DATA BASE SYSTEMS

LECTURE 3

PROPOSED BY

DR: ALSHAIMAA MOSTAFA MOHAMMED

CHAPTER 2

E-R MODELS

Types of keys

- A composite key is a key that is composed of more than one attribute.
- An attribute that is a part of a key is called a key attribute.
- superkey is a key that can uniquely identify any row in the table. In other words,

a superkey functionally determines every attribute in the row.

- A candidate key is a minimal superkey—that is, a superkey without any unnecessary attributes. A candidate key is based on a full functional dependency.
- Entity integrity is the condition in which each row (entity instance) in the table has Its own unique identity.
- To ensure entity integrity, the primary key has two requirements:
  (1) all of the values in the primary key must be unique and
  (2) no key attribute in the primary key can contain a null
- Null values are problematic in the relational model.
- A null is the absence of any data value, and it is never allowed in any part of the primary key.

- A foreign key (FK) is the primary key of one table that has been placed into another table to create a common attribute.
- Foreign keys are used to ensure referential integrity, the condition in which every reference to an entity instance by another entity instance is valid.
- In other words, every foreign key entry must either be null or a valid value in the primary key of the related table.
- Secondary key is defined as a key that is used strictly for data retrieval purposes.

- Controlled redundancy:
  - Makes the relational database work
  - Tables within the database share common attributes
    - Enables tables to be linked together
  - Multiple occurrences of values not redundant when required to make the relationship work
  - Redundancy exists only when there is unnecessary duplication of attribute values

NULLS

- No data entry
- Not permitted in primary key
- Should be avoided in other attributes
- Can represent
  - An unknown attribute value
  - A known, but missing, attribute value
  - A “not applicable” condition

–Can create problems when functions such as COUNT, AVERAGE, and SUM are used

–Can create logical problems when relational tables are linked

FIGURE 3.2 An example of a simple relational database

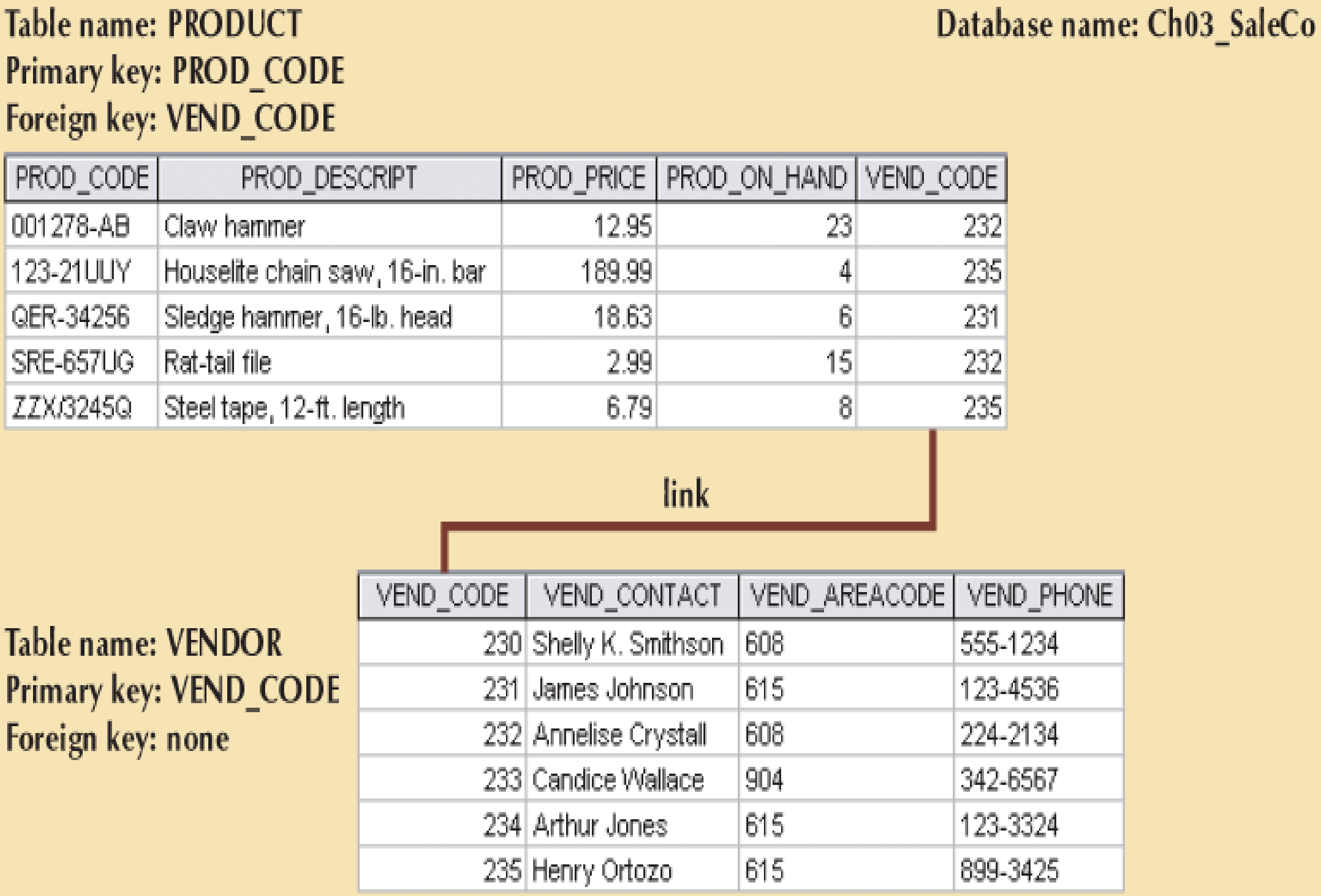


Table name: PRODUCT
Primary key: PROD_CODE
Foreign key: VEND_CODE

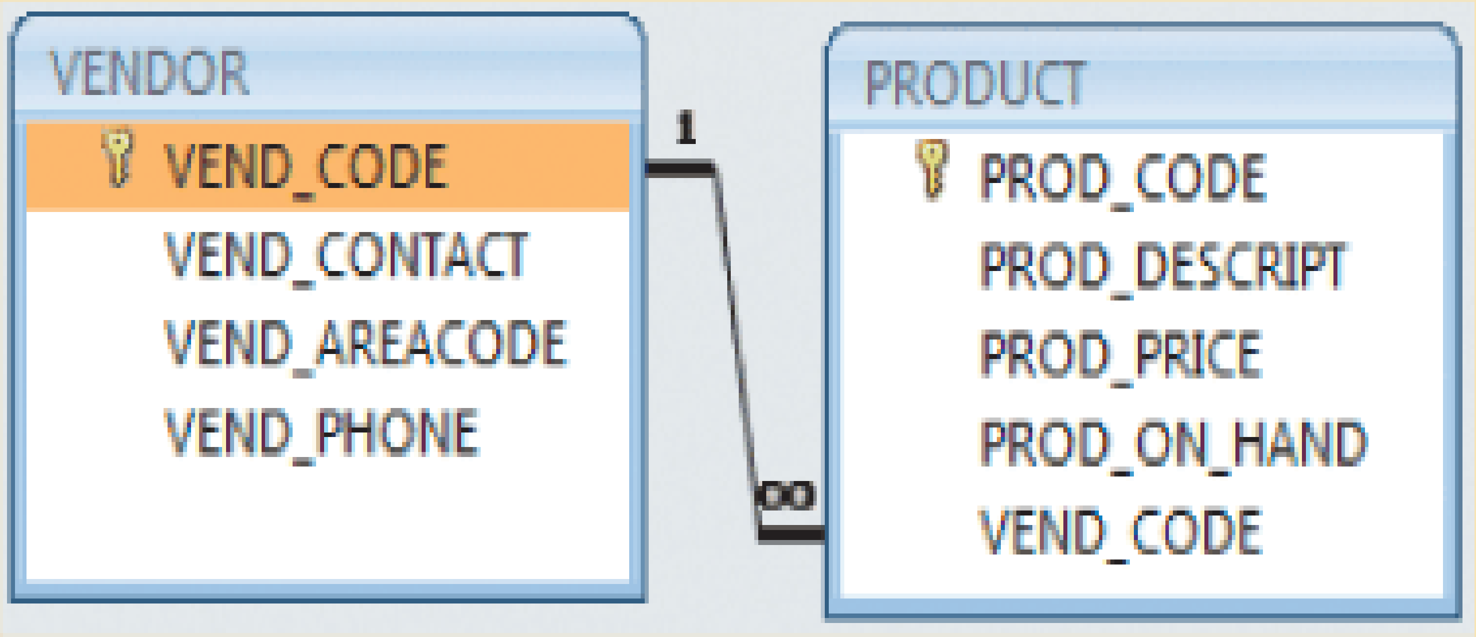
Database name: Ch03_SaleCo

PROD_CODE	PROD_DESCRIPT	PROD_PRICE	PROD_ON_HAND	VEND_CODE
001278-AB	Claw hammer	12.95	23	232
123-21UUY	Houselite chain saw, 16-in. bar	189.99	4	235
QER-34256	Sledge hammer, 16-lb. head	18.63	6	231
SRE-657UG	Rat-tail file	2.99	15	232
ZZX/3245Q	Steel tape, 12-ft. length	6.79	8	235

Table name: VENDOR
Primary key: VEND_CODE
Foreign key: none

VEND_CODE	VEND_CONTACT	VEND_AREACODE	VEND_PHONE
230	Shelly K. Smithson	608	555-1234
231	James Johnson	615	123-4536
232	Annelise Crystall	608	224-2134
233	Candice Wallace	904	342-6567
234	Arthur Jones	615	123-3324
235	Henry Ortozo	615	899-3425

FIGURE 3.3 The relational diagram for the Ch03_SaleCo database



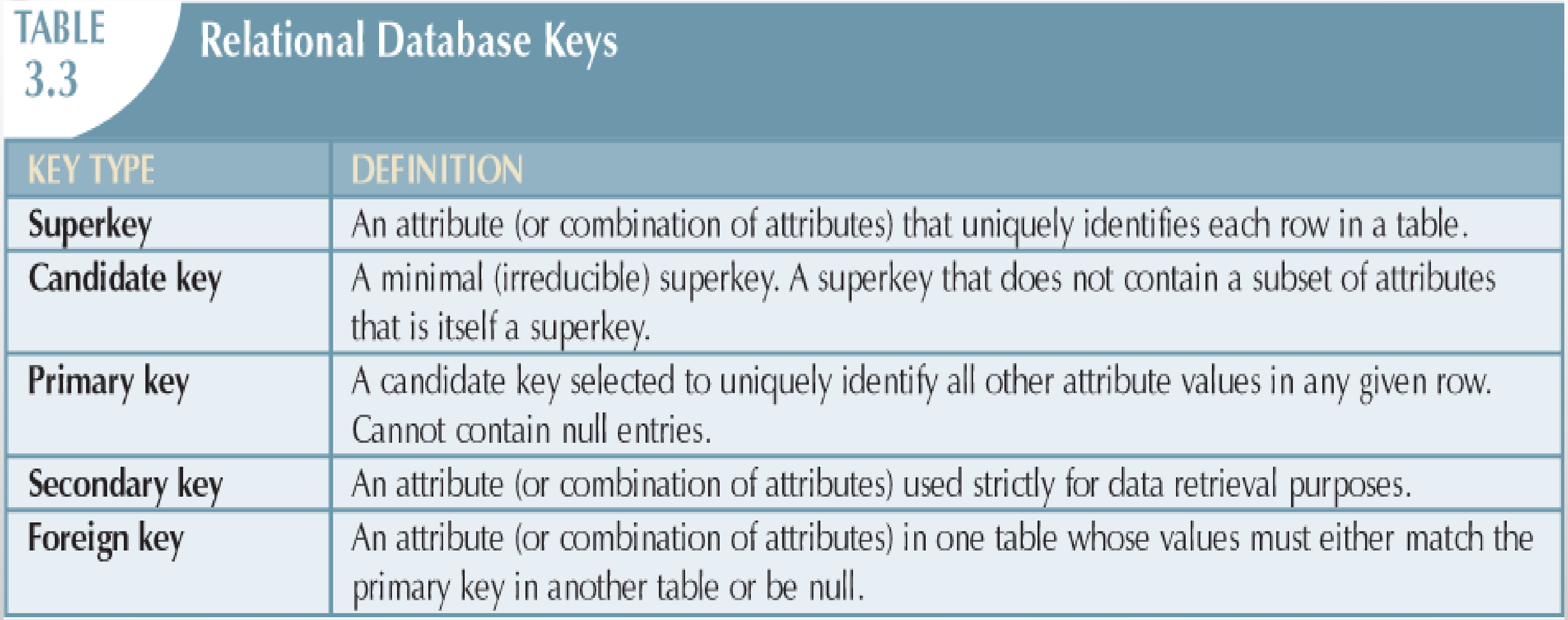


TABLE 3.3 Relational Database Keys

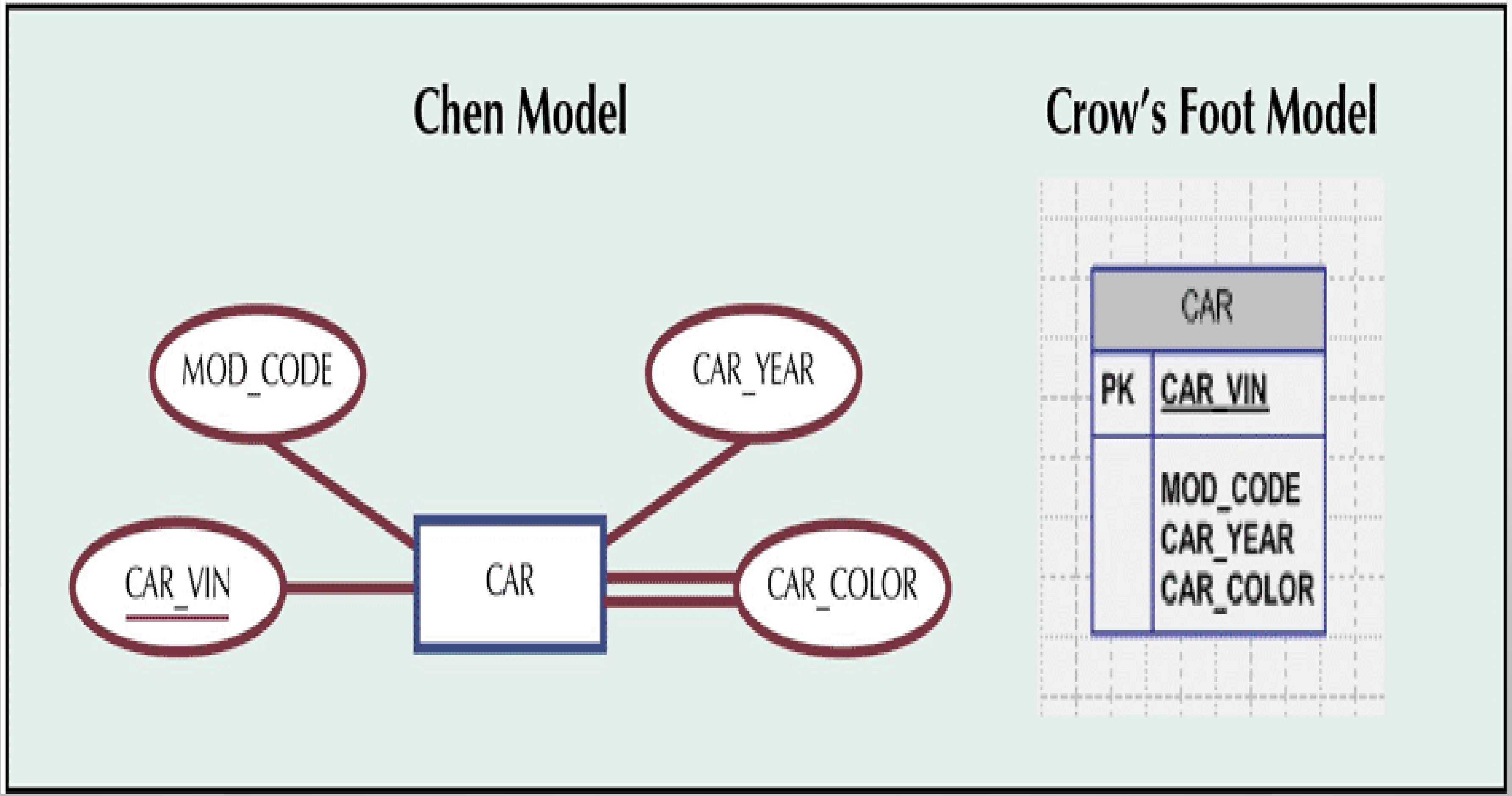
KEY TYPE	DEFINITION
Superkey	An attribute (or combination of attributes) that uniquely identifies each row in a table.
Candidate key	A minimal (irreducible) superkey. A superkey that does not contain a subset of attributes that is itself a superkey.
Primary key	A candidate key selected to uniquely identify all other attribute values in any given row. Cannot contain null entries.
Secondary key	An attribute (or combination of attributes) used strictly for data retrieval purposes.
Foreign key	An attribute (or combination of attributes) in one table whose values must either match the primary key in another table or be null.

//Composite and simple attributes

- **Composite attribute**
  - Not to be confused with composite key.
  - This is an attribute that can be broken down into more atomic attributes
    - Address can be divided into street, city, state and zip
- **Simple attribute**– no further division possible
  - ➢ To facilitate detailed queries, it is wise to change composite attributes into a series of simple attributes.
- **Single-value attribute** – can have only one value (social security number)
  - ➢ a single-valued attribute is not necessarily a simple attribute.
- **Multivalued attributes** – can have many values.
  - ➢ a person may have several college degrees,
- **Multivalued attributes are shown by a double line connecting the attribute to the entity.**

A multivalued attribute in an entity

FIGURE 4.3 A MULTIVALUED ATTRIBUTE IN AN ENTITY



Resolving multivalued attribute problems

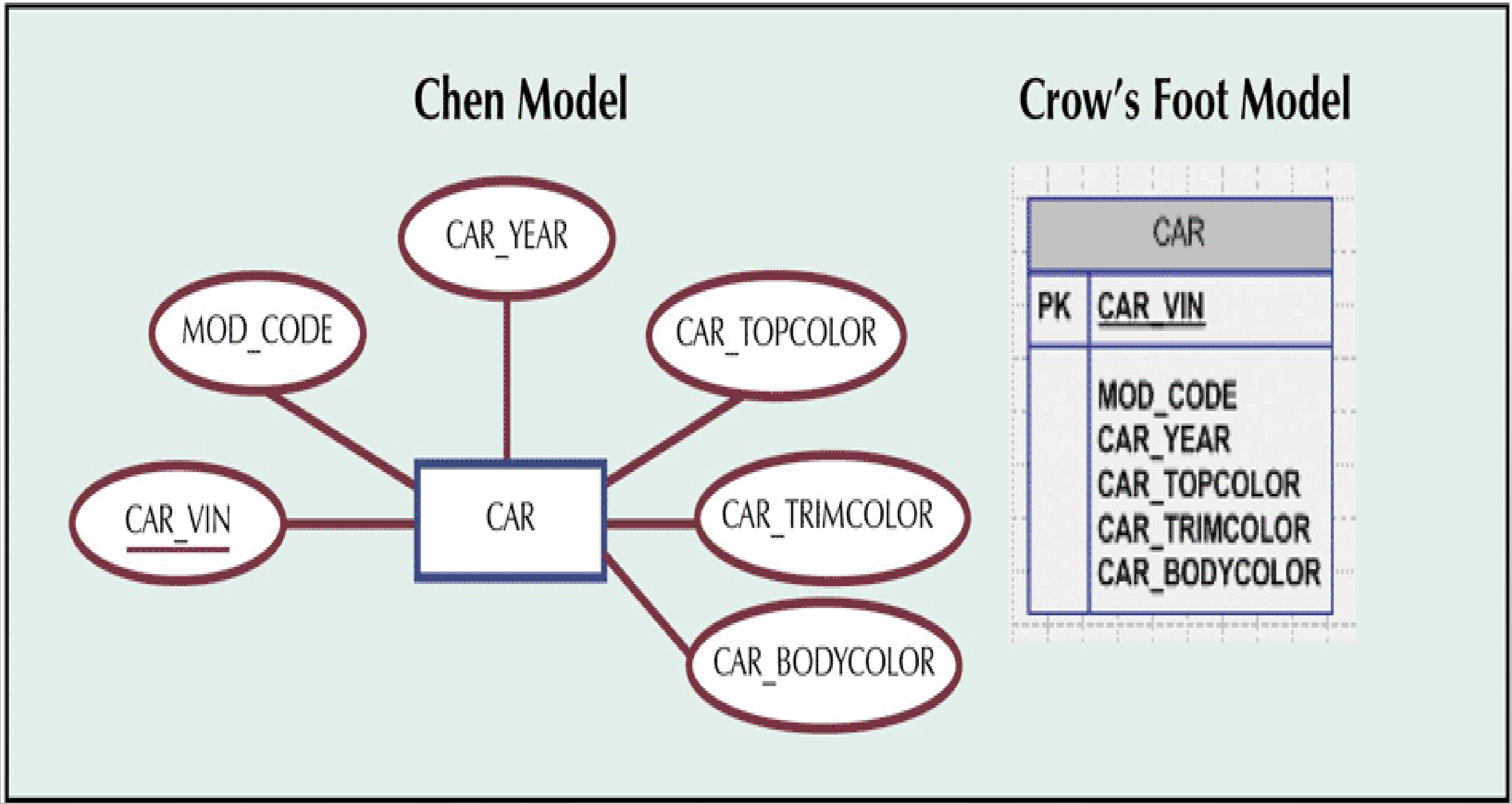
- **Although the conceptual model can handle multivalued attributes, *you should not implement them in the relational DBMS.* Instead, follow one of these two options**

1. Within original entity, create several new attributes, one for each of the original multivalued attribute's components

    - ■ CAR_COLOR can be split into CAR_TOPCOLOR, CAR_BODYCOLOR and CAR_TRIMCOLOR
    - ■ Can lead to major structural problems in the table.
        - If some cars have many types of colors and others have few colors, then all cars need to have attributes to handle the maximum number of colors. But many of those fields will be null for many rows.

Splitting the multivalued attribute into new attributes

FIGURE 4.4 SPLITTING THE MULTIVALUED ATTRIBUTE INTO NEW ATTRIBUTES



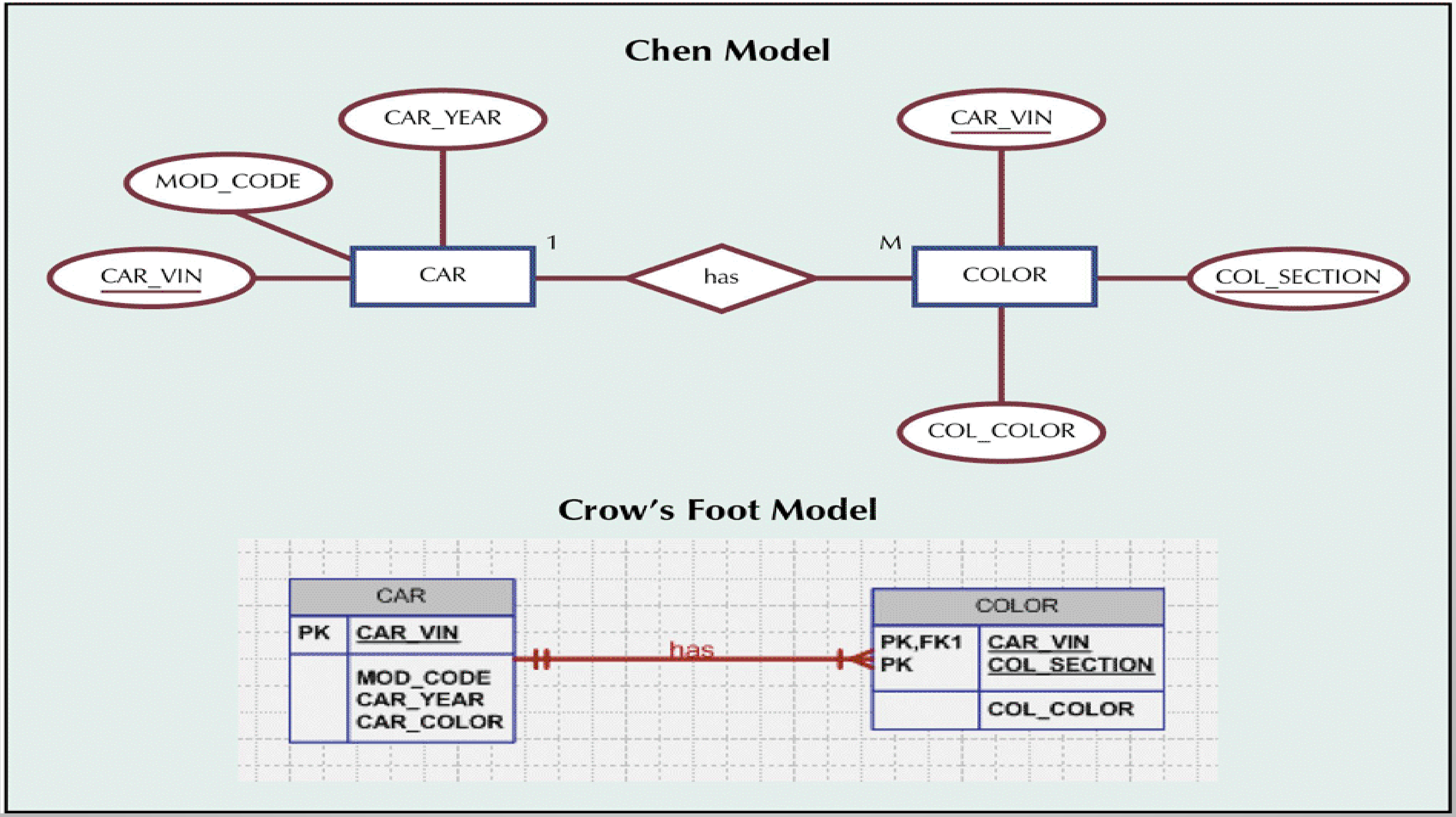
CAR	
PK	CAR_VIN
	MOD_CODE CAR_YEAR CAR_TOPCOLOR CAR_TRIMCOLOR CAR_BODYCOLOR

Resolving multivalued attribute problems

2. Create a new entity composed of the original multivalued attribute’s components.
   - The new entity is related to the original entity in a 1:M relationship
   - Color needs to be defined only for those sections that have color. This is done in the COL_SECTION attribute

A new entity set composed of a multivalued attribute's components

FIGURE 4.5 A New Entity Set Composed of a Multivalued Attribute's Components

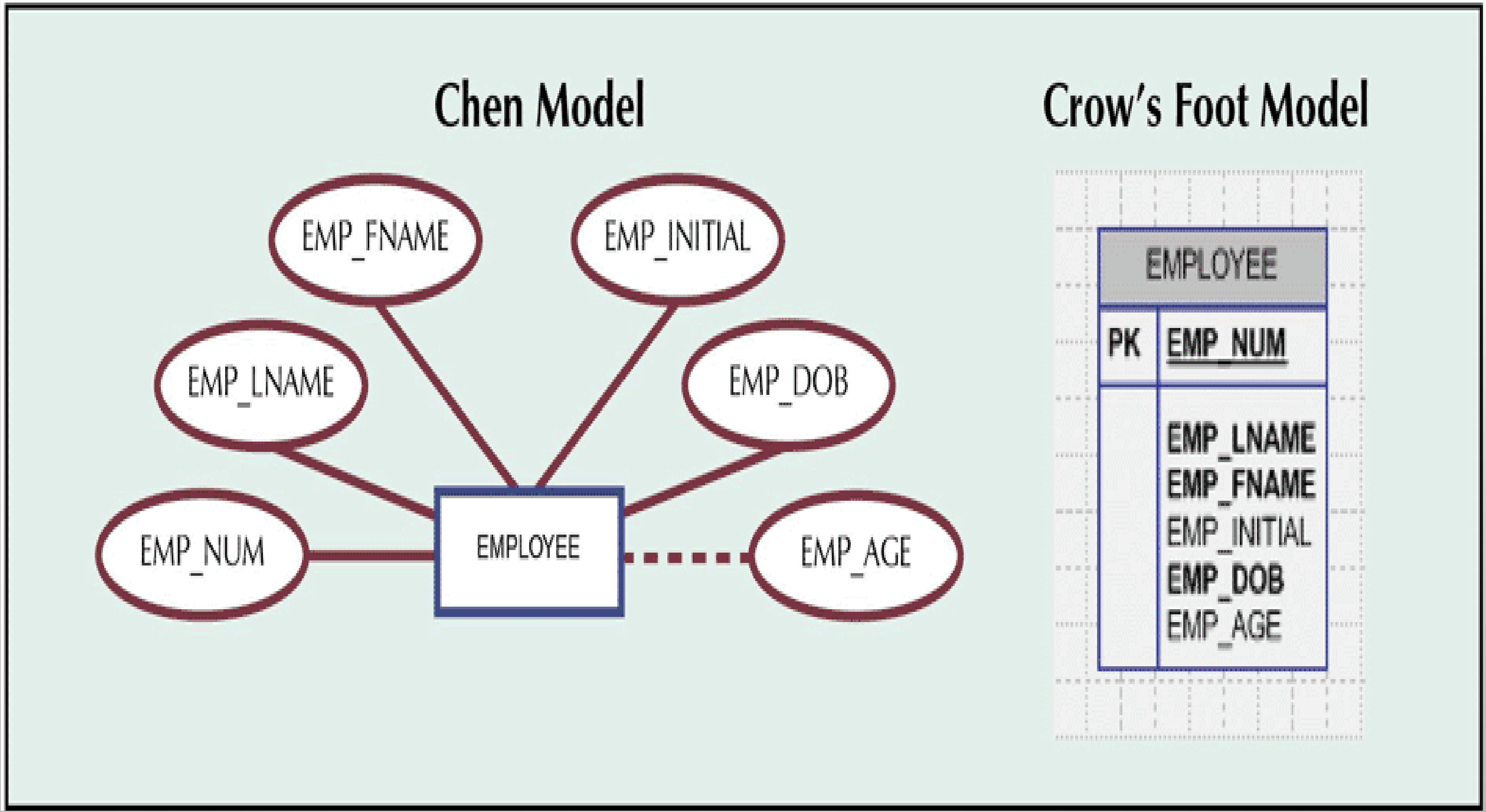


Derived attributes

- **Attribute whose value may be calculated (derived) from other attributes**
  - Age can be calculated by subtracting date of birth from current date
- **Need not be physically stored within the database but can be based on processing requirements**
- **Can be derived by using an algorithm**
- **Denoted by a dashed line in the Chen model**

Depiction of a derived attribute

FIGURE 4.6 DEPICTION OF A DERIVED ATTRIBUTE



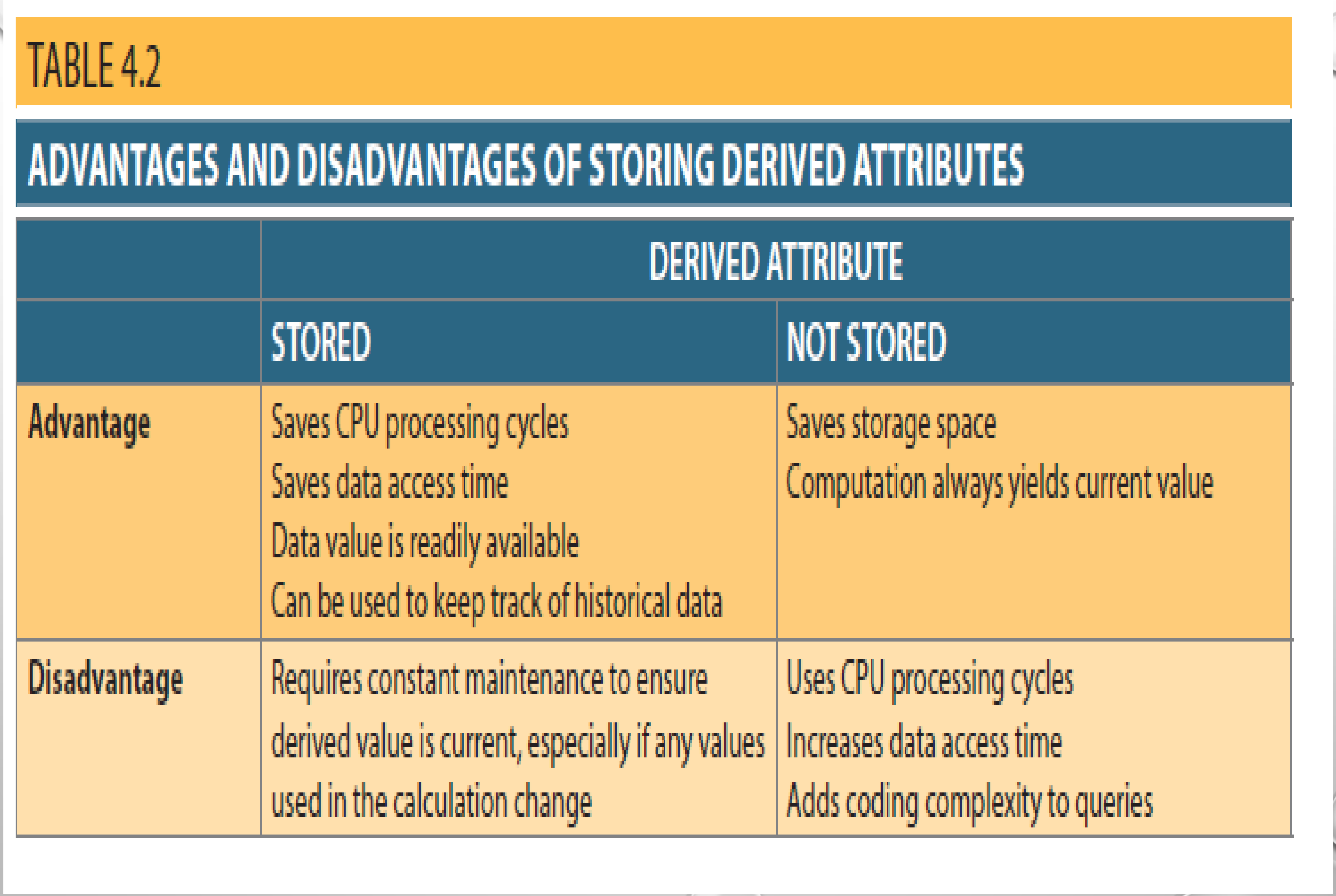


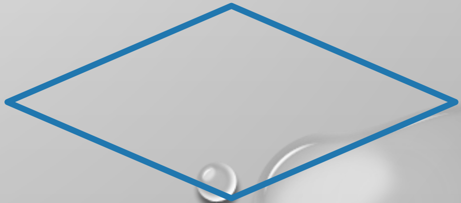
TABLE 4.2

ADVANTAGES AND DISADVANTAGES OF STORING DERIVED ATTRIBUTES

	DERIVED ATTRIBUTE	
	STORED	NOT STORED
Advantage	Saves CPU processing cycles Saves data access time Data value is readily available Can be used to keep track of historical data	Saves storage space Computation always yields current value
Disadvantage	Requires constant maintenance to ensure derived value is current, especially if any values used in the calculation change	Uses CPU processing cycles Increases data access time Adds coding complexity to queries

Relationships

- A relationship is the association among several entities. It connects different entities through a meaningful relation.
- A relationship set is a set of relationships of the same type.
- each relationship is identified by a name that describes the relationship. The relationship name is an active or passive verb.
  - for example, a STUDENT takes a CLASS
- Represented by diamond shapes



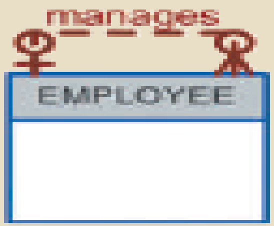
Relationship degree

- Indicates number of associated entities or participants
- Unary relationship
  - Association is maintained within a single entity (employee within the EMPLOYEE entity is the manager for one or more employees within that entity - EMPLOYEE has a relationship with itself.)
- Binary relationship
  - Two entities are associated(“a PROFESSOR teaches one or more CLASSes)

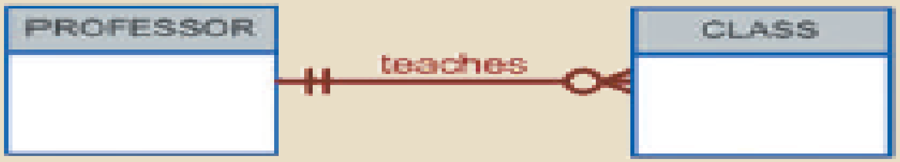
Relationship degree

- Ternary relationship
  - Three entities are associated
    - A DOCTOR writes one or more PRESCRIPTIONs.
    - A PATIENT may receive one or more PRESCRIPTIONs.
    - A DRUG may appear in one or more PRESCRIPTIONs.

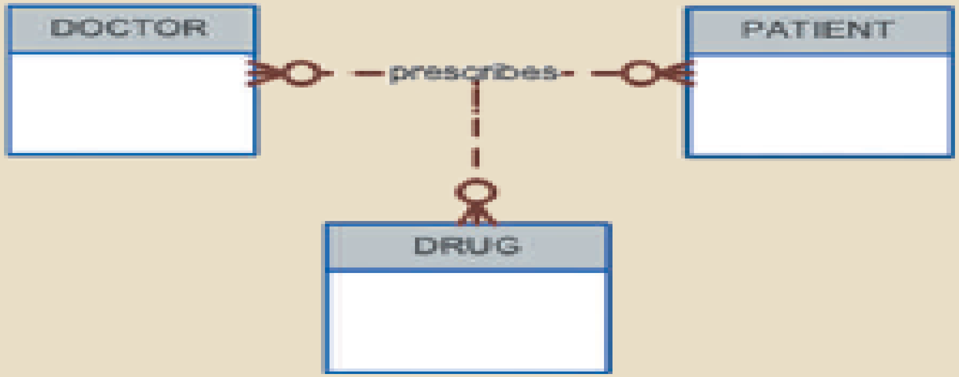
Unary relationship



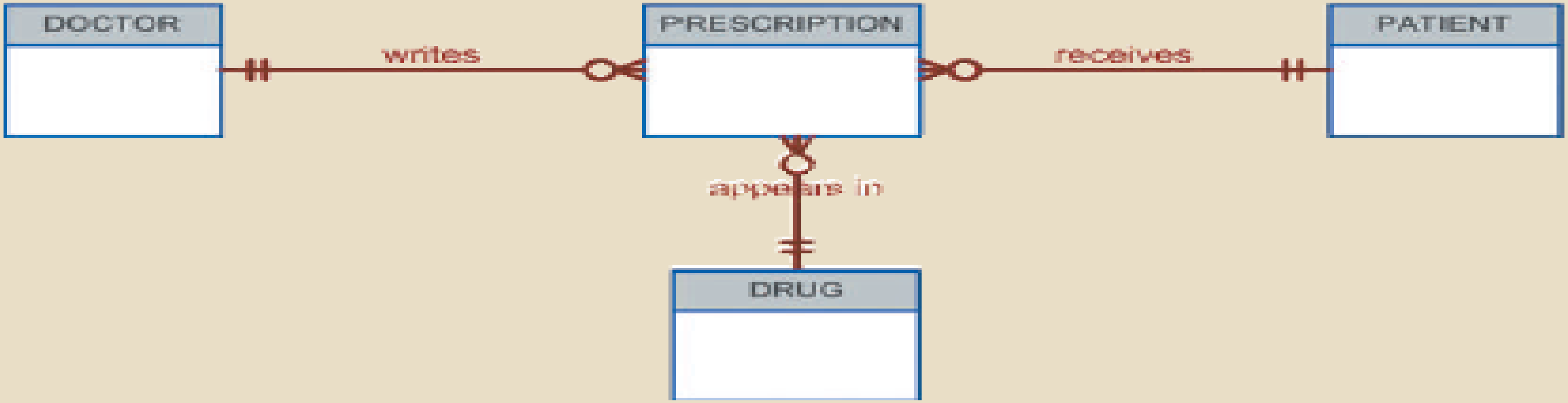
Binary relationship



Ternary relationship (Conceptual)

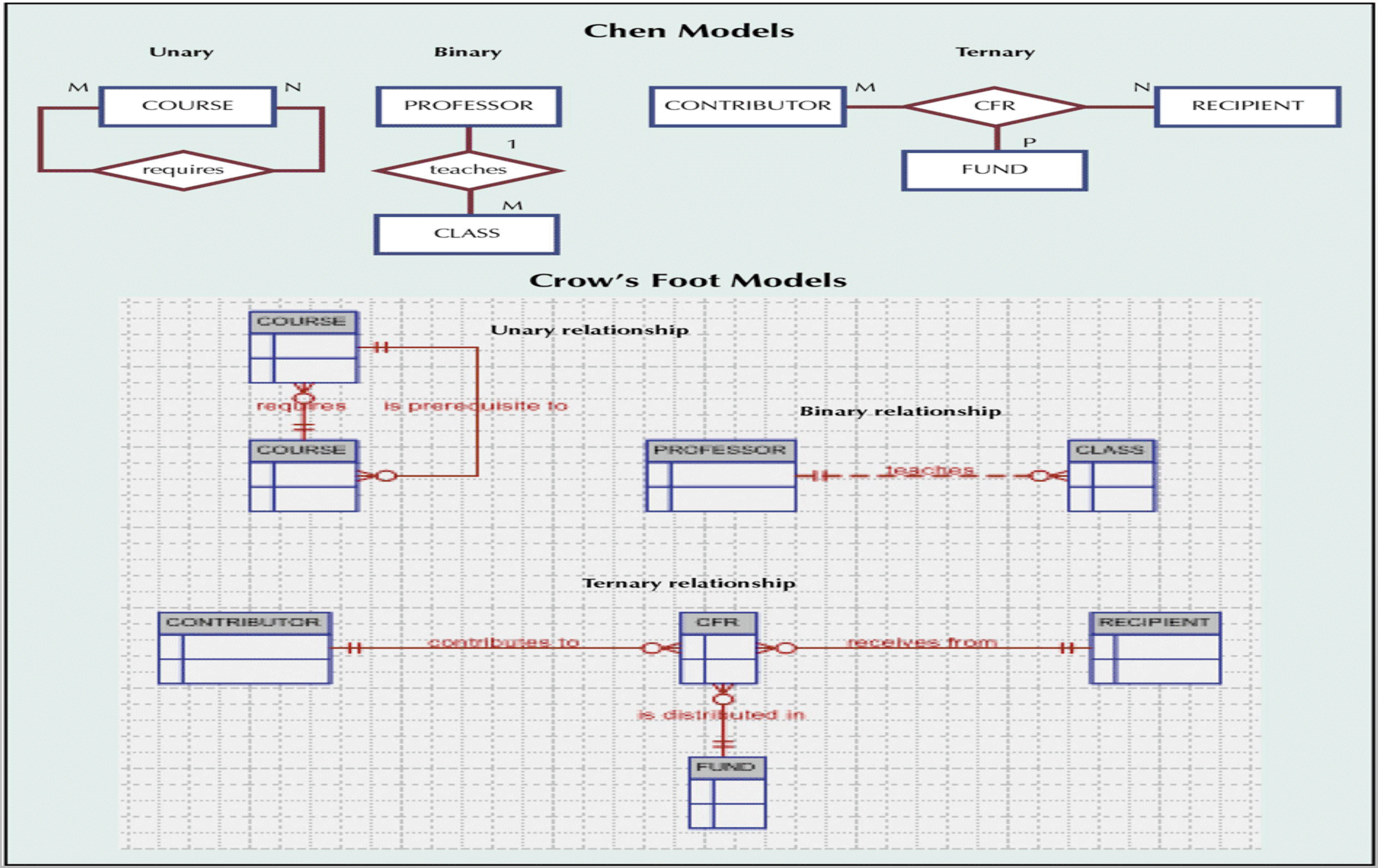


Ternary relationship (Logical)



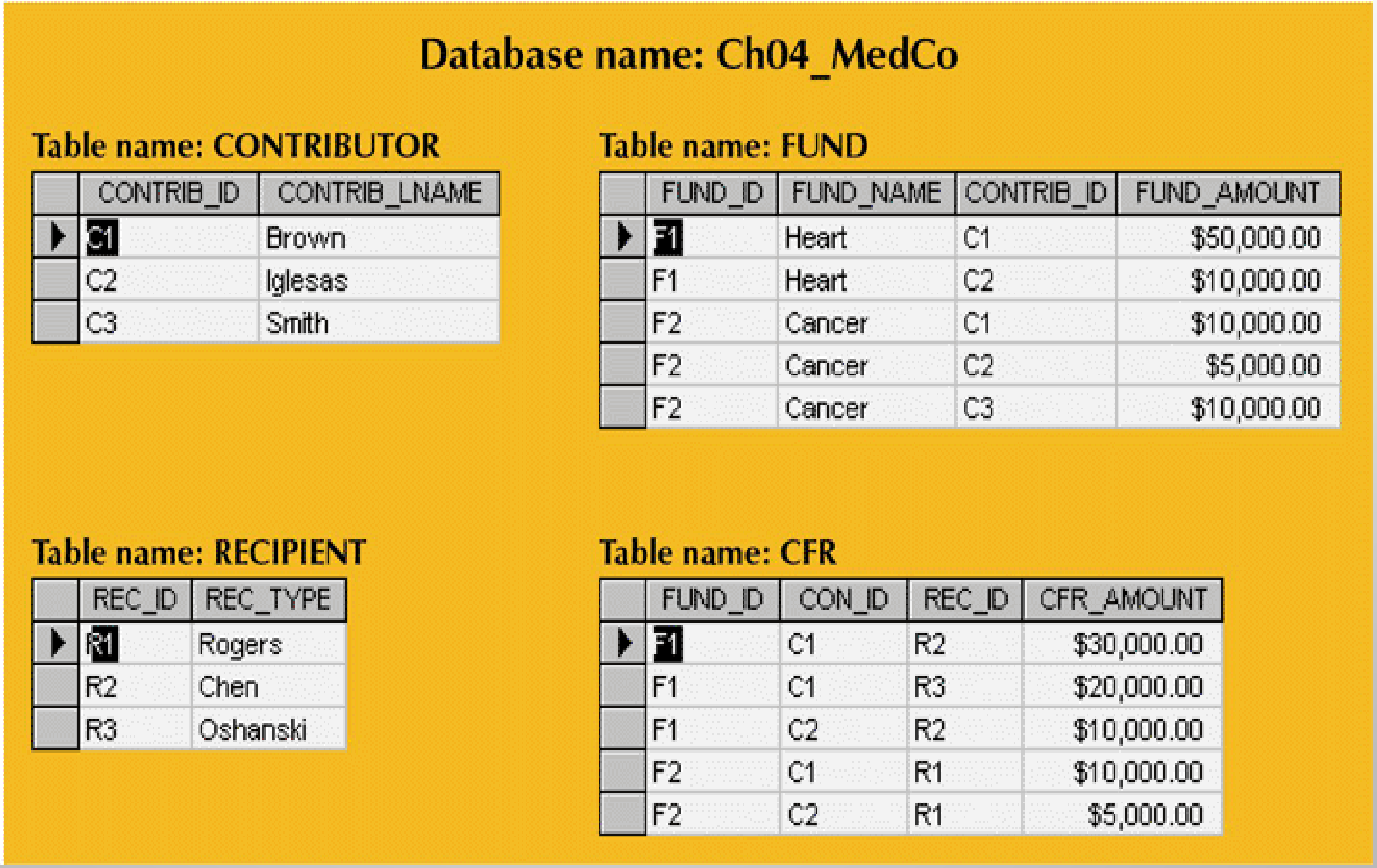
Three types of relationships

FIGURE 4.16 THREE TYPES OF RELATIONSHIPS



The implementation of a ternary relationship

FIGURE 4.17 The Implementation of a Ternary Relationship



Database name: Ch04_MedCo

Table name: CONTRIBUTOR

CONTRIB_ID	CONTRIB_LNAME
C1	Brown
C2	Iglesas
C3	Smith

Table name: FUND

FUND_ID	FUND_NAME	CONTRIB_ID	FUND_AMOUNT
F1	Heart	C1	$50,000.00
F1	Heart	C2	$10,000.00
F2	Cancer	C1	$10,000.00
F2	Cancer	C2	$5,000.00
F2	Cancer	C3	$10,000.00

Table name: RECIPIENT

REC_ID	REC_TYPE
R1	Rogers
R2	Chen
R3	Oshanski

Table name: CFR

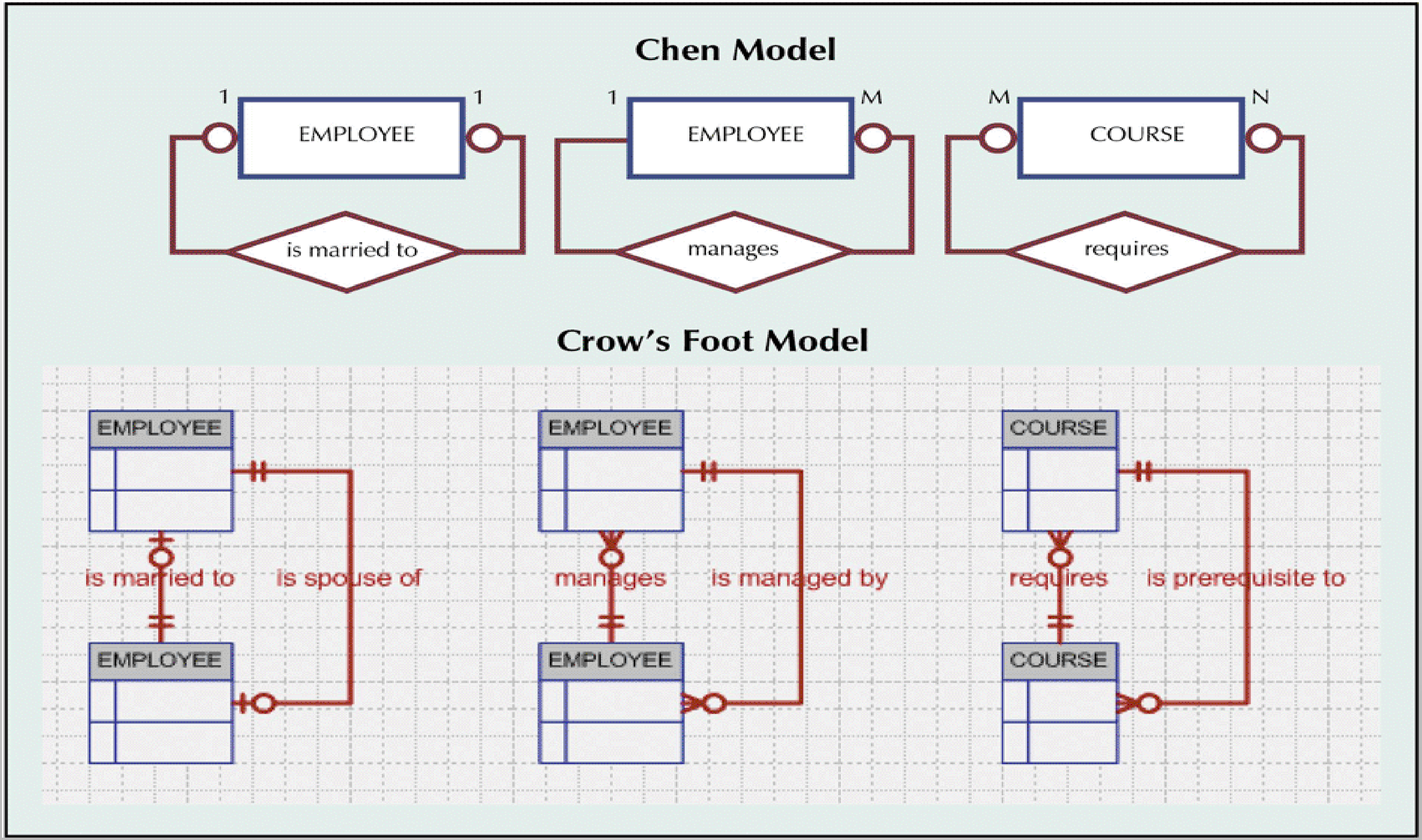
FUND_ID	CON_ID	REC_ID	CFR_AMOUNT
F1	C1	R2	$30,000.00
F1	C1	R3	$20,000.00
F1	C2	R2	$10,000.00
F2	C1	R1	$10,000.00
F2	C2	R1	$5,000.00

Role and Recursive relationships

- **Role** : the function of any entity which it plays in relationship set is called that entity's role. *E.G.,* Employee plays the role of worker in his department.
- **Recursive relationship set** : when the same entity sets participate in same relationship set more than once with different roles each time, then this type of recursive relationship set is known as recursive relationship set. *E.G.,* Consider an example of relationship set works_in and two entity set student and college. A student who attends weekend classes in college as student may also be lecturer in that college. Then this person plays two roles (student, faculty) in same relationship set work_in.

An ER representation of recursive relationships

FIGURE 4.18 An ER Representation of Recursive Relationships



Mapping Constraints

- There are certain constraints in E-R model. Data in the database must follow the constraints.
- Constraints act as rules to which the contents of database must conform.
- There are *two types* of mapping constraints : (*a*) *mapping cardinalities*, (*b*) *participation constraints.*

Connectivity and cardinality

- Connectivity :Used to describe the relationship classification, values of are “one” or “many”.
- Cardinality : Expresses the specific number of entity occurrences associated with one occurrence of the related entity
- Cardinality is indicated by placing the appropriate numbers beside the entities, using the format (x,y).
- The first value represents the minimum number of associated entities, while the second value represents the maximum number of associated entities.
- Established by very concise statements known as *business rules*

Degree of Relationship Sets

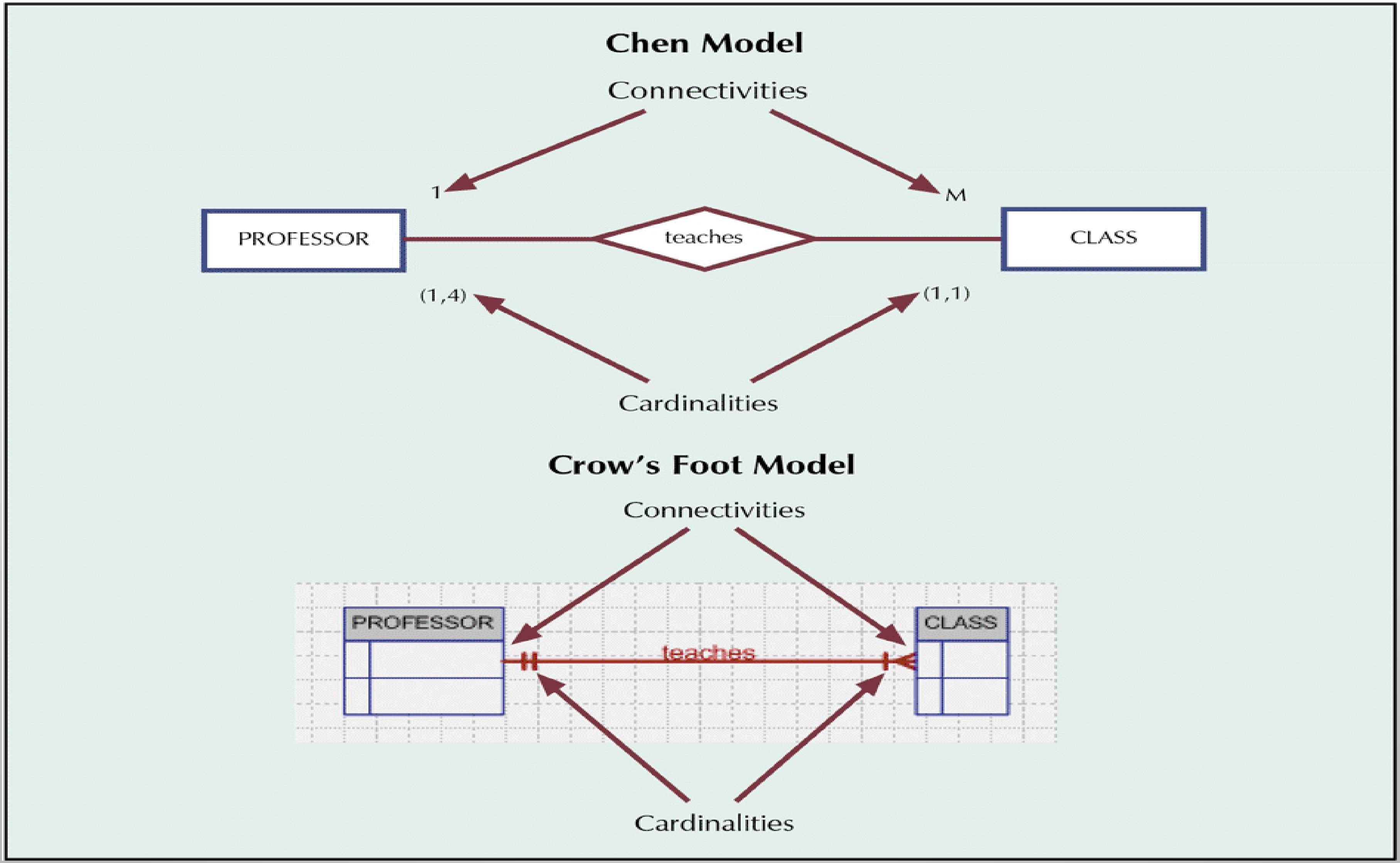
- One-to-many (1:M) relationship: e.g. a customer (the "one") may generate many invoices, but each invoice (the "many") is generated by only a single customer. The "CUSTOMER generates INVOICE" relationship would also be labeled 1:M.
- Many-to-many (M:N or M:M) relationship: e.g. a student can take many classes and each class can be taken by many students, thus yielding the M:N label for the relationship expressed by "STUDENT takes CLASS."

Degree of Relationship Sets

- One-to-one (1:1) relationship: e.g. retail company’s management structure may require that each of its stores be managed by a single employee. In turn, each store manager, who is an employee, manages only a single store. Therefore, the relationship “EMPLOYEE manages STORE” is labeled 1:1.
- Constraint: a restriction placed on the data help to ensure data integrity, expressed in the form of rules

CONNECTIVITY AND CARDINALITY IN AN ERD

FIGURE 4.7 Connectivity and Cardinality in an ERD



The 1:1 recursive relationship “EMPLOYEE is married to EMPLOYEE”

FIGURE 4.19 THE 1:1 RECURSIVE RELATIONSHIP “EMPLOYEE IS MARRIED TO EMPLOYEE”

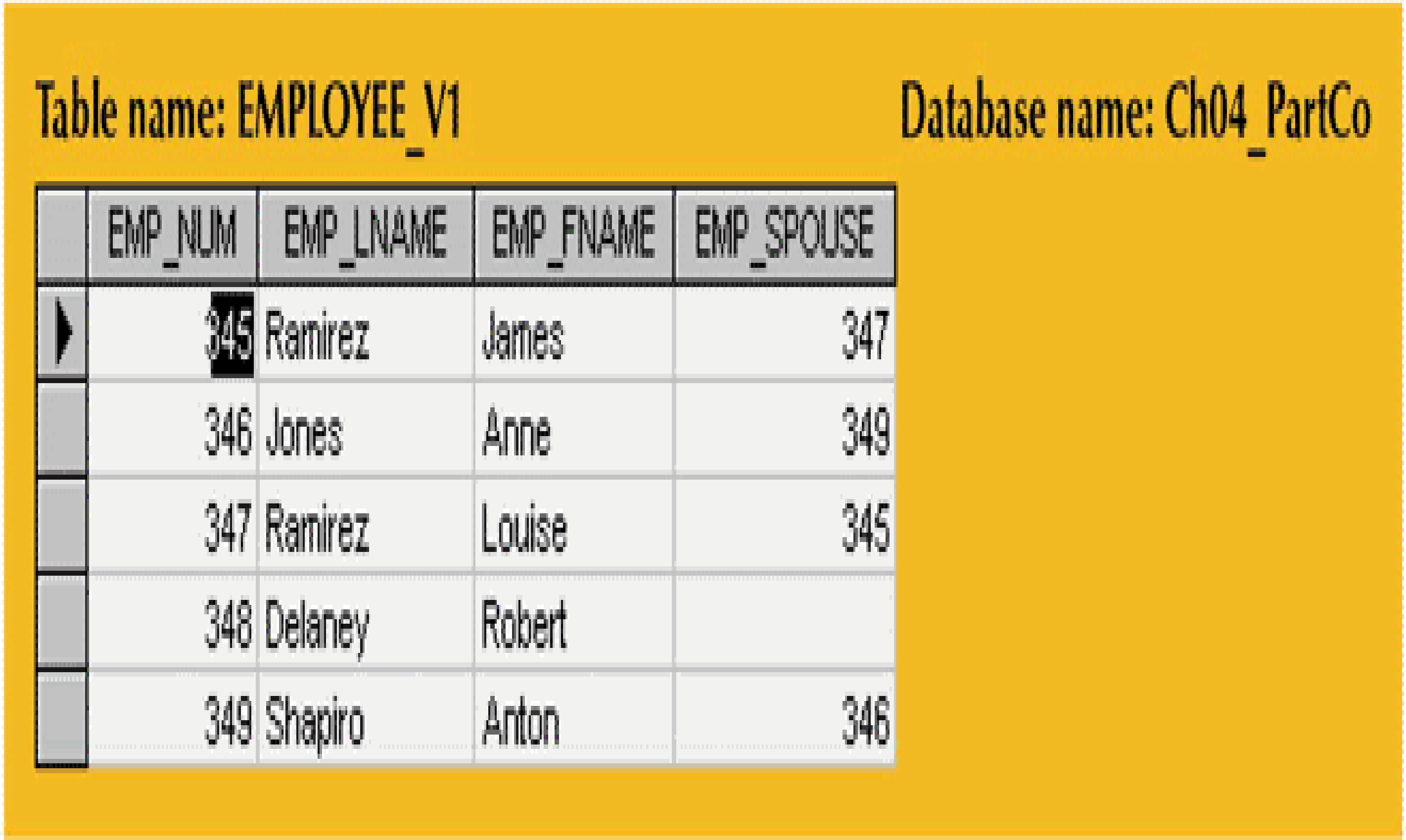


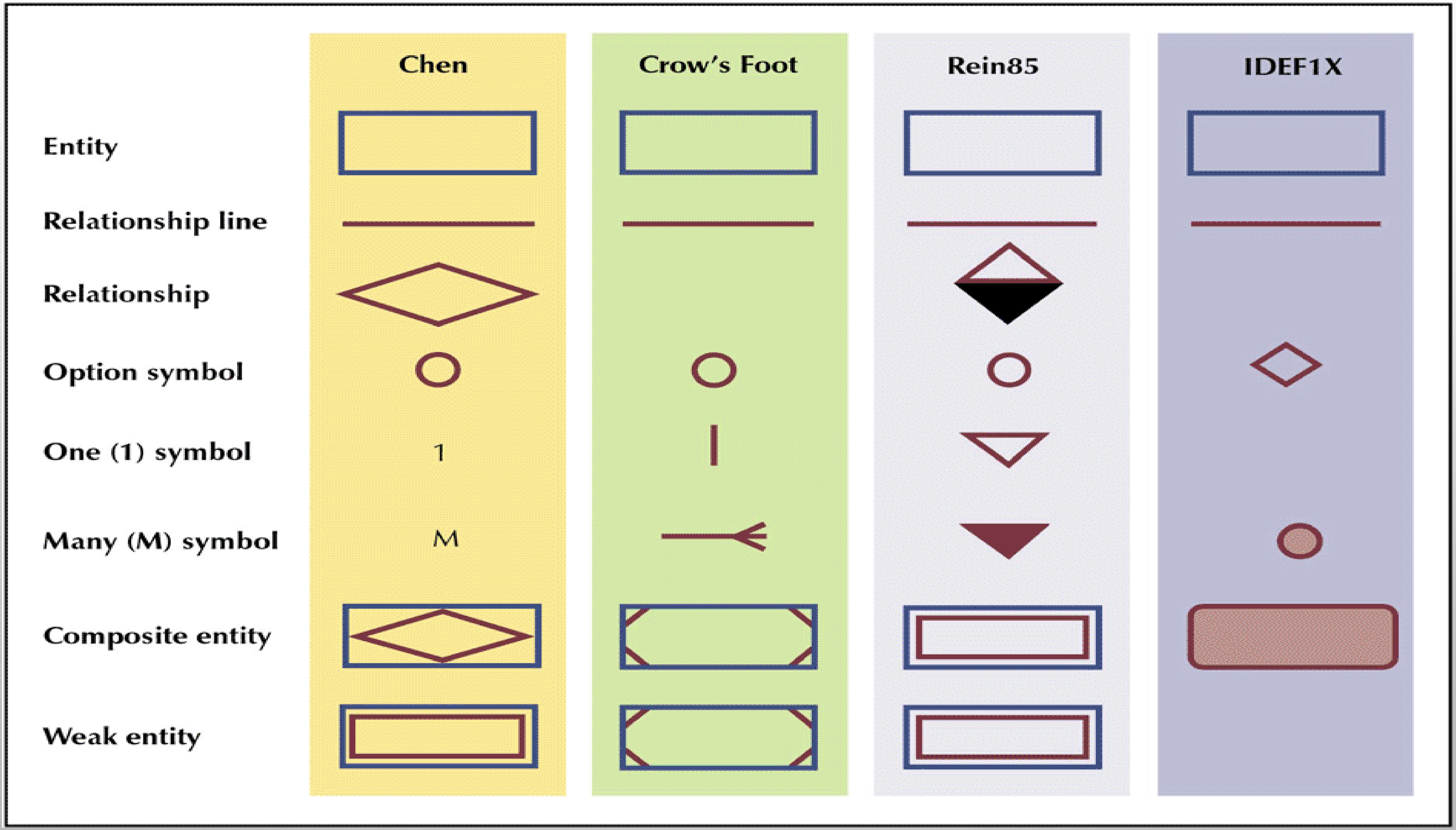
Table name: EMPLOYEE_V1

Database name: Ch04_PartCo

EMP_NUM	EMP_LNAME	EMP_FNAME	EMP_SPOUSE
345	Ramirez	James	347
346	Jones	Anne	349
347	Ramirez	Louise	345
348	Delaney	Robert	
349	Shapiro	Anton	346

A comparison of ER modeling symbols

FIGURE 4.31 A COMPARISON OF ER MODELING SYMBOLS



Types of Entity Sets

- Entity set having any key attributes are known as strong entity sets.
- Entity sets having no key attributes are known as weak entity sets.

Relationship Strength

- The concept of relationship strength is based on how the primary key of a related entity is defined.
- Weak (non-identifying) relationships
    - One entity is not existence-independent on another entity
        - The PK of the related entity does not contain a PK component of the parent entity.
        - The CLASS PK did not inherit the PK component from the COURSE entit

Course(**<u>crs_code</u>**, dept_code, crs_desc,crs_credit)

Class(**<u>class_code</u>**, crs-code,class_section,…)

A weak (non-identifying) relationship between COURSE and CLASS

FIGURE 4.8 A Weak (Non-Identifying) Relationship Between COURSE and CLASS

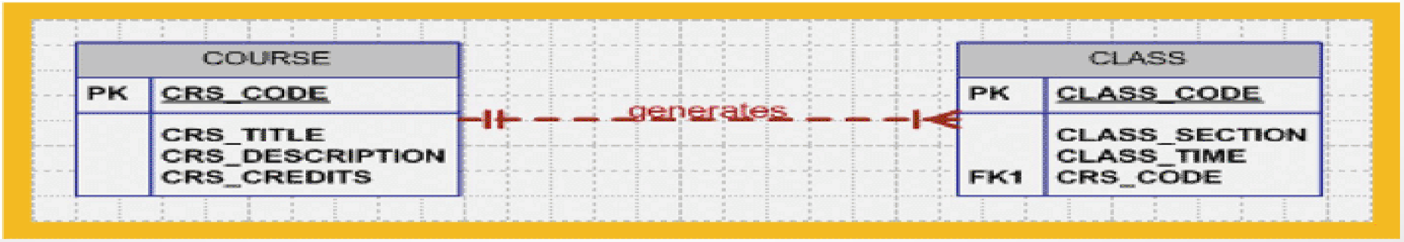


FIGURE 4.9 A Weak Relationship Between COURSE and CLASS

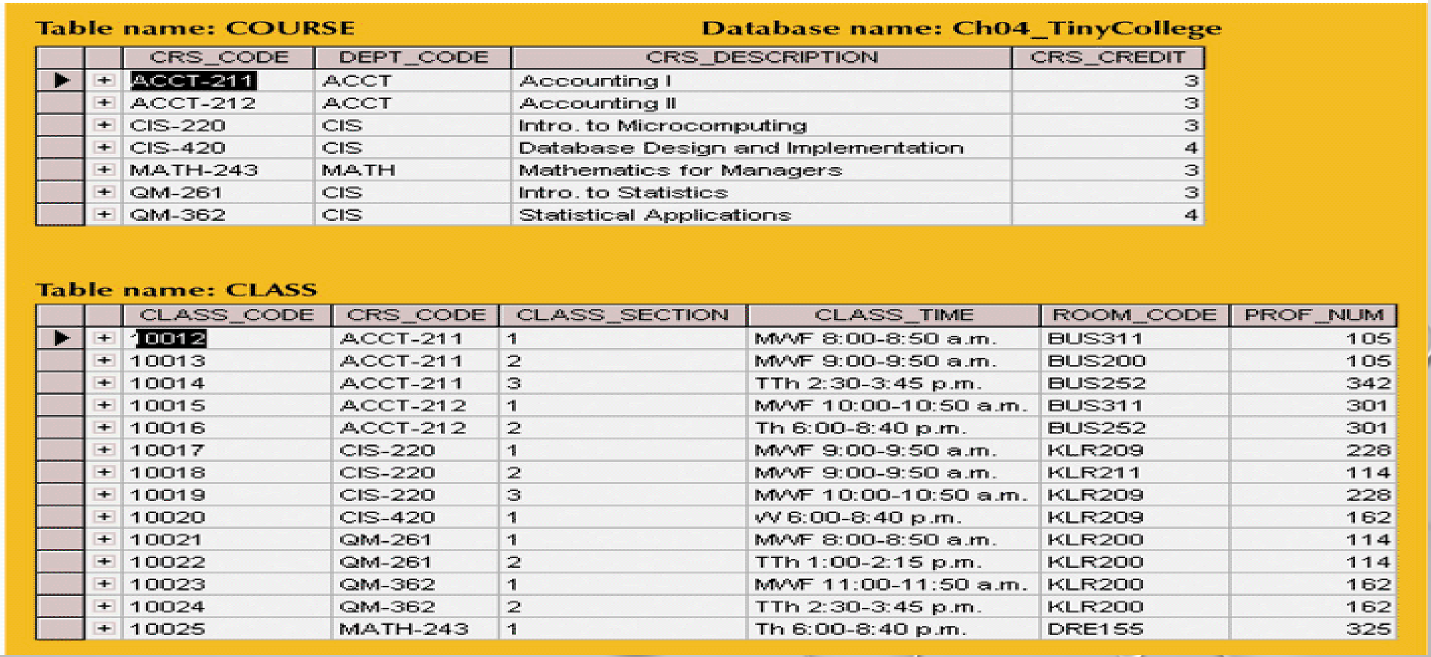


Table name: COURSE

Database name: Ch04_TinyCollege

CRS_CODE	DEPT_CODE	CRS_DESCRIPTION	CRS_CREDIT
ACCT-211	ACCT	Accounting I	3
ACCT-212	ACCT	Accounting II	3
CIS-220	CIS	Intro. to Microcomputing	3
CIS-420	CIS	Database Design and Implementation	4
MATH-243	MATH	Mathematics for Managers	3
QM-261	CIS	Intro. to Statistics	3
QM-362	CIS	Statistical Applications	4

Table name: CLASS

CLASS_CODE	CRS_CODE	CLASS_SECTION	CLASS_TIME	ROOM_CODE	PROF_NUM
10012	ACCT-211	1	MWF 8:00-8:50 a.m.	BUS311	105
10013	ACCT-211	2	MWF 9:00-9:50 a.m.	BUS200	105
10014	ACCT-211	3	TTh 2:30-3:45 p.m.	BUS252	342
10015	ACCT-212	1	MWF 10:00-10:50 a.m.	BUS311	301
10016	ACCT-212	2	Th 6:00-8:40 p.m.	BUS252	301
10017	CIS-220	1	MWF 9:00-9:50 a.m.	KLR209	228
10018	CIS-220	2	MWF 9:00-9:50 a.m.	KLR211	114
10019	CIS-220	3	MWF 10:00-10:50 a.m.	KLR209	228
10020	CIS-420	1	W 6:00-8:40 p.m.	KLR209	162
10021	QM-261	1	MWF 8:00-8:50 a.m.	KLR200	114
10022	QM-261	2	TTh 1:00-2:15 p.m.	KLR200	114
10023	QM-362	1	MWF 11:00-11:50 a.m.	KLR200	162
10024	QM-362	2	TTh 2:30-3:45 p.m.	KLR200	162
10025	MATH-243	1	Th 6:00-8:40 p.m.	DRE155	325

Relationship strength

- Strong (identifying) relationships
  - Related entities are existence-dependent
  - Whenever the PK of the related entity contains a PK component of the parent entity

Course(**crs_code**, dept_code, crs_desc,crs_credit)

Class(**crs_code,class_code**, crs-code, class_section,…)

A strong (identifying) relationship between COURSE and CLASS

FIGURE 4.10 A Strong (Identifying) Relationship Between COURSE and CLASS

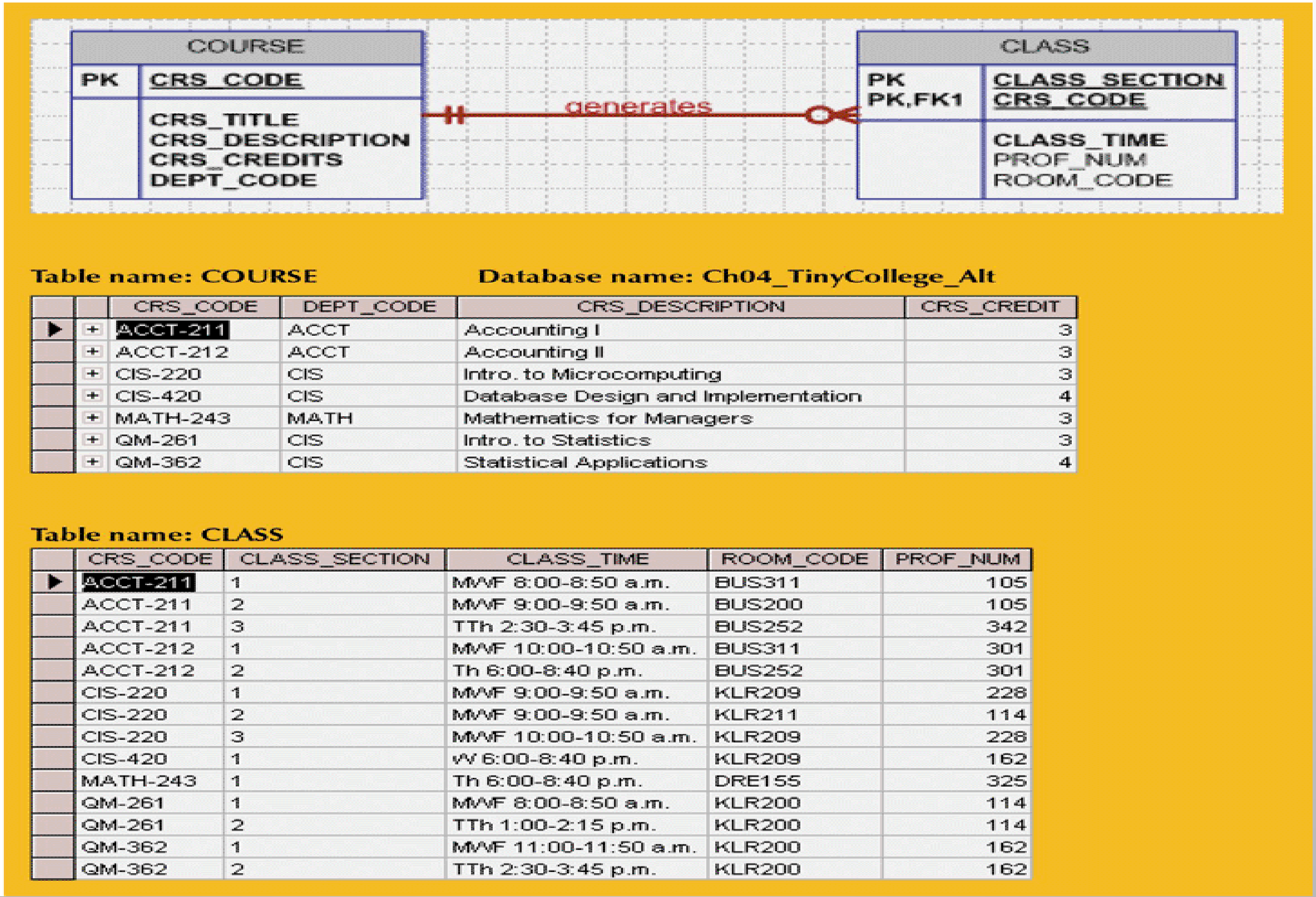


Table name: COURSE

Database name: Ch04_TinyCollege_Alt

CRS_CODE	DEPT_CODE	CRS_DESCRIPTION	CRS_CREDIT
ACCT-211	ACCT	Accounting I	3
ACCT-212	ACCT	Accounting II	3
CIS-220	CIS	Intro. to Microcomputing	3
CIS-420	CIS	Database Design and Implementation	4
MATH-243	MATH	Mathematics for Managers	3
QM-261	CIS	Intro. to Statistics	3
QM-362	CIS	Statistical Applications	4

Table name: CLASS

CRS_CODE	CLASS_SECTION	CLASS_TIME	ROOM_CODE	PROF_NUM
ACCT-211	1	MWF 8:00-8:50 a.m.	BUS311	105
ACCT-211	2	MWF 9:00-9:50 a.m.	BUS200	105
ACCT-211	3	TTh 2:30-3:45 p.m.	BUS252	342
ACCT-212	1	MWF 10:00-10:50 a.m.	BUS311	301
ACCT-212	2	Th 6:00-8:40 p.m.	BUS252	301
CIS-220	1	MWF 9:00-9:50 a.m.	KLR209	228
CIS-220	2	MWF 9:00-9:50 a.m.	KLR211	114
CIS-220	3	MWF 10:00-10:50 a.m.	KLR209	228
CIS-420	1	W 6:00-8:40 p.m.	KLR209	162
MATH-243	1	Th 6:00-8:40 p.m.	DRE155	325
QM-261	1	MWF 8:00-8:50 a.m.	KLR200	114
QM-261	2	TTh 1:00-2:15 p.m.	KLR200	114
QM-362	1	MWF 11:00-11:50 a.m.	KLR200	162
QM-362	2	TTh 2:30-3:45 p.m.	KLR200	162