

## PERTEMUAN 4

### KONSEP DAN ARSITEKTUR BASIS DATA

#### A. TUJUAN PEMBELAJARAN

Setelah mempelajari materi ini, diharapkan mahasiswa mampu memahami konsep basis data, memahami berbagai macam model data dan arsitektur basis data.

#### B. URAIAN MATERI

##### 1. Model Data

Menurut A. Silberschatz, H F. Korth dan S. Sudarshan dalam bukunya yang berjudul "*Database System Concept: Sixth edition*", pada tahun 2011. Model data merupakan alat utama untuk menyediakan abstraksi data. Model data merupakan konsep yang dapat digunakan untuk menjelaskan struktur dari *database* yaitu tipe data, relasi dan *constraint*. Model data menyediakan cara untuk mendeskripsikan desain dari basis data pada tingkat *Physical, Logical* dan *View Levels*.

Menurut C. Coronel dan S. Morris, Model data adalah representasi yang relatif sederhana, biasanya grafis, dari struktur data dunia nyata yang lebih kompleks. Secara umum, model adalah abstraksi dari objek atau peristiwa dunia nyata yang lebih kompleks. Fungsi utama dari Model yaitu untuk membantu memahami kompleksitas dari lingkungan dunia nyata. Dasar dari sebuah model data adalah entitas, atribut, hubungan(relasi) dan batasan(*constraint*).

Entitas adalah orang, tempat, benda, atau peristiwa yang datanya akan dikumpulkan dan di simpan. Entitas mewakili jenis objek tertentu di dunia nyata, yang berarti entitas "dapat dibedakan" yang artinya setiap entitas itu unik dan berbeda.

Atribut adalah karakteristik dari suatu entitas. Sebagai contoh, entitas pelanggan akan dijelaskan dengan atribut seperti Nama, Nomor telepon, dan Alamat.

Hubungan menggambarkan asosiasi antar entitas. Misalnya, ada hubungan antara pelanggan dan penjual yang dapat dijelaskan sebagai berikut: Seorang Penjual dapat melayani banyak Pelanggan, dan setiap Pelanggan dapat dilayani oleh satu Penjual saja. Model data menggunakan tiga tipe dari hubungan atau relasi, yaitu : *One-to-many*, *Many-to-Many* dan *One-to-one*.

Contoh dari *One-to-many* yaitu seperti lukisan hanya bisa dibuat oleh satu pelukis, tetapi pelukis bisa membuat banyak lukisan. Sedangkan *Many-to-Many* contohnya seperti kelas memiliki banyak siswa, dan siswa bisa memiliki banyak kelas. Sedangkan *One-to-one* contohnya seperti, satu computer memiliki satu prosesor, dan satu prosesor dimiliki satu computer.

Batasan (*constraint*) adalah Batasan yang ditempatkan pada data. Batasan sangatlah penting dikarenakan membantu memastikan integritas dari data. Didefinisikan sebagai property atau keadaan tertentu yang harus dipatuhi oleh data dalam *database*. Batasan biasanya diekspresikan dalam bentuk aturan. Contohnya setiap kelas harus memiliki satu guru. Jenis umum dari *constraint* adalah sebagai berikut :

- a. *Not null* – setiap nilai di dalam kolom tidak boleh NULL.
- b. *Unique* – nilai pada kolom tertentu harus unik untuk setiap baris di dalam tabel.
- c. *Primary key* – nilai dalam kolom tertentu harus unik untuk setiap baris dalam tabel dan bukan NULL, biasanya setiap tabel dalam *database* harus memiliki *primary key*, ini digunakan untuk mengidentifikasi dari masing-masing individu kolom.
- d. *Foreign key* – nilai di kolom tertentu harus mereferensikan record yang ada pada tabel lain (melalui *Primary key* atau Batasan unik lainnya).
- e. *Check* – ekspresi ditentukan, yang harus di evaluasi ke true agar kendala terpenuhi.

Berikut adalah salah satu contoh dari *primary key* dan *foreign key* :

#	Name	Type
<input type="checkbox"/> 1	Kd_admin	int(11)
<input type="checkbox"/> 2	Nm_admin	varchar(30)
<input type="checkbox"/> 3	User_name	varchar(30)
<input type="checkbox"/> 4	Password	varchar(30)
<input type="checkbox"/> 5	Hak_akses	varchar(20)

**Gambar 0.1 Primary key**

#	Name	Type
<input type="checkbox"/> 1	Kd_karyawan	int(11)
<input type="checkbox"/> 2	Kd_divisi	int(11)
<input type="checkbox"/> 3	Nm_karyawan	varchar(30)
<input type="checkbox"/> 4	Almt_karyawan	text
<input type="checkbox"/> 5	Tlp_karyawan	varchar(15)
<input type="checkbox"/> 6	Jenis_kel	varchar(10)
<input type="checkbox"/> 7	Gaji_pokok	double
<input type="checkbox"/> 8	Kd_admin	int(11)

**Gambar 0.2 Foreign key**

Gambar kunci berwarna kuning tersebut adalah *primary key*, berarti *field* kd\_admin disana menjadi *primary key*. sedangkan gambar dengan warna lebih gelap/abu-abu, itu adalah yang dinamakan *foreign key*.

Ada sejumlah model data yang berbeda, yaitu :

a. Model Data Relasional

Dimana data serta hubungan antar data direpresentasikan atau disajikan oleh sejumlah table dan masing-masing table terdiri dari beberapa *field* yang namanya unik. Model Relasional adalah adalah ilustrasi dari Model Data Berbasis Catatan. Model ini memberi tahu cara terbaik untuk benar-benar mengawasi informasi dalam memori sekunder, yang juga akan memengaruhi cara kita mengumpulkan informasi dan menyusun informasi umum yang terkait dengan kerangka kerja yang kita buat. model informasi yang paling

banyak digunakan, dan sebagian besar kerangka kerja kumpulan data saat ini didasarkan pada model relasional. Contoh tabel dan keterhubungannya :

MHS

**Tabel 0.1** Tabel Data Mahasiswa

NIM	NAMA	ALAMAT
231412	Imam	Jakarta
223144	Dani	Bogor
223545	Jaya	Bogor
232075	Aris	Tangerang

MKUL

**Tabel 0.2** Tabel Data Matakuliah

KDMK	MTKULIAH	SKS
123	Basis Data	2
321	Pemrograman	3
134	Pancasila	2

NILAI

**Tabel 0.3** Tabel Data Nilai


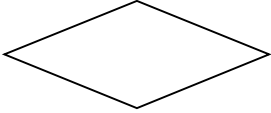
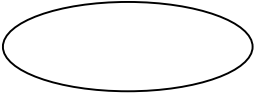

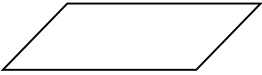
NIM	KDMK	UTS	UAS
231412	123	70	75
223144	321	60	75
223545	134	90	70

232075	123	65	75
--------	-----	----	----

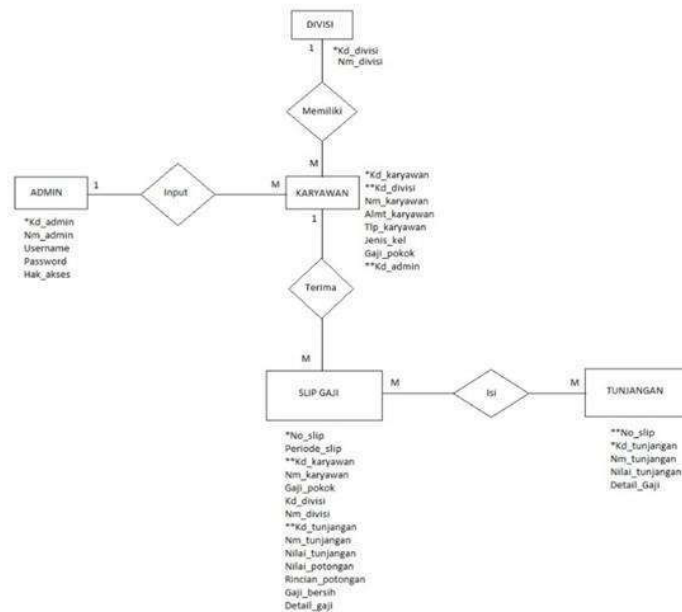
b. Model *Data entity relationship*

Model *Data entity relationship* (ER) menggunakan kumpulan dari objek dasar yang disebut entitas dan garis yang menghubungkan diantara objek tersebut. Entitas adalah suatu objek yg dapat dibedakan dari objek lain. Model data ini juga akan membantu pada saat melakukan analisis dan perancangan *database*, karena model data ini akan menunjukkan bermacam-macam data yang dibutuhkan dan hubungan antar data. Berikut adalah simbol-simbol dari Model *Data entity relationship* (ER).

**Tabel 0.4** Simbol Model *Data entity relationship* (ER)

Notasi	Keterangan
	Entitas, yaitu kumpulan dari objek yang dapat diidentifikasi secara unik
	Relasi, yaitu hubungan yang terjadi antara satu atau lebih entitas. Jenis hubungan antara lain : satu ke satu, satu ke banyak, banyak ke banyak
	Atribut, yaitu karakteristik dari <i>entity</i> atau relasi yang merupakan penjelasan detail tentang entitas
	Garis, hubungan antara <i>entity</i> dengan atributnya dan himpunan entitas dengan himpunan relasi
	<i>Input/output</i> data, yaitu proses <i>input/output</i> data, parameter, informasi

berikut adalah contoh dari model *data entity relationship* :



**Gambar 0.3** Model *Data entity relationship* (ER)

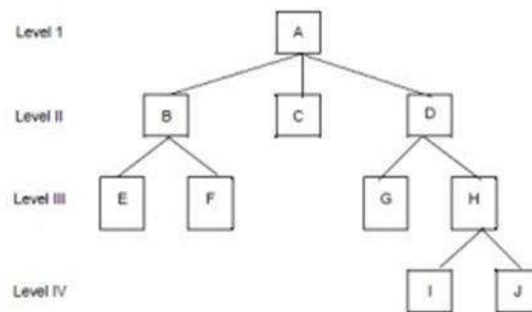
c. Model Data Berbasis Objek

Masalah dunia nyata yang semakin kompleks menunjukkan kebutuhan akan model data yang lebih mewakili dunia nyata. Dalam model data berbasis objek, baik data maupun hubungannya terdapat dalam satu struktur yang dikenal sebagai objek. Model data berbasis objek mencerminkan cara yang sangat berbeda untuk mendefinisikan dan menggunakan entitas. Seperti entitas dalam model data relasional, sebuah objek dijelaskan oleh konten faktualnya. Namun tidak seperti entitas, objek mencakup informasi tentang hubungan antara fakta di dalam objek, serta informasi tentang hubungan dengan objek lain. Oleh karena itu, fakta di dalam objek diberi makna yang lebih besar.

d. Model Data Hirarki

Model hierarki dikembangkan pada 1960-an untuk mengelola sejumlah besar data untuk proyek manufaktur yang kompleks, seperti roket Apollo yang mendarat di bulan pada 1969. Struktur logika dasar dari model ini diwakili oleh bentuk pohon terbalik. Struktur model hirarki berisi *level* atau tingkatan atau segmen. Dalam hierarki, lapisan yang lebih tinggi dianggap sebagai induk dari segmen atau tingkatan tepat di bawahnya. Model hirarki

menggambarkan sekumpulan hubungan *one-to-many* antara tingkatan induk atau tingkatan yang lebih tinggi dengan tingkatan dibawahnya.



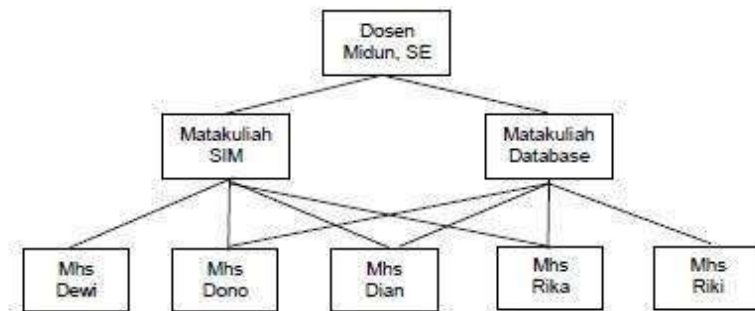
**Gambar 0.4** Model Data Hirarki

e. Model Data Jaringan

Model jaringan dibuat untuk merepresentasikan hubungan data yang kompleks secara lebih efektif daripada model hierarki, untuk meningkatkan kinerja *database*, dan untuk menerapkan standar *database*. Namun, tidak seperti model hierarki, model jaringan memungkinkan satu data memiliki lebih dari satu induk. Sementara model data jaringan sudah jarang digunakan saat ini, namun definisi dari konsep standar basis data yang muncul dari model data jaringan masih digunakan oleh model data modern, seperti :

- 1) *Schema* adalah organisasi konseptual dari seluruh *database* seperti yang dilihat oleh administrator *database*.
- 2) *SubSchema* mendefinisikan bagian dari *database* yang dilihat oleh program aplikasi yang benar-benar menghasilkan informasi yang diinginkan dari data di dalam *database*.
- 3) Bahasa manipulasi data (DML) mendefinisikan lingkungan di mana data dapat dikelola dan digunakan untuk bekerja dengan data dalam *database*.
- 4) Bahasa definisi data skema (DDL) memungkinkan administrator *database* untuk menentukan komponen skema.

Berikut adalah contoh dari model data jaringan :



**Gambar 0.5** Model Data Jaringan

## 2. Schema

*Schema* adalah grup logis dari objek *database* seperti tabel dan indeks yang terkait satu sama lain. Biasanya, skema tersebut dimiliki oleh satu pengguna atau aplikasi. Satu *database* dapat menampung beberapa skema yang dimiliki oleh pengguna atau aplikasi yang berbeda. *Schema* berguna karena mereka mengelompokkan tabel menurut pemilik (atau fungsi) dan menerapkan tingkat keamanan pertama dengan mengizinkan setiap pengguna untuk melihat hanya tabel milik pengguna tersebut.

*Schema* terdiri dari 3 jenis, yaitu *physical schema*, *logical schema* dan *view schema*. Desain basis data pada *physical level* disebut *physical schema*, bagaimana data yang disimpan dalam blok penyimpanan dijelaskan pada *level* ini. Desain basis data pada tingkat *logical* disebut *logical schema*, pemrogram dan admin basis data bekerja pada tingkat ini, pada tingkat ini data dapat dijelaskan sebagai salah satu jenis catatan data tertentu yang disimpan dalam struktur data, namun detail *internal* seperti implementasi struktur data tersembunyi pada *level* ini (tersedia di *physical level*). Desain basis data pada *level view* disebut *view schema*. Pada *level* ini biasanya menjelaskan interaksi pengguna akhir dengan sistem basis data.

## 3. Instance

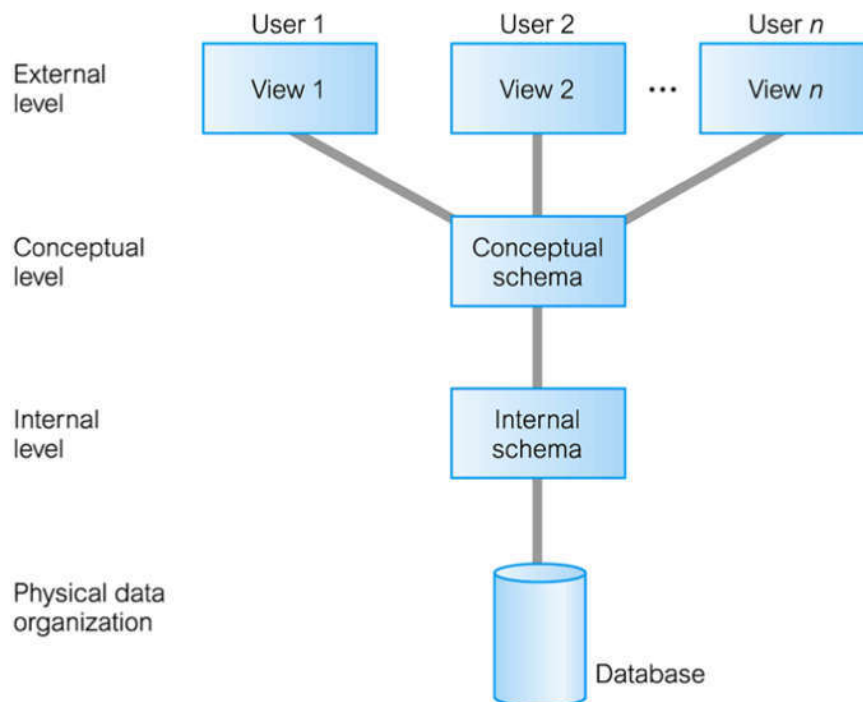
Data yang disimpan dalam *database* pada saat tertentu disebut *instance database*. *Instance database* adalah sekumpulan struktur memori dan proses latar belakang yang mengakses sekumpulan file *database*. Prosesnya bisa



dibagikan oleh semua pengguna. Struktur memori yang digunakan untuk menyimpan sebagian besar data yang diminta dari *database*. *Instance* ini membantu meningkatkan kinerja *database* dengan mengurangi jumlah I/O yang dilakukan terhadap file data. Parameter *instance* ada dua tipe yaitu *dynamic* dan *static*. Parameter *dynamic* bisa diubah ketika *instance* sedang jalan sedangkan parameter *static* tidak bisa, artinya perubahan parameter *static* harus dilakukan di *initfile* dan *instance* harus di-restart.

#### 4. Arsitektur Basis Data

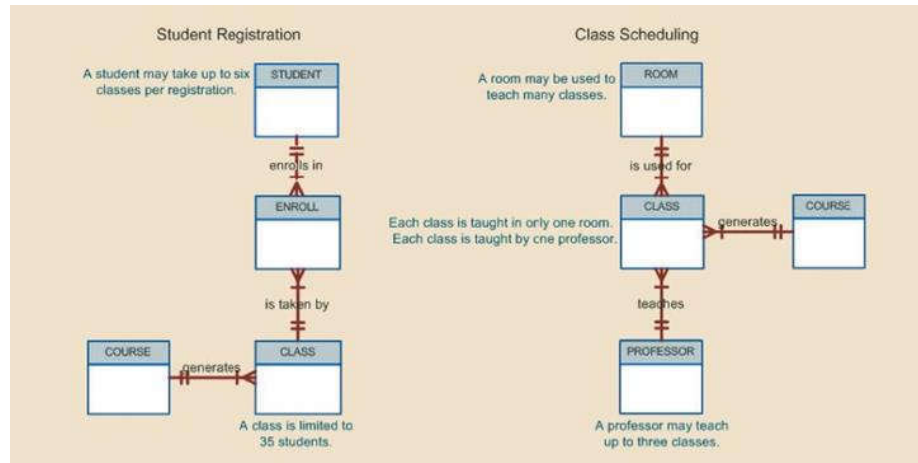
Arsitektur Basis data merupakan suatu kumpulan data yang tersimpan secara sistematis, dimana user dapat melihat data dan bagaimana cara user melihat data tersebut. Ada tiga *level* arsitektur menurut ANSI/SPARC, yaitu *External level*, *Conceptual level* dan *Internal level*.



**Gambar 0.6** Arsitektur Basis Data

*External level* adalah *level* tertinggi dalam arsitektur tiga *level* dan paling dekat dengan pengguna. *Level* ini juga dikenal sebagai *View level*. *External level* hanya memperlihatkan konten *database* yang relevan kepada pengguna dalam bentuk tampilan dan menyembunyikan data lainnya. Jadi pengguna yang berbeda dapat melihat *database* sebagai tampilan yang berbeda sesuai

kebutuhan masing-masing. Karena data sedang dimodelkan, diagram ER akan digunakan untuk merepresentasikan *external view*. Representasi tertentu dari tampilan eksternal dikenal sebagai *external schema*. Untuk mengilustrasikan *external level*, kita menggunakan data dari ruang lingkup kampus atau perkuliahan.



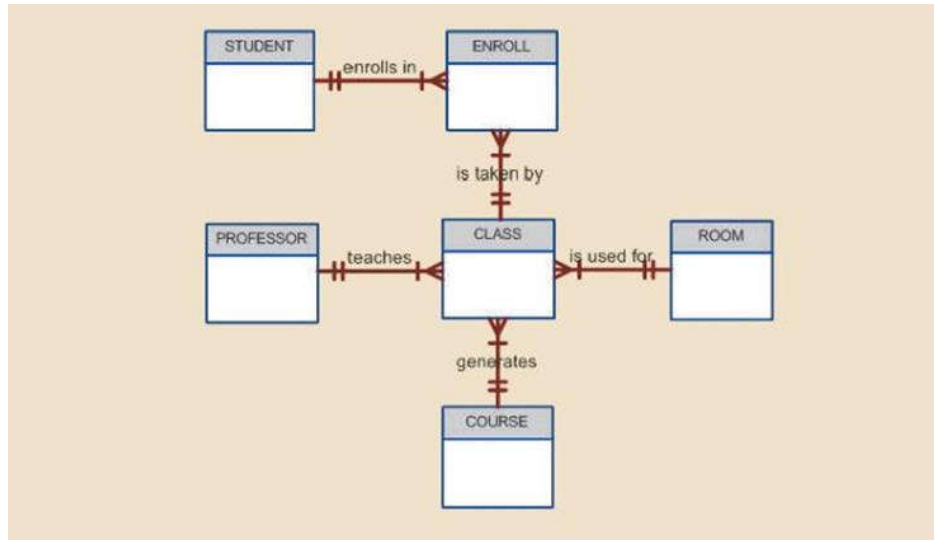
**Gambar 0.7 External Level**

- Seorang professor bisa mengajar banyak kelas dan setiap kelas hanya bisa diajar oleh satu orang professor, disini terjadi hubungan *one-to-many* diantara professor dan kelas.
- Kelas dapat terdaftar banyak siswa, dan siswa dapat mendaftar banyak kelas. Disini terjadi hubungan *many-to-many* diantara kelas dan siswa.
- Setiap kursus dapat menghasilkan banyak kelas, tetapi setiap kelas hanya merujuk pada satu kursus.
- Terakhir, satu kelas membutuhkan satu ruangan tetapi satu ruangan mungkin dijadwalkan untuk banyak kelas.

*Internal level* adalah *level* terendah dari ke tiga *level* arsitektur. *Internal level* menjelaskan bagaimana data tersebut tersimpan pada basis data. Pada *level* ini, data disimpan di harddisk dalam bentuk bit atau pada *level* yang sedikit tinggi dapat dikatakan data tersebut tersimpan dalam bentuk file dan folder.

*Conceptual level* menghubungkan antara kedua *level* yaitu *level internal* dan *level external*. *Conceptual level* merepresentasikan seluruh muatan dari basis data. *Conceptual level* tidak peduli dengan bagaimana data pada *database* disimpan, *level* ini menjelaskan bagaimana *database* ditampilkan

secara konsep dan relasi-relasinya diantara beberapa tabel. Berikut adalah contoh dari *conceptual level*, kita menggunakan ruang lingkup dari kampus atau perkuliahan.



**Gambar 0.8** *Conceptual level*

### C. SOAL LATIHAN/TUGAS

Untuk mengetahui apakah anda sudah memahami konsep dan arsitektur *system* basis data, kerjakan Latihan berikut :

1. Jelaskan pengertian dari model data?
2. Ada berapakah jenis model data, dan jelaskan!
3. Buatlah satu Model *data entity relationship* (ER), dengan kasus penjualan mobil!
4. Ada berapakah jenis arsitektur basis data, dan jelaskan!
5. Jelaskan kegunaan dari indepedensi data!

## D. REFERENSI

Connolly, T., & Begg, C. (2005). *Database System: A Practical Approach to Design, Implementation, and Management, Fourth Edition*. Harlow: Pearson Education Limited.

Coronel, C., & Morris, S. (2017). *Database System: Design, Implementation, & Management, 13th Edition*. Boston: Cengage Learning, Inc.

Silberschatz, A., Korth, H. F., & Sudarshan, S. (2011). *Database System Concept ; Sixth Edition*. New York: McGraw-Hill.

## GLOSARIUM

**Constraint** adalah Batasan yang ditempatkan pada data. Batasan sangatlah penting dikarenakan membantu memastikan integritas dari data.

**One to one** adalah baris data pada tabel pertama dihubungkan hanya ke satu baris data pada tabel ke dua. Hubungan antara *file* pertama dan *file* kedua adalah satu berbanding satu.

**One to many** adalah setiap baris data dari tabel pertama dapat dihubungkan ke satu baris atau lebih data pada tabel ke dua. Hubungan antara *file* pertama dan *file* kedua adalah satu berbanding banyak atau banyak berbanding satu.

**Many to many** adalah Satu baris atau lebih data pada tabel pertama bisa dihubungkan ke satu atau lebih baris data pada tabel ke dua. Artinya ada banyak baris di tabel satu dan tabel dua yang saling berhubungan satu sama lain. Hubungan tabel pertama dan tabel kedua adalah banyak berbanding banyak.