MATA KULIAH – ANALISIS DAN PERANCANGAN SISTEM INFORMASI

TOPIK 6: Data Design

Disusun oleh:

Wahyu Indra Satria, S.Kom., M.Kom.

Learning Objectives

(Tujuan Pembelajaran)

- Memahami dasar konsep dari *data design* seperti struktur data, *relational database* serta DBMS
- Memahami komponen utama *database management system* (DBMS)
- Merancang *entity relationship diagram* (ERD)

Dasar Konsep *Data Design*

➢ Struktur Data:

- Struktur data adalah kerangka kerja untuk mengatur, menyimpan, dan mengelola data.
- Struktur data terdiri dari *file* atau tabel yang berinteraksi dengan berbagai cara.
- Setiap *file* atau tabel berisi data tentang orang, tempat, benda, atau peristiwa.
- Misalnya satu file atau tabel mungkin berisi data tentang pelanggan, *dan* file atau tabel lain mungkin menyimpan data tentang produk, pesanan, pemasok, atau karyawan.
- Banyak sistem lama yang menggunakan ***file processing***, karena bekerja dengan baik dengan *hardware mainframe* dan *input batch*.
- Beberapa perusahaan masih menggunakan metode ini untuk menangani struktur data volume besar secara teratur karena dapat menghemat biaya dalam situasi tertentu.
- Contohnya perusahaan kartu kredit yang memposting ribuan transaksi harian dari *file* TRANSACTIONS ke saldo akun yang disimpan dalam *file* CUSTOMERS, seperti yang ditunjukkan pada **Figure 9–1 (pada *slide* 4)**.
- Untuk proses yang relatif sederhana tersebut, ***file processing*** mungkin bisa menjadi pilihan.

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

Dasar Konsep *Data Design* (lanjutan…)

➢ Struktur Data: (lanjutan…)

CUSTOMERS

Customer Number	Due Date	Balance Due
S3914	11-31-2019	86.52

TRANSACTIONS

Transaction Number	Customer Number	Date	Code	Amount
109823497	S3914	11-03-2019	Charge	110.50
210048576	S3914	11-04-2019	Return	23.98
994184424	X9810	11-04-2019	Charge	62.67

FIGURE 9-1 A credit card company that posts thousands of daily transactions might consider a file processing option.

Dasar Konsep *Data Design* (lanjutan…)

➢ *Modern Relational Database*:

- Seiring berjalannya waktu, *modern relational database* menjadi model standar untuk para pengembang sistem.

- Contoh bengkel mobil berikut akan membandingkan kedua konsep tersebut.

➢ Contoh *Data Design*: **Bengkel Mario vs Bengkel Danica**

- Bayangkan dua bengkel yang sangat mirip tetapi menggunakan dua desain sistem informasi yang berbeda.

- Sebut saja **Bengkel Mario** dan **Bengkel Danica**.

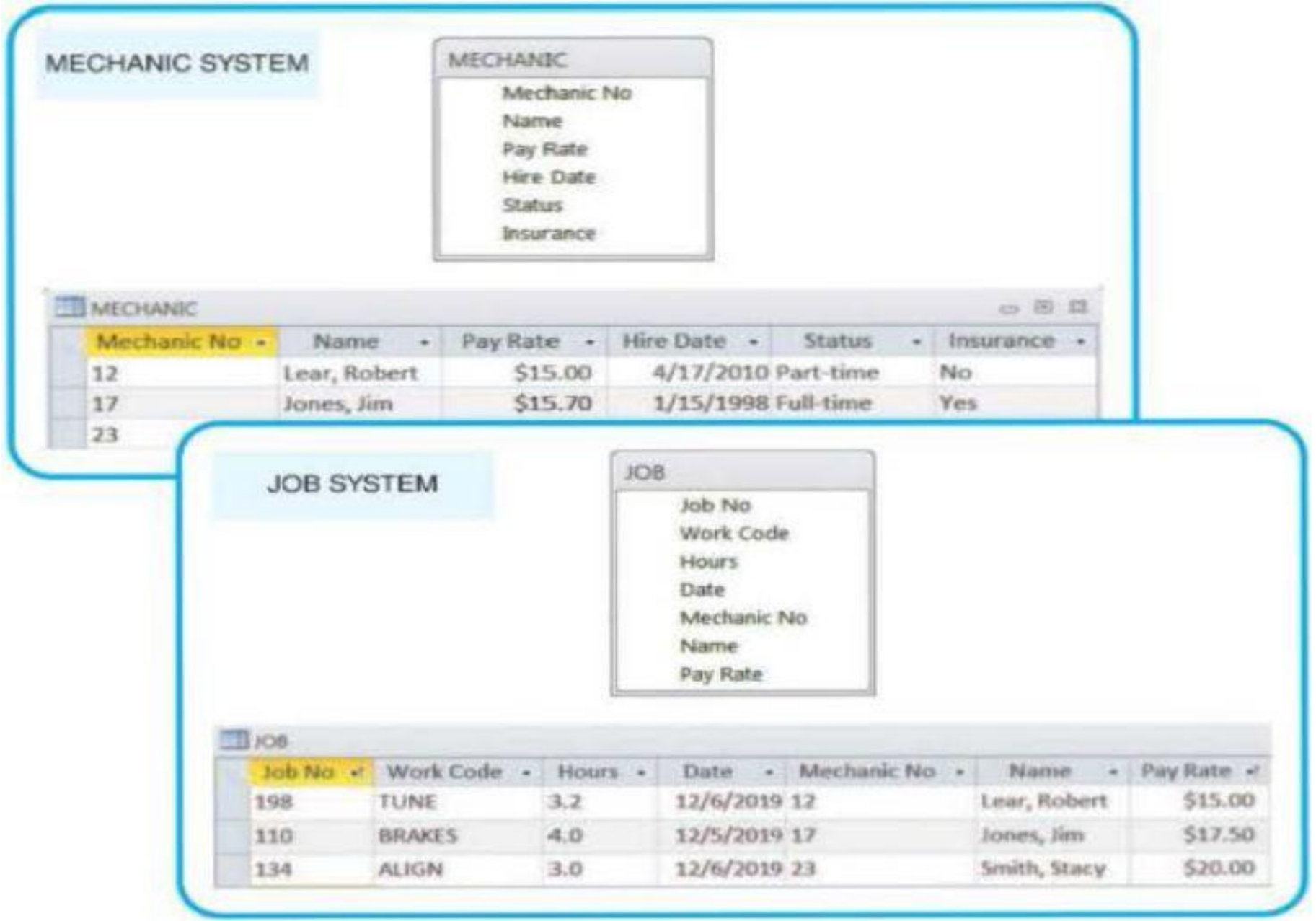
- **Bengkel Mario** menggunakan **dua *file–oriented system***, sedangkan **Bengkel Danica** menggunakan ***database management system*** **(DBMS)**.

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

Dasar Konsep *Data Design* (lanjutan...)

➢ Contoh *Data Design*: Bengkel Mario (*File–Oriented System*)

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.



Dasar Konsep *Data Design* (lanjutan…)

➢ Contoh *Data Design*: Bengkel Mario (lanjutan…)

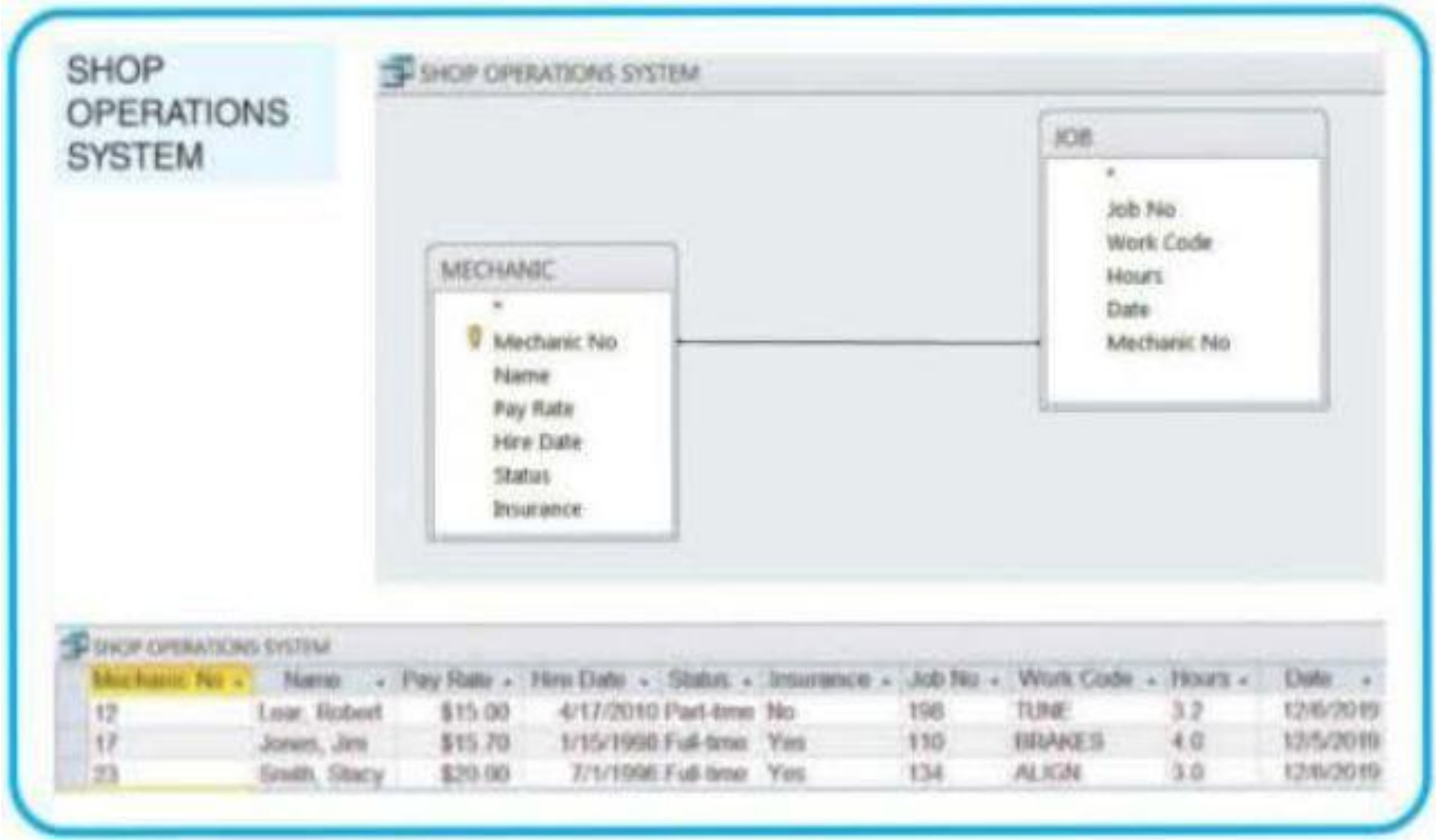
- Bengkel Mario mengandalkan **dua *file–oriented systems***, terkadang disebut ***file processing systems***, untuk mengelola bisnisnya.
- Kedua sistem menyimpan data dalam *file* terpisah yang tidak terhubung atau ditautkan satu dengan lainnya dimana:
- MECHANIC SYSTEMS menggunakan MECHANIC *file* untuk menyimpan data tentang pegawai Bengkel Mario.
- JOB SYSTEM menggunakan JOB file untuk menyimpan data tentang pelayanan/pekerjaan yang dilaksanakan pada Bengkel Mario.
- Sayangnya, menggunakan dua sistem yang terpisah berarti bahwa beberapa data disimpan di dua tempat yang berbeda, dan data tersebut mungkin konsisten atau tidak.
- Misalnya, tiga item data (**Mechanic No, Name, and Pay Rate**) disimpan di kedua file.
- Redundansi ini merupakan kelemahan utama dalam ***file processing systems*** karena mengancam kualitas dan integritas data.

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

Dasar Konsep *Data Design* (lanjutan...)

➢ Contoh *Data Design*: Bengkel Danica (DBMS)

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.



Dasar Konsep *Data Design* (lanjutan…)

➢ Contoh *Data Design*: Bengkel Danica (lanjutan…)

- Bengkel Danica menggunakan *database management system* (DBMS) dengan dua tabel terpisah yang kemudian digabungkan, sehingga keduanya seperti sebuah tabel besar.

- Pada SHOP OPERATION SYSTEM, kedua tabel terhubung oleh ***Mechanic No. Field***, yang kemudian disebut sebagai ***common field*** karena menghubungkan kedua tabel terpisah.

- Perhatikan bahwa (kecuali untuk *common field*), tidak ada item data lain yang diduplikasi.

- Desain DBMS disebut juga sebagai *relational databse* atau *relational model*, yang diperkenalkan pada 1970an dan berlanjut sebagai pendekatan dominan untuk mengatur, menyimpan, serta mengelola data bisnis.

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

Komponen Utama DBMS

Overview Mengenai DBMS:

- ***Database management system* (DBMS)** adalah kumpulan *tools*, *features*, dan *interfaces* (antarmuka), yang memungkinkan pengguna untuk menambahkan, memperbarui, mengelola, mengakses, serta menganalisis data.
- Dari sudut pandang pengguna, **keuntungan utama DBMS** adalah menawarkan akses data yang tepat waktu, interaktif, dan fleksibel, dimana secara spesifik dijabarkan meliputi:
  - ***Scalability*** → Sistem dapat diperluas, dimodifikasi, atau dirampingkan dengan mudah untuk memenuhi kebutuhan perusahaan bisnis yang berubah dengan cepat.
  - ***Economic of Scale*** → Desain *database* memungkinkan pemanfaatan perangkat keras yang lebih baik.
  - ***Enterprise–Wide Application*** → Sebuah DBMS biasanya dikelola oleh seseorang yang disebut *database administrator* (DBA), dimana DBA menilai keseluruhan kebutuhan sistem beserta memelihara *database*, untuk kepentingan seluruh organisasi daripada satu departemen atau pengguna saja.
  - ***Stronger Standards*** → Administrasi *database* yang efektif membantu memastikan bahwa standar untuk nama data, format, dan dokumentasi pada sistem, diikuti secara seragam di seluruh organisasi.
  - ***Better Security*** → DBA dapat menentukan prosedur otorisasi untuk memastikan bahwa hanya pengguna sah yang dapat mengakses *database*, dan dapat mengizinkan pengguna berbeda untuk memiliki tingkat akses berbeda pula.
  - ***Data Independence*** → Sistem yang berinteraksi dengan DBMS relatif independen, dari bagaimana data fisik di–*maintaned* (dijaga).

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

Komponen Utama DBMS (lanjutan…)

➢ *Overview* Mengenai DBMS: (lanjutan…)

FIGURE 9-5 In this example, a sales database can support four separate business systems.

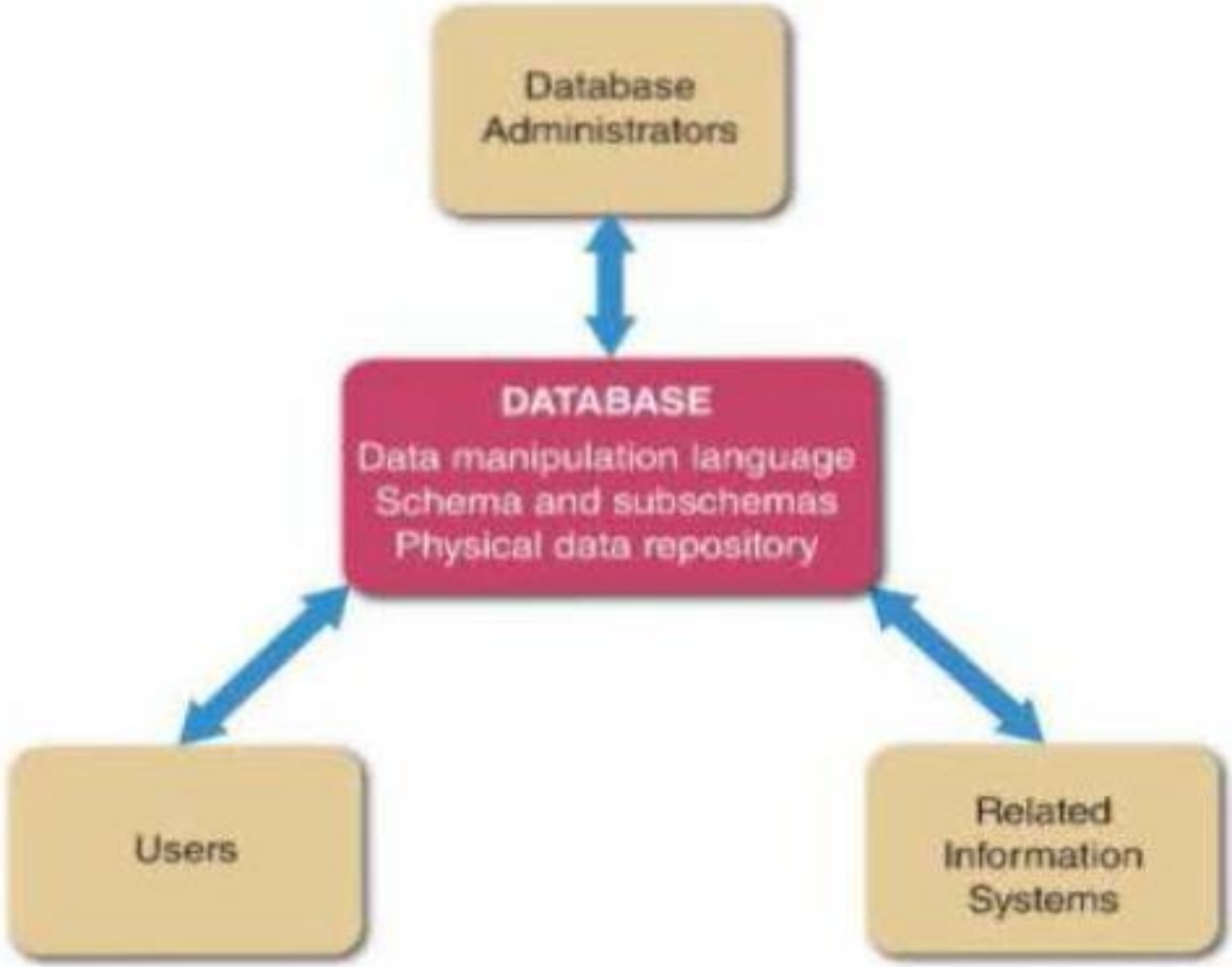


FIGURE 9-6 In addition to interfaces for users, database administrators, and related information systems, a DBMS also has a data manipulation language, a schema and subschemas, and a physical data repository.

Komponen Utama DBMS (lanjutan…)

1. *Interfaces* untuk *Users*, *Database Admin*, dan Sistem Terkait

- Ketika pengguna, DBA, dan sistem informasi terkait meminta data serta layanan, maka DBMS memproses permintaan tersebut, memanipulasi data, lalu memberikan respons.
- ***Data manipulation language* (DML)** mengendalikan operasi *database*, termasuk menyimpan, mengambil, memperbarui, dan menghapus data.
- Kebanyakan DBMS komersial, seperti *Oracle* dan DB2 IBM, menggunakan DML.

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

- Beberapa produk *database* seperti *Microsoft Access* juga menyediakan lingkungan grafis yang mudah digunakan, yang memungkinkan pengguna mengendalikan operasi bersama dengan perintah berbasis menu (*menu–driven commands*).
  - ***Interfaces* untuk *Users*** → Pengguna biasanya bekerja dengan *query* yang telah ditentukan sebelumnya dan perintah *switchboard*, tetapi juga menggunakan bahasa *query* untuk mengakses data yang disimpan (Bisa dengan ***Query By Example*–QBE** atau ***Structured Query Language*–SQL**).
  - ***Interfaces* untuk *Database Admin*** → Sebagian besar DBMS menyediakan program utilitas untuk membantu DBA dalam membuat dan memperbarui struktur data, mengumpulkan dan melaporkan pola penggunaan basis data, serta mendeteksi dan melaporkan penyimpangan basis data.
  - ***Interfaces* untuk Sistem Informasi Terkait** → DBMS dapat mendukung beberapa sistem informasi terkait yang menyediakan *input* ke DBMS, dan yang memerlukan data spesifik dari DBMS. Tidak seperti *user interface*, tidak ada campur tangan manusia yang diperlukan untuk komunikasi dua arah antara DBMS dan sistem informasi terkait.

Komponen Utama DBMS (lanjutan…)

2. *Schema* (Skema)

- ***Schema (skema)*** adalah definisi lengkap dari *database*, termasuk deskripsi semua bidang (*fields*), tabel (*tables*) , dan hubungan (*relationships*).
- Satu atau lebih subskema juga dapat didefinisikan.
- Subskema adalah tampilan *database* yang digunakan oleh satu/lebih sistem atau pengguna.
- Sebuah subskema mendefinisikan hanya memilih bagian dari *database*, yang sistem tertentu atau kebutuhan pengguna, atau yang diperbolehkan untuk mengaksesnya.
- Contohnya untuk melindungi privasi individu, sistem manajemen proyek tidak boleh diizinkan untuk mengambil tarif gaji karyawan.
- Dalam hal ini, subskema sistem manajemen proyek (*project management system*) tidak akan menyertakan bidang tarif pembayaran.
- Perancang basis data (*database designer*) juga menggunakan subskema untuk membatasi tingkat akses yang diizinkan.
- Contohnya pengguna, sistem, atau lokasi tertentu mungkin diizinkan untuk membuat, mengambil, memperbarui, atau menghapus data, bergantung pada kebutuhan mereka dan kebijakan keamanan perusahaan.

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

Komponen Utama DBMS (lanjutan…)

3. *Physical Data Repository* (Penyimpanan Data Fisik)

- Pada tahap proses pengembangan sistem, ada tahapan dimana *data dictionary* diubah menjadi *physical data repository*, yang juga berisi skema dan subskema.
- Repositori fisik mungkin terpusat, atau mungkin didistribusikan di beberapa lokasi.
- Selain itu, data bersama mungkin dikelola oleh satu DBMS atau beberapa sistem.
- Untuk mengatasi potensi konektivitas *database* dan masalah akses, perusahaan menggunakan *software* sesuai dengan ODBC, yang memungkinkan terjadinya komunikasi di antara berbagai sistem beserta DBMS.
- ***Open database connectivity*** **(ODBC)** atau konektivitas *database* terbuka adalah protokol standar industri yang memungkinkan *software* dari vendor–vendor berbeda untuk melakukan interaksi dan bertukar data.
- Standar umum lain yang biasa digunakan yaitu ***java database connectivity*** **(JDBC)** atau konektivitas *database* java.
- JDBC memungkinkan aplikasi java untuk bertukar data dengan *database* apa pun yang menggunakan statement SQL dan sesuai dengan JDBC.

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

Komponen Utama DBMS (lanjutan…)

3. *Physical Data Repository* (Penyimpanan Data Fisik) (lanjutan…)

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

FIGURE 9-7 Using QBE, a user can display all 2019 Ford Fusions that have a power moonroof and are either Lime Squeeze or Blue Candy color.

Komponen Utama DBMS (lanjutan…)

➢ *Web–based Design* (Internet sebagai *front endlinterface* dari DBMS):

CHARACTERISTIC	EXPLANATION
Global access	The Internet enables worldwide access, using existing infrastructure and standard telecommunications protocols.
Ease of use	Web browsers provide a familiar interface that is user-friendly and easily learned.
Multiple platforms	Web-based design is not dependent on a specific combination of hardware or software. All that is required is a browser and an Internet connection.
Cost effectiveness	Initial investment is relatively low because the Internet serves as the communication network. Users require only a browser, and web-based systems do not require powerful workstations. Flexibility is high because numerous outsourcing options exist for development, hosting, maintenance, and system support.
Security issues	Security is a universal issue, but Internet connectivity raises special concerns. These can be addressed with a combination of good design, software that can protect the system and detect intrusion, stringent rules for passwords and user identification, and vigilant users and managers.
Adaptability issues	The Internet offers many advantages in terms of access, connectivity, and flexibility. Migrating a traditional database design to the web, however, can require design modification, additional software, and some added expense.

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

Merancang *Entity Relationship Diagram* (ERD)

➢ Istilah–Istilah dalam ERD *Data Design*:

- Istilah desain data termasuk ***entity*, *table*, *file*, *field*, record/tuple,** dan ***key field***.
  - ***Entity*** → *Entity* adalah orang, tempat, benda, atau peristiwa yang datanya dikumpulkan dan dikelola. Misalnya sistem penjualan *online* dapat mencakup entitas bernama CUSTOMER, ORDER, PRODUCT, dan SUPPLIER.
  - ***Table/FIle*** → Data diatur ke dalam *table*/*file*. *Table*/*file* berisi sekumpulan *record* terkait, yang menyimpan data tentang entitas tertentu. Setiap kolom mewakili *field* atau karakteristik *entity*, dan setiap baris mewakili *record*, yang merupakan contoh individu, atau kejadian dari *entity*.
  - ***Field*** → *Field* disebut juga *attributes*, adalah karakteristik atau fakta tunggal tentang suatu *entity*. Misalnya entitas CUSTOMER mungkin menyertakan ID *customer*, Nama Depan, Nama Belakang, Alamat, Kota, Negara Bagian, Kode Pos, dan Alamat Email.
  - ***Record*** → *Record* disebut juga *tuple*, adalah karakteristik atau fakta tunggal tentang suatu *entity*. Misalnya entitas CUSTOMER mungkin menyertakan ID *customer*, Nama Depan, Nama Belakang, Alamat, Kota, Negara Bagian, Kode Pos, dan Alamat Email.

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

Merancang *Entity Relationship Diagram* (ERD) (lanjutan…)

➢ Istilah–Istilah dalam ERD *Data Design*: (lanjutan…)

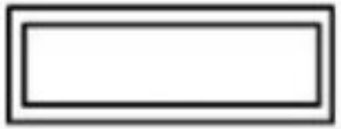
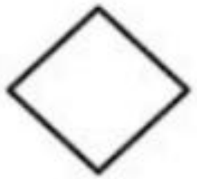
Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

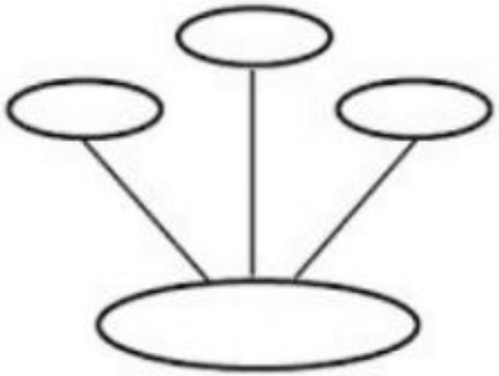
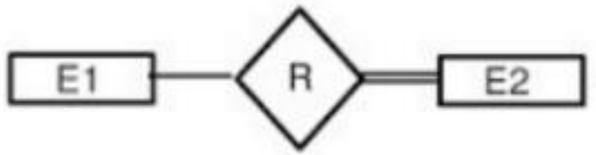
Notasi	Keterangan
	Entitas adalah suatu objek yang dapat diidentifikasi dalam lingkungan pemakai.
	Relasi menunjukkan adanya hubungan di antara sejumlah entitas yang berbeda.
	Atribut berfungsi mendeskripsikan karakter entitas (atribut yang berfungsi sebagai *key* diberi garis bawah).
Garis _____	Garis sebagai penghubung antara relasi dan entitas atau relasi dan entitas dengan atribut.

Merancang *Entity Relationship Diagram* (ERD) (lanjutan…)

➢ Istilah–Istilah dalam ERD *Data Design*: (lanjutan…)

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

Symbol	Keterangan
	= Entity
	= Weak Entity
	= Relationship
	= Identifying Relationship
	= Atribut
	= Atribut Kunci
	= Atribut Multivalue

Symbol	Keterangan
	= Atribut Komposit
	= Atribut Derivatif
	= Total Participation of E2 in R
	= Cardinality Ratio 1:N For E1:E2 in R

Merancang *Entity Relationship Diagram* (ERD) (lanjutan…)

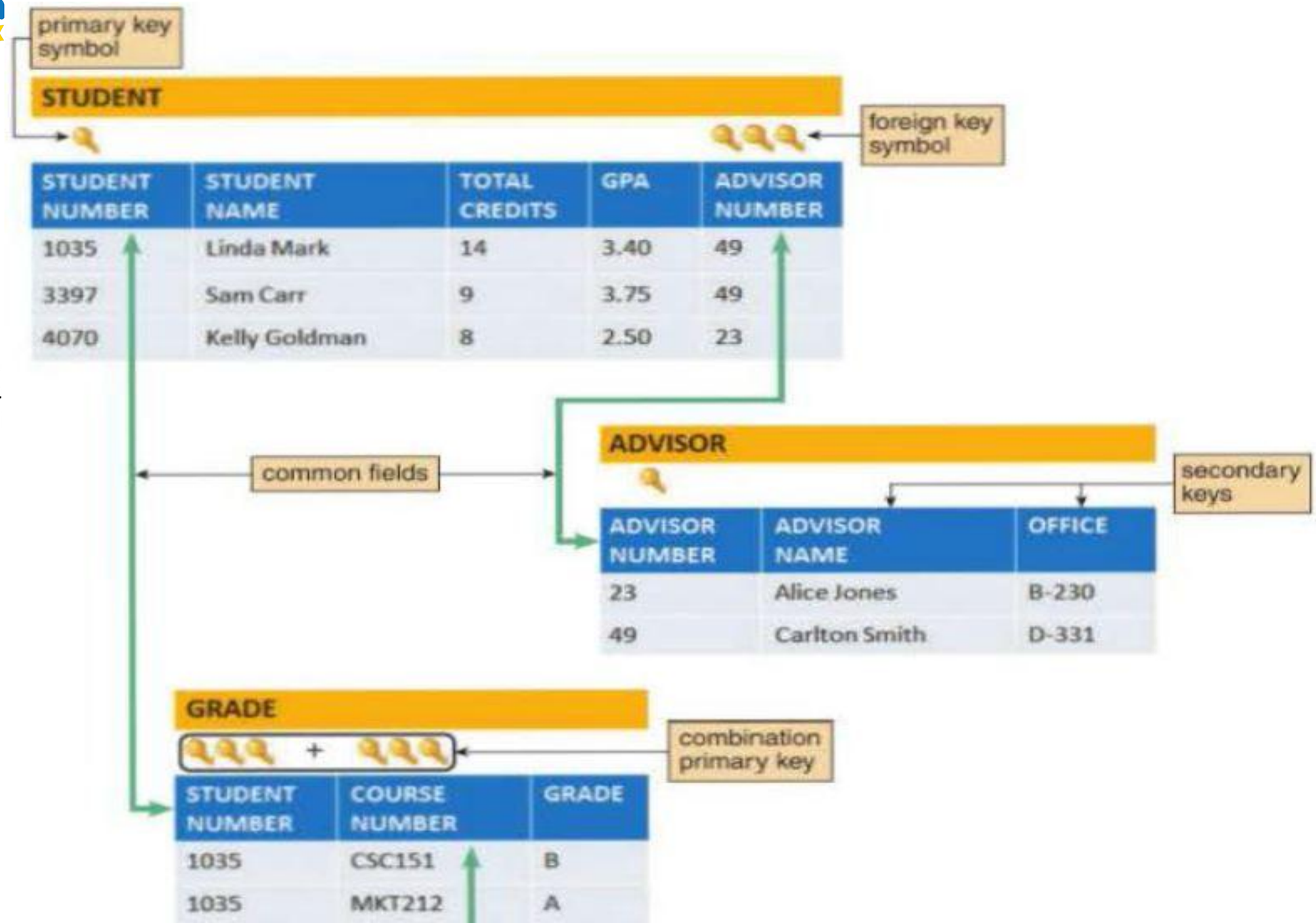
➢ Istilah–Istilah dalam ERD *Data Design*: (lanjutan…)

- ***Key field*** digunakan selama fase desain sistem digunakan untuk mengatur, mengakses, dan memelihara struktur data. Ada empat jenis ***key field*** yaitu ***primary keys*****,** ***candidate keys*****,** ***foreign keys*****,** dan ***secondary keys*****.**
  - ***Primary Key*** → *Primary key* adalah *field*/kombinasi field yang secara unik dan minimal mengidentifikasi anggota tertentu dari suatu entitas misalnya nomor induk Mahasiswa.
  - ***Candidate Key*** → *Field* yang dapat berfungsi sebagai *primary key* disebut sebagai *candidate key* misalnya pengganti nomor induk Mahasiswa bisa dengan NIK.
  - ***Foreign Key*** → *Recall field* yang ada di lebih dari satu tabel dan dapat digunakan untuk membentuk hubungan atau tautan antar tabel.
  - ***Secondary Key*** → *Secondary key* adalah *field* atau kombinasi *field* yang dapat digunakan untuk mengakses atau mengambil *record* dimana *value* dari *secondary key* tidak unik contohnya adalah nomor kode pos dari seorang Mahasiswa.

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

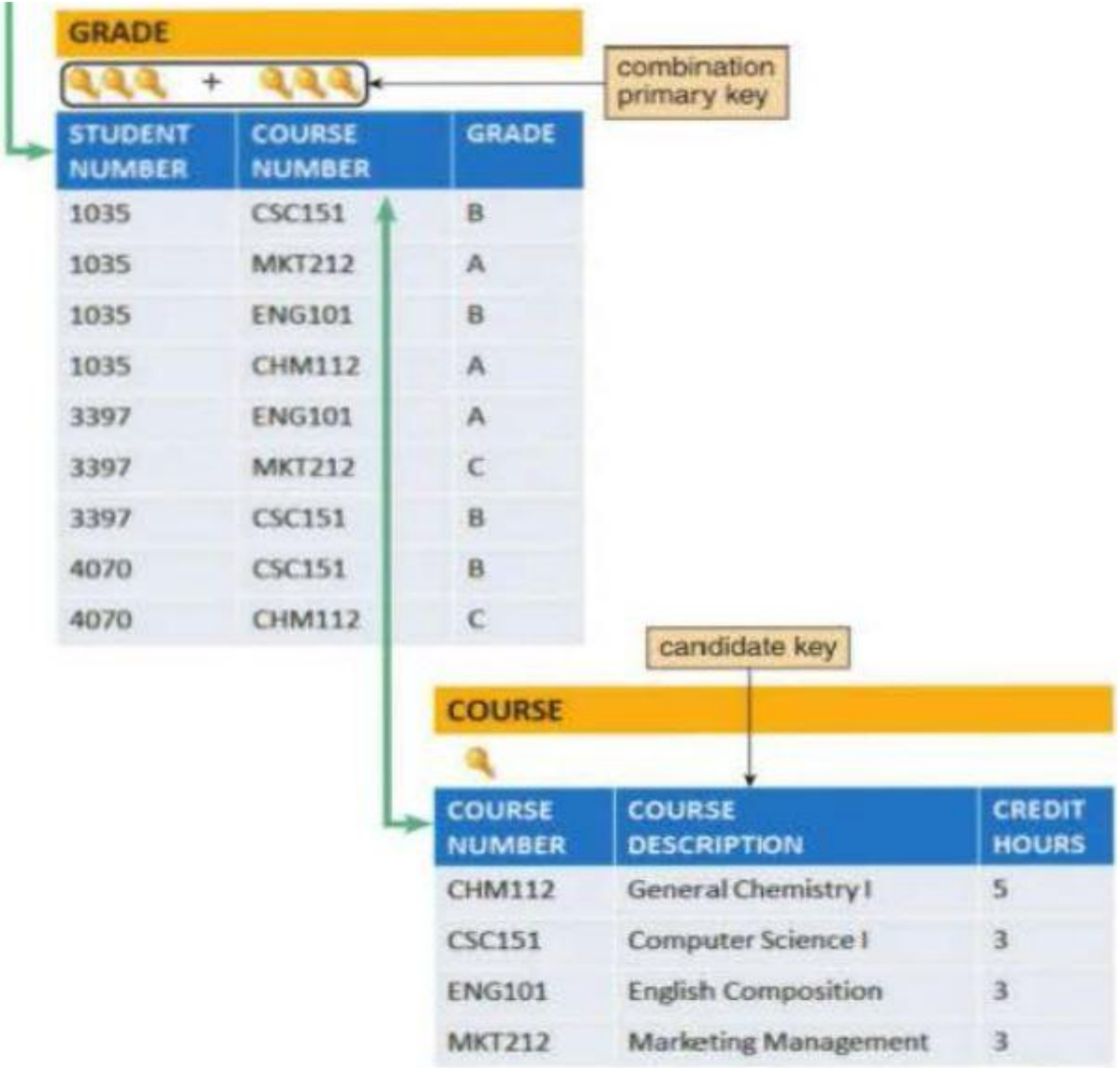
Merancang *Entity Relationship Diagram* (ERD) (lanjutan…)

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.



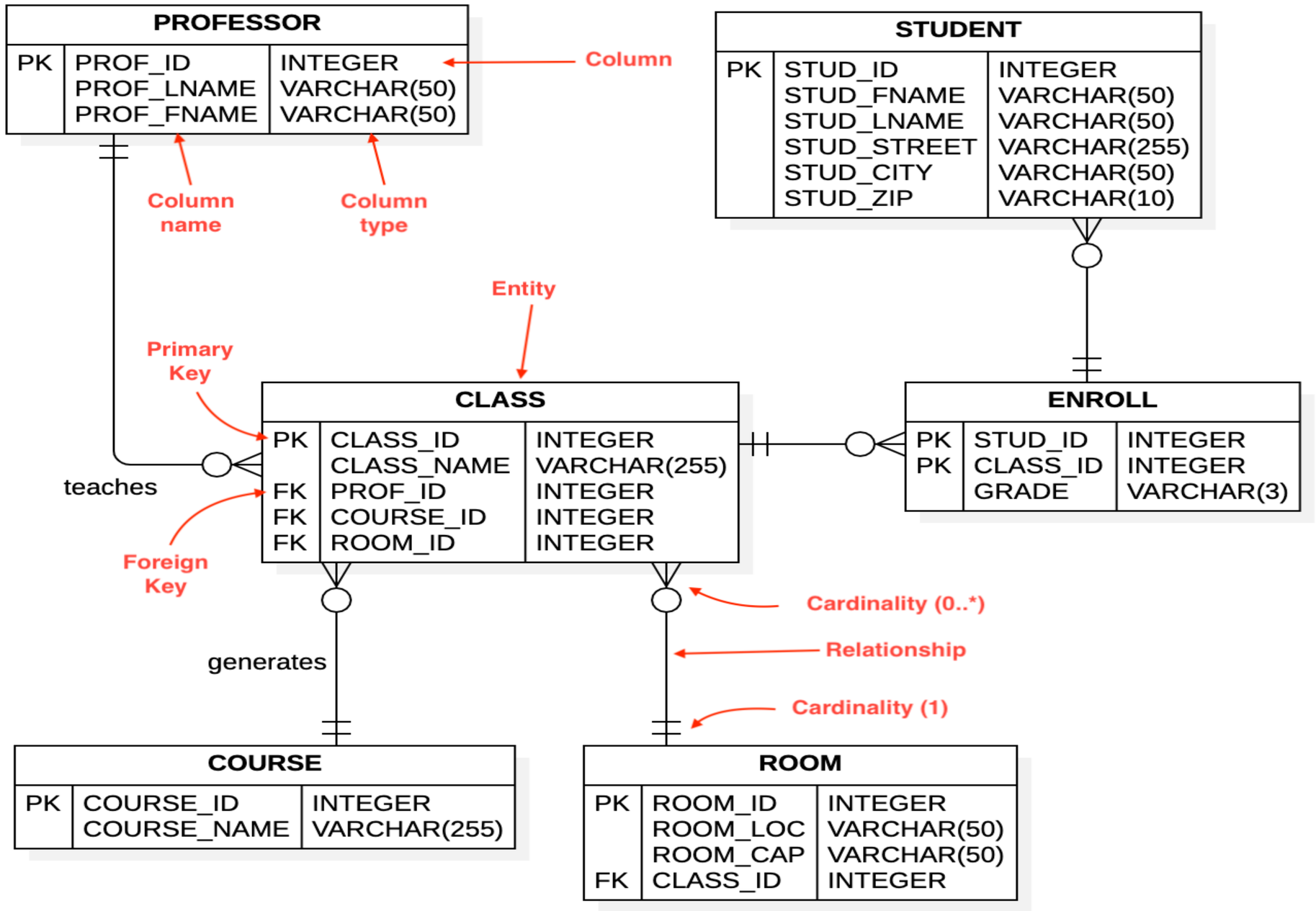
Merancang *Entity Relationship Diagram* (ERD) (lanjutan…)

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.



Merancang *Entity Relationship Diagram* (ERD) (lanjutan...)

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.



Merancang *Entity Relationship Diagram* (ERD) (lanjutan…)

➢ Cara Umum dalam Membuat ERD:

Langkah–langkah umum untuk membuat *entity relationship diagram*/ERD adalah:

1. **Lakukan identifikasi dan tentukan seluruh entitas yang akan digunakan** → Menentukan entitas apa saja yang akan digunakan dalam diagram, langkah ini dimulai dengan membuat sebuah persegi panjang lalu tuliskan deskripsi singkat mengenai entitas tersebut.

2. **Identifikasi dan menjelaskan relasi dari entitas** → Mencari entitas yang memiliki relasi, lalu membuat garis yang menghubungkan kedua entitas tersebut, serta harus menentukan relasi antara satu entitas dengan entitas yang lainnya. Dapat menggunakan simbol belah ketupat untuk mendeskripsikan hubungan relasinya, serta menjelaskan jenis relasi apa yang digunakan oleh suatu entitas apakah *one to one*, *one to many*, atau *many to many*.

3. **Tambahkan atribut yang diperlukan** → Tambahkan atribut–atribut yang diperlukan serta tentukan *atribut key* pada setiap entitas. Lambangkan *atribut key* tersebut dengan bentuk oval dan berikanlah deskripsi pada lambang tersebut.

4. **Lengkapi diagram** → Langkah terakhir adalah melengkapi diagram sesuai dengan kebutuhan dari sistem atau *database* yang sedang dikembangkan. Pada tahap ini harus lebih teliti untuk memeriksa pada setiap komponen seperti simbol beserta deskripsi yang salah atau mungkin tertukar.

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

Merancang *Entity Relationship Diagram* (ERD) (lanjutan…)

➢ Ada Tiga Tipe Relasi dalam ERD yaitu:

- ***One–to–One***

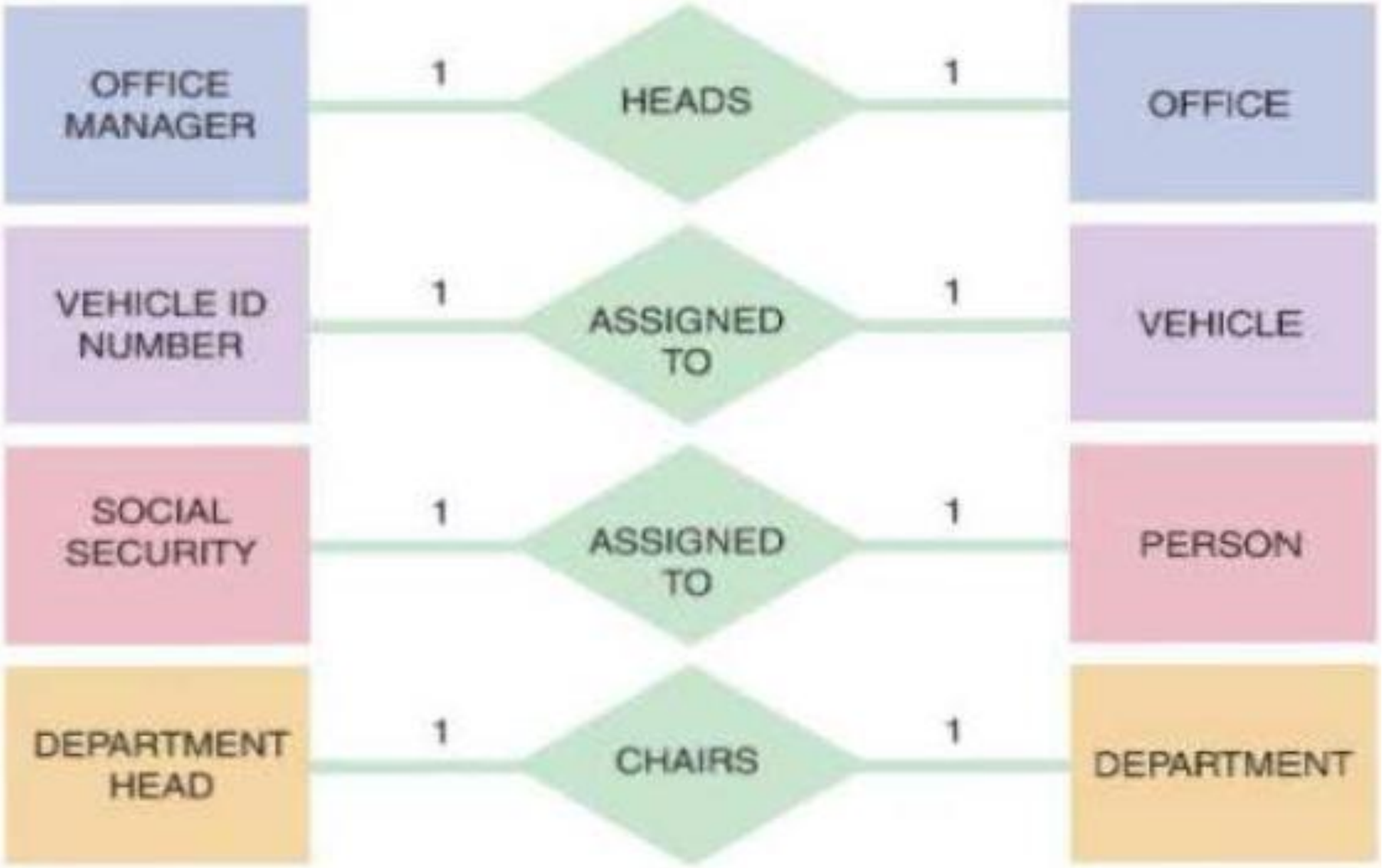


FIGURE 9-13 Examples of one-to-one (1:1) relationships.

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

Merancang *Entity Relationship Diagram* (ERD) (lanjutan…)

➢ Ada Tiga Tipe Relasi dalam ERD yaitu: (lanjutan…)

- ***One–to–Many***

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

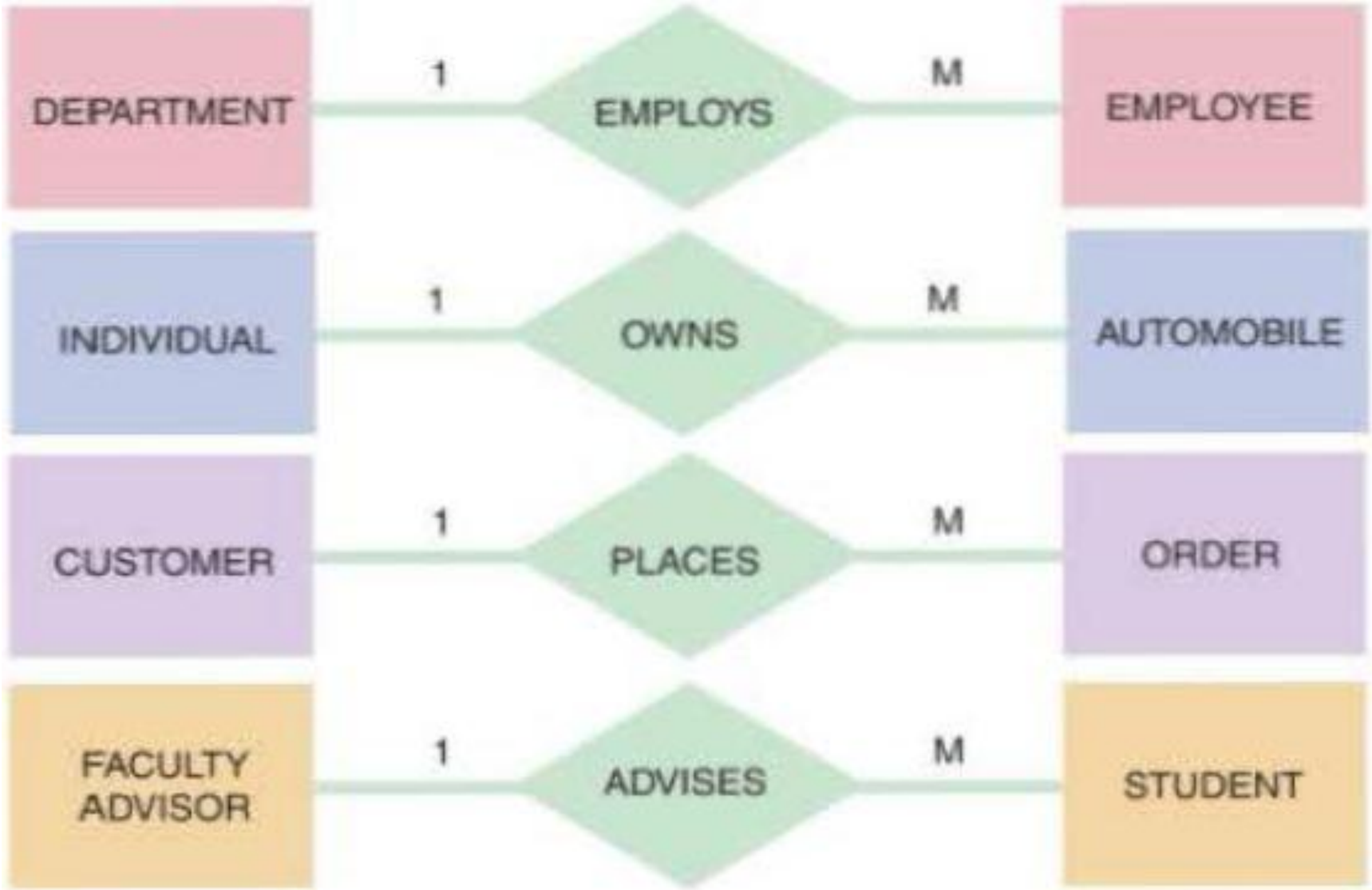


FIGURE 9-14 Examples of one-to-many (1:M) relationships.

Merancang *Entity Relationship Diagram* (ERD) (lanjutan…)

➢ Ada Tiga Tipe Relasi dalam ERD yaitu: (lanjutan…)

- ***Many–to–Many***

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

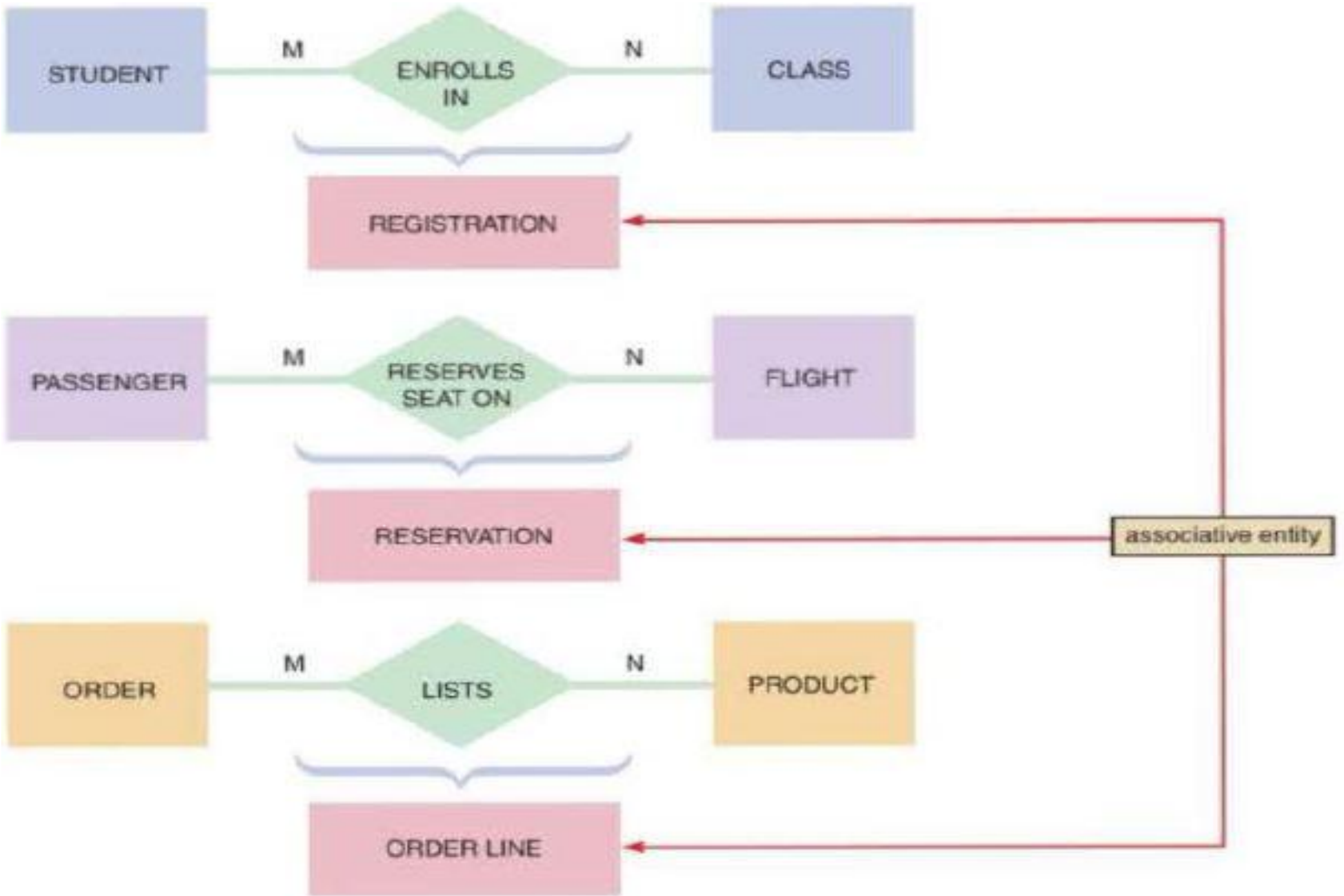


FIGURE 9-15 Examples of many-to-many (M:N) relationships. Notice that the event or transaction that links the two entities is an associative entry with its own set of attributes and characteristics.

Merancang *Entity Relationship Diagram* (ERD) (lanjutan…)

➢ Contoh ERD untuk Sistem Penjualan:

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

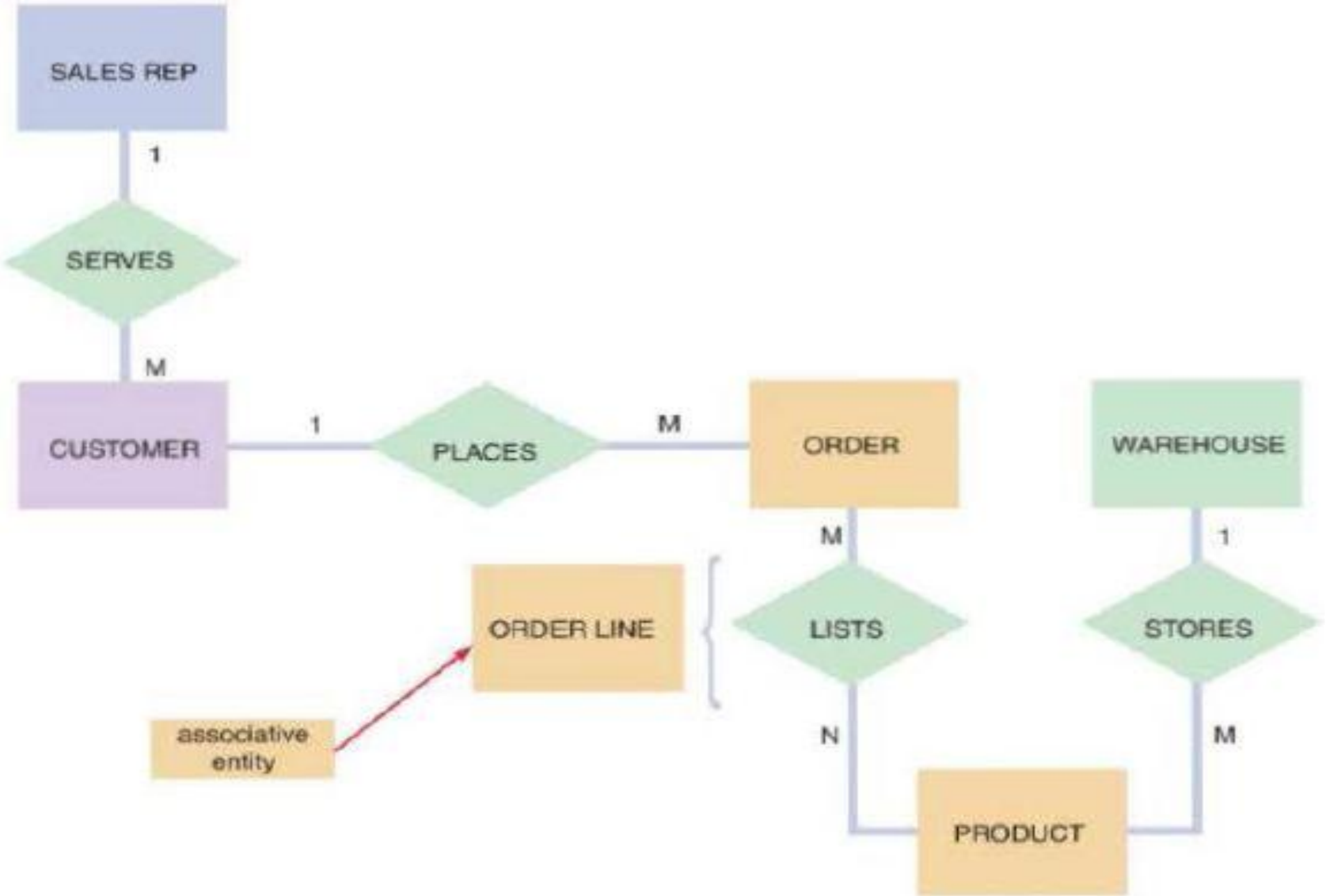


FIGURE 9-16 An entity-relationship diagram for SALES REP, CUSTOMER, ORDER, PRODUCT, and WAREHOUSE. Notice that the ORDER and PRODUCT entities are joined by an associative entity named ORDER LINE.

Merancang *Entity Relationship Diagram* (ERD) (lanjutan…)

➢ *Cardinality* dalam ERD:

- *Cardinality* menggambarkan hubungan numerik antara dua entitas dan menunjukkan bagaimana *instance* dari satu entitas berhubungan dengan *instance* entitas lain.

- Misalnya hubungan antara dua entitas: PELANGGAN dan PESANAN.

- Satu pelanggan dapat memiliki satu pesanan, banyak pesanan, atau tidak ada pesanan sama sekali, tetapi masing–masing pesanan harus memiliki satu dan hanya satu pelanggan.

- Seorang analis sistem dapat memodelkan interaksi ini dengan menambahkan notasi kardinalitas yang menggunakan simbol khusus untuk mewakili hubungan/relasi.

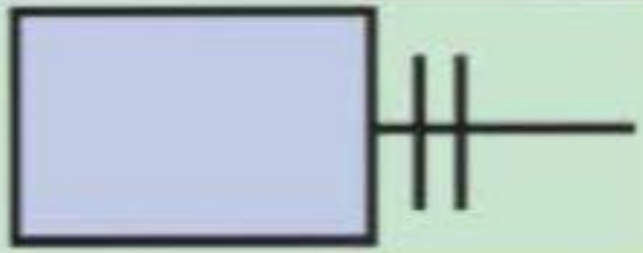
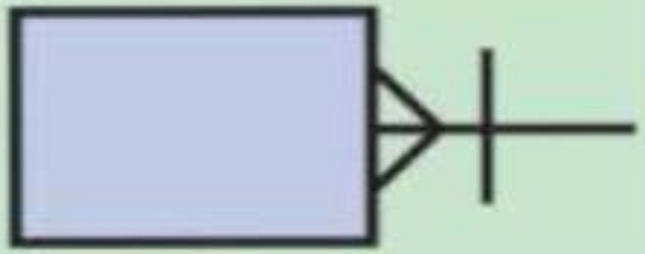
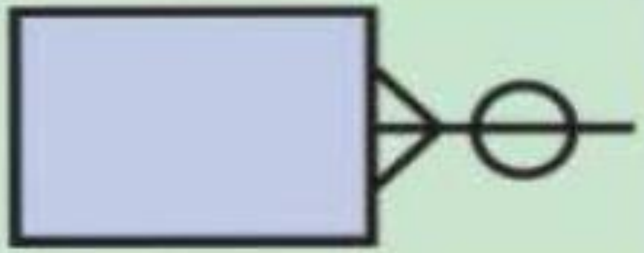
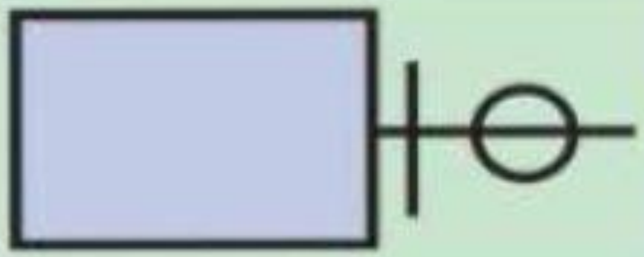
- Disebut notasi kaki gagak (***crow's foot***) karena bentuknya yang meliputi lingkaran, batang, dan simbol yang menunjukkan berbagai kemungkinan.

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

Merancang *Entity Relationship Diagram* (ERD) (lanjutan…)

➢ *Cardinality* dalam ERD: (lanjutan…)

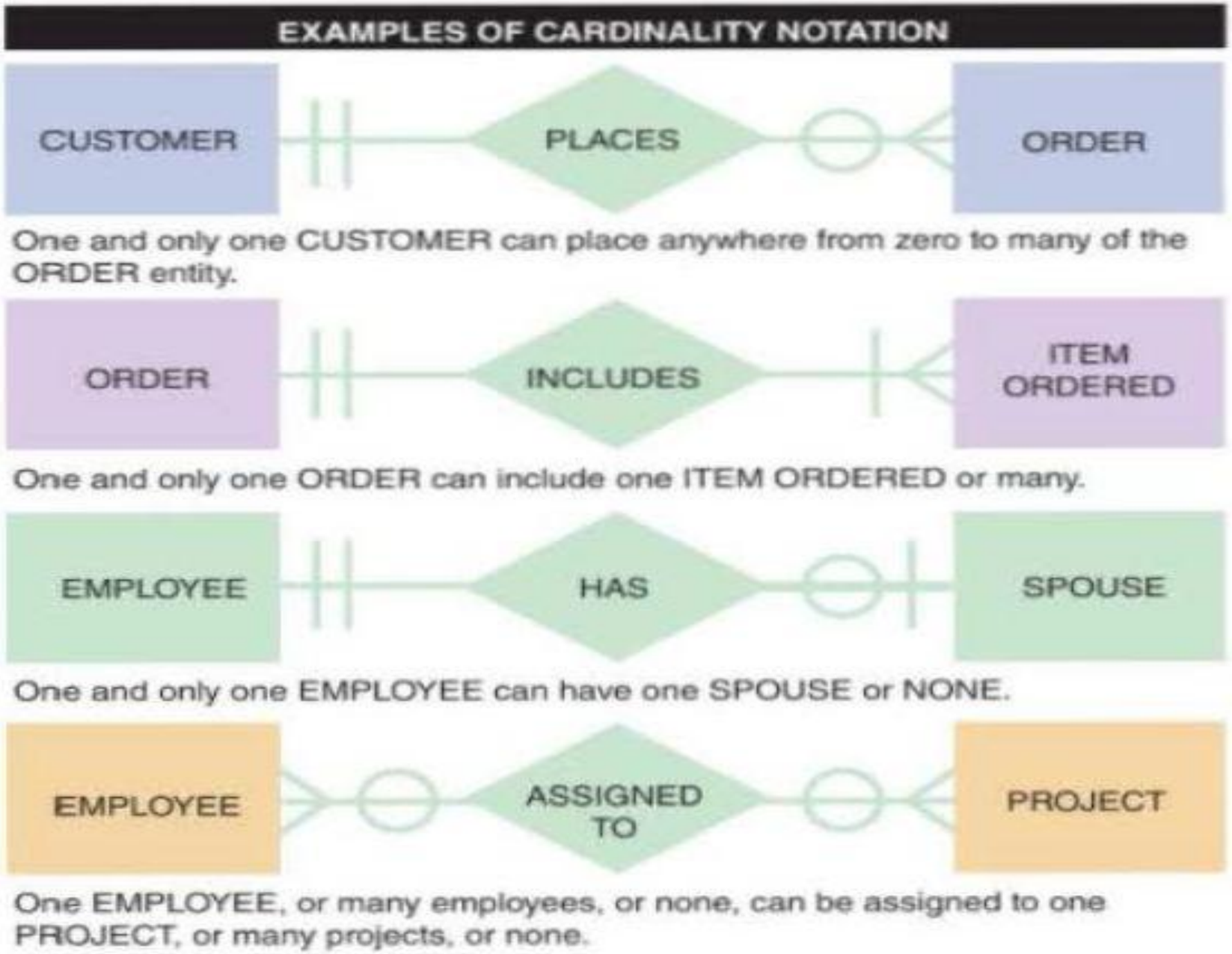
Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

SYMBOL	MEANING	UML REPRESENTATION
	One and only one	1
	One or many	1..*
	Zero, or one, or many	0..*
	Zero, or one	0..1

Merancang *Entity Relationship Diagram* (ERD) (lanjutan…)

➢ *Cardinality* dalam ERD: (lanjutan…)

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.



Merancang *Entity Relationship Diagram* (ERD) (lanjutan…)

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 269.

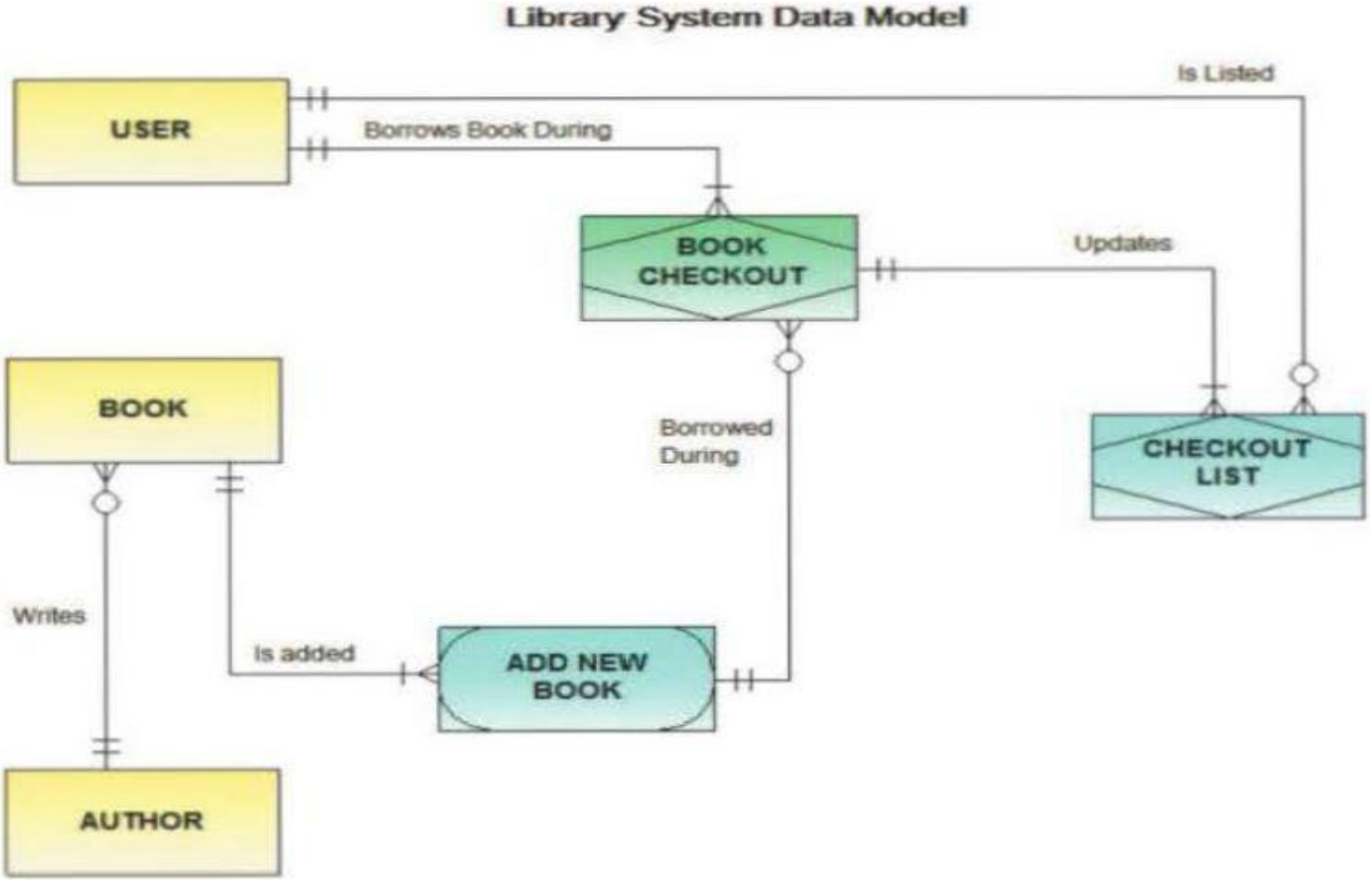


FIGURE 9-19 An ERD for a library system drawn with Visible Analyst. Notice that crow's foot notation has been used and relationships are described in both directions.

REFERENSI

- Tilley, Scott, System Analysis and Design, CENGAGE, USA, 2020.
- Dennis, Alan, System Analysis and Design: An Object Oriented Approach with UML, Wiley, USA, 2015.
- Satzinger, Jackson, Burd, System Analysis and Design in A Changing World, CENGANE, USA, 2012.
- Langer, Arthur, Analysis and Design of Information Systems, Springer, USA, 2008.
- Bentlet, Whitten, System Analysis and Design Methods, McGraw–Hill Irwin, USA, 2007.
- Wasson, Charles, System Analysis, Design, and Development, Concepts, Principles, and Practices, Wiley–Interscience, Canada, 2006.
- researchgate.com
- salaryexplorer.com.

TERIMA KASIH