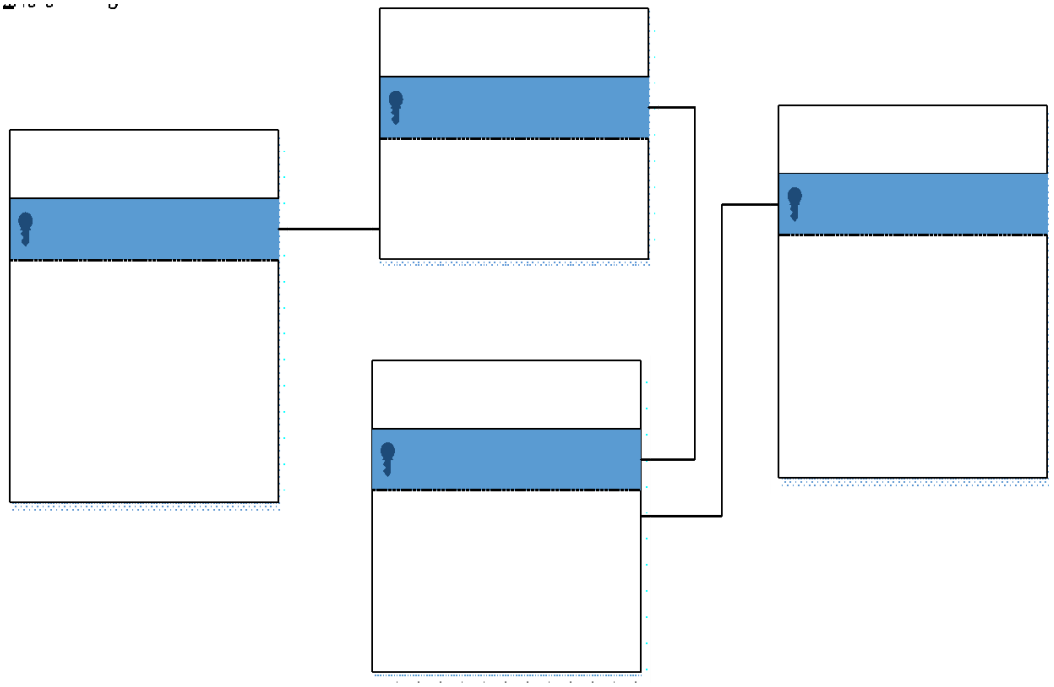
- Tanyakan kepada asisten praktikum / dosen berkaitan dengan hal-hal yang kurang jelas / mengalami kesulitan saat melaksanakan kegiatan praktikum

3.1 Pendahuluan

Pemodelan tabel komplek merupakan hubungan antar setiap tabel yang terdapat dalam sebuah database. Dalam sebuah database yang berelasional, data dapat ditampilkan dari dua tabel atau tiga tabel yang berbeda. Akan tetapi, apabila beberapa tabel tersebut merupakan yang berelasi, dilakukan teknik seleksi relasi khusus.

3.2 Praktikum

Dalam materi ini akan dijabarkan pembuatan tabel - tabel didalam database, dimana antar tabel saling terkait satu sama lainnya. Rancangan database pada contoh kasus pada transaksi penjualan dalam sebuah toko ditampilkan pada gambar 2.



Gambar 2. Desain Database

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 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 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 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 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)

```
)
```

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. Pengertian Foreign Key adalah 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:

Insert Data Customer

- INSERT INTO customer VALUES ('P001','Ahmad Sutopo','Jln Kepiting 45 Banyuwangi','254654','a.sutopo@email.com');
- INSERT INTO customer VALUES ('P002','Indah Lestari','Jln. Setaman 32 Banyuwangi','443426','indah54@email.com');
- INSERT INTO customer VALUES ('P003','Budi Suprapto','Jln. Suropati 56 Banyuwangi','663452','budi.s@email.com');
- INSERT INTO customer VALUES ('P004','Hendrik Maulana','Jln. Merdeka 13 Banyuwangi','351854','hendrikjabrik@email.com');
- INSERT INTO customer VALUES ('P005','Rahmat Arif','Jln. Jaya Giri 14 Banyuwangi','35627','arif99@email.com');

Insert Data Barang

- INSERT INTO barang VALUES ('B001','Buku Tulis','biji','2700','20');
- INSERT INTO barang VALUES ('B002','Pulpen','buah','3000','20');
- INSERT INTO barang VALUES ('B003','Penggaris','buah','3000','20');
- INSERT INTO barang VALUES ('B004','Pensil','buah','2000','10');

Insert Data Transaksi

- INSERT INTO transaksi VALUES ('1','P001','2017-10-10');
- INSERT INTO transaksi VALUES ('2','P002','2017-10-12');
- INSERT INTO transaksi VALUES ('3','P002','2017-10-12');
- INSERT INTO transaksi VALUES ('4','P004','2017-10-16');
- INSERT INTO transaksi VALUES ('5','P001','2017-10-17');

Insert Data Detail Transaksi

- INSERT INTO detail_transaksi VALUES ('1','B002','1','3000');
- INSERT INTO detail_transaksi VALUES ('1','B003','1','3000');
- INSERT INTO detail_transaksi VALUES ('2','B004','3','2000');
- INSERT INTO detail_transaksi VALUES ('3','B001','1','2700');
- INSERT INTO detail_transaksi VALUES ('3','B002','2','3000');

- INSERT INTO detail_transaksi VALUES ('4','B001','1','2700');
- INSERT INTO detail_transaksi VALUES ('4','B003','2','3000');
- INSERT INTO detail_transaksi VALUES ('4','B004','1','2000');
- INSERT INTO detail_transaksi VALUES ('5','B002','2','3000');

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.

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:

Menggabung dua tabel dengan kondisi Where

- SELECT tabel1.*, tabel2.* FROM tabel1, tabel2 WHERE tabel1.PK=tabel2.FK;
- SELECT customer.id_customer, customer.nama_customer, transaksi.id_transaksi, transaksi.tgl_pesan FROM customer, transaksi WHERE customer.id_customer = transaksi.id_customer;

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;
- SELECT customer.id_customer, customer.nama_customer, transaksi.id_transaksi, transaksi.tgl_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:

Menggabung tabel dengan left join

- SELECT tabel1.*, tabel2.* FROM tabel1 LEFT JOIN tabel2 ON tabel1.PK=tabel2.FK;
- SELECT customer.id_customer, customer.nama_customer, transaksi.id_transaksi, transaksi.tgl_pesan FROM customer LEFT JOIN transaksi ON customer.id_customer=transaksi.id_customer;

Menggabung tabel dengan right Join

- SELECT tabel1.*, tabel2.* FROM tabel1 RIGHT JOIN tabel2 ON tabel1.PK=tabel2.FK;
- SELECT customer.id_customer, customer.nama_customer, transaksi.id_transaksi, transaksi.tgl_pesan FROM customer RIGHT JOIN transaksi ON customer.id_customer=transaksi.id_customer;

3.3 Soal Latihan

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

nama_barang	jumlah
…	…
…	…

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 J**oin**). Hasil tabel seperti berikut:

Id_transaksi	Id_barang	Nama_barang	harga	Jumlah
…	…	…	…	…
…	…	…	…	…

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:

Nama_customer	jumlah
…	…
…	…
…	…

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. Dikumpulkan pada pertemuan selanjutnya dalam bentuk softcopy.