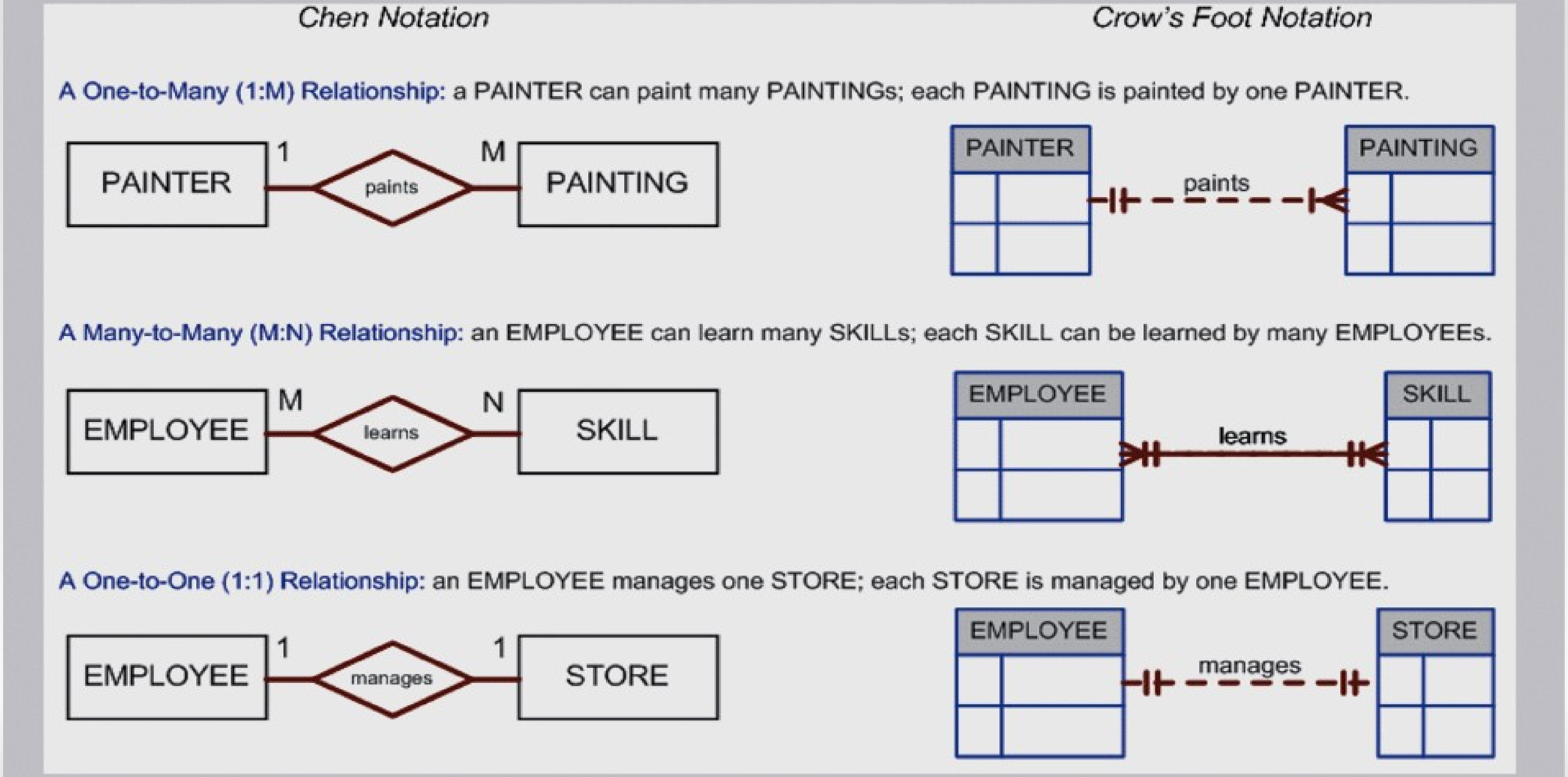
Entity Relationship (ER) Modeling

ERDs

- A diagram of the end-user view of a DB
- Primary goal is to model
  - Attributes
  - Relationships
  - Entities (duh)

They look like this!

FIGURE 2.5 The Chen and Crow's Foot notations



A One-to-Many (1:M) Relationship: a PAINTER can paint many PAINTINGs; each PAINTING is painted by one PAINTER.

A Many-to-Many (M:N) Relationship: an EMPLOYEE can learn many SKILLs; each SKILL can be learned by many EMPLOYEEs.

A One-to-One (1:1) Relationship: an EMPLOYEE manages one STORE; each STORE is managed by one EMPLOYEE.

The Types

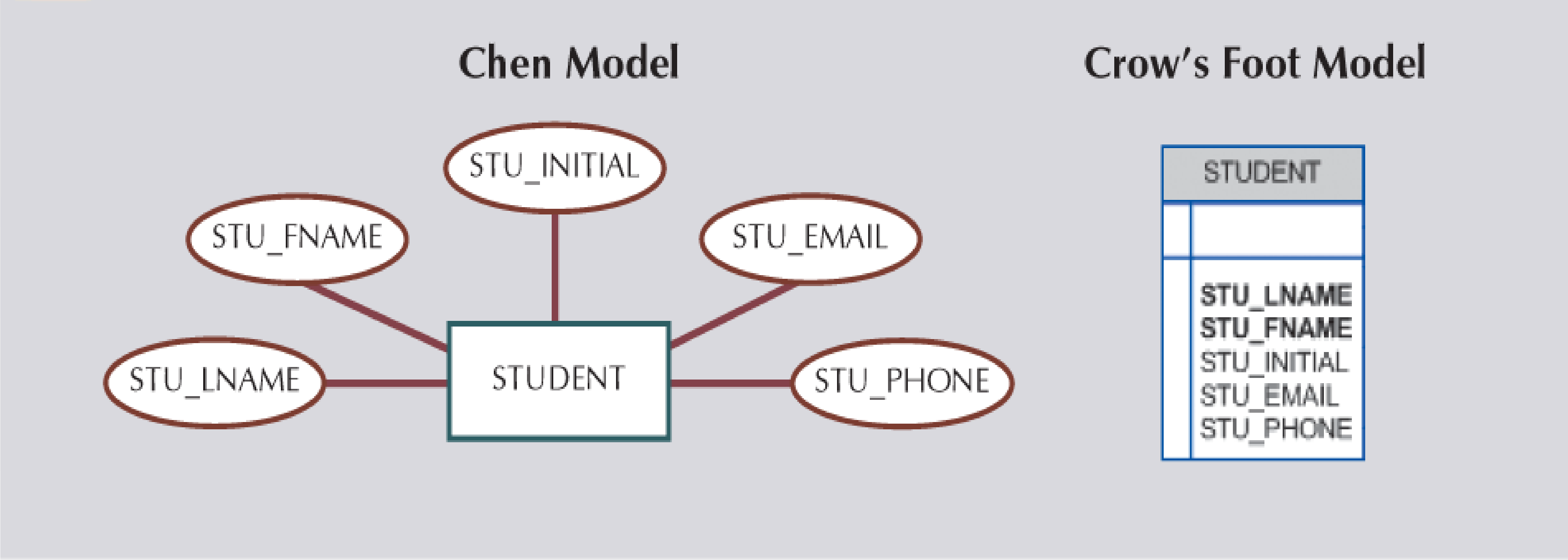
- Chen
  - Emphasis on *modeling*
- Crow's Foot
- UML
  - Both of these focus more on *design and implementation*
- In reality you can implement your DB from either

Review

- Entities → corresponds real-world objects
  - *Entity* and *object* are used interchangeably
  - Entity Set → collection of entities
  - Entity Instance or Occurrence → a particular entity, basically a row in a table
  - Represented by a named rectangle in ERDs
- Attributes → entity characteristics
  - Connected ovals in Chen and listed in the rectangle in Crow's
  - Can be required or optional (but optional can be bad if implemented wrong!)

Attributes Examples

FIGURE 4.1 The attributes of the STUDENT entity



Review

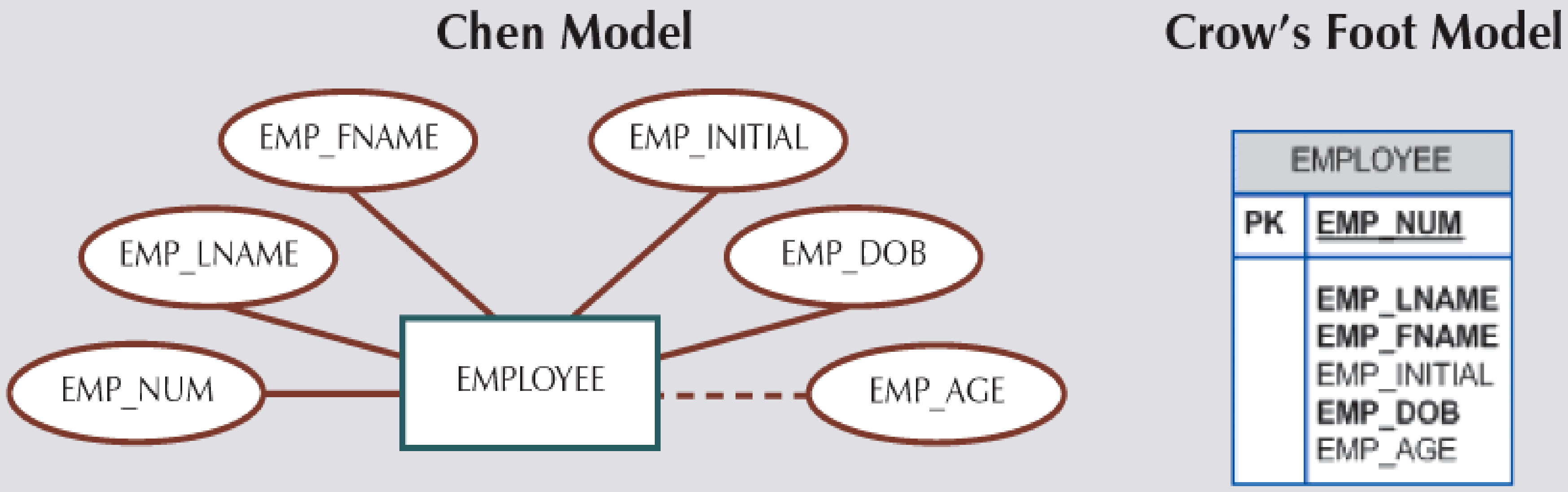
- Attributes
  - Have domains: rules about what values are valid
  - Can be **simple** or **composite** (composite can be a bit sticky as to whether they should be broken down)
- **Identifiers** (aka primary keys)
  - Can be based on a single attribute or multiple (**composite**)
  - When using composites you must be careful about making sure they will be unique

More on Attributes

- Can be single- or multi-valued
  - Be careful, there is not necessarily a relationship between single-valued and being a simple attribute
  - We don't implement multi-valued though! (pg 109)
- **Derived Attributes** → values that can be calculated or reasoned from other attributes
  - These may or may not be stored in the database
  - Why?

Derived Attributes in ERDs

FIGURE 4.6 Depiction of a derived attribute



Chen Model

Crow's Foot Model

EMPLOYEE	
PK	EMP_NUM
	EMP_LNAME EMP_FNAME EMP_INITIAL EMP_DOB EMP_AGE

Stored vs. Calculated

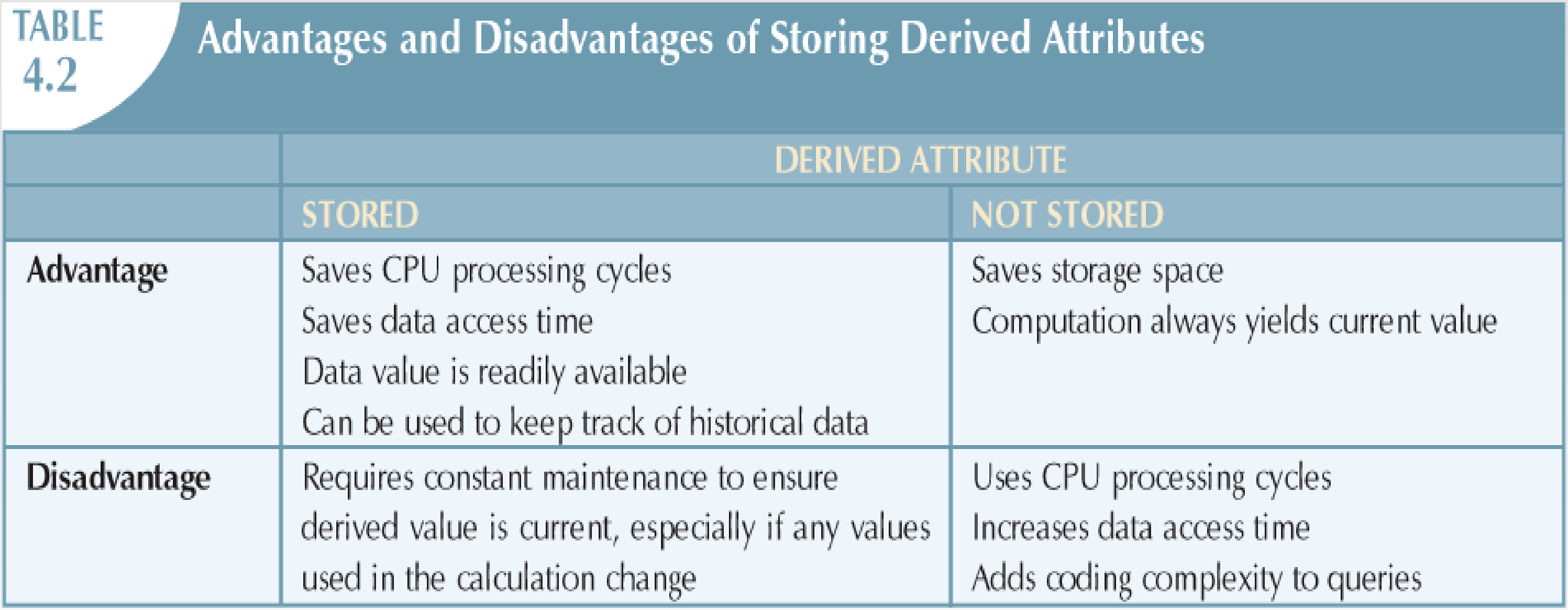
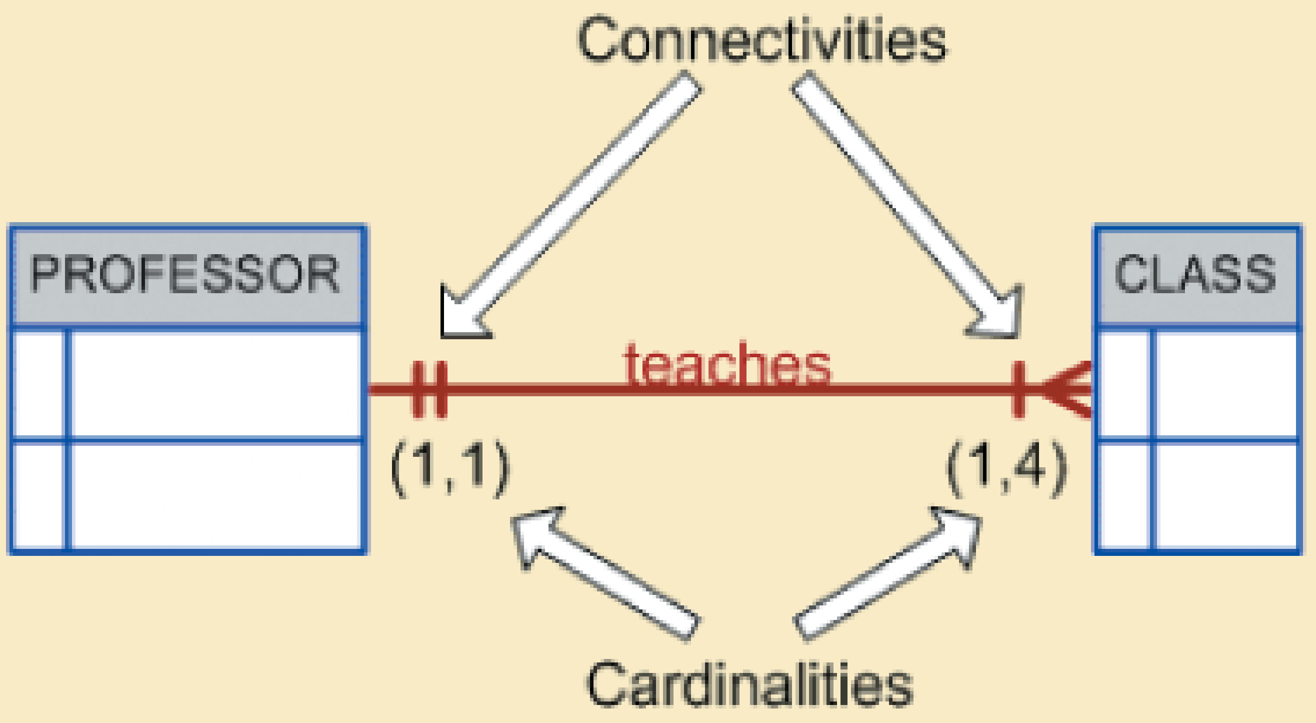


TABLE 4.2 Advantages and Disadvantages of Storing Derived Attributes

	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

Connectivity and Cardinality

FIGURE 4.7 Connectivity and cardinality in an ERD



Existence Dependence

- An entity is **existence dependent** when its only reason for being in the database is to be associated with another entity
  - It has a mandatory foreign key attribute, meaning it cannot be null
  - In terms of the college database, an Emergency Contact entity is existence dependent upon a Student
- Something may also be **existence independent**
  - An example might be in-state transfer credits

Relationship Strength

- **Weak** or **Non-identifying** → the FK and *only* the FK of the related table is the PK of another
  - Sales had the Agent's ID (the Agent PK) as a FK
  - Pretty much the standard type of FK relationship we've been talking about in our examples so far
  - Drawn as a dashed line in a Crow's Foot ERD
- **Strong** or **Identifying** → the FK of the related table is also involved in its PK, along with being the PK of another table (solid line in Crow's)
  - If Sales no longer has a numeric PK but a composite of sales date, agent ID, and VIN then the relationship becomes **strong**

Gotcha

- When creating tables, you must create the 1 side of a 1:M relationship first... why?
- Also note that the way a relationship is determined is by looking at the table that contains the foreign key (the ***related table***)

Entities can be weak too!

- If a related entity is both existence-dependent and has a strong relationship with its parent, it is a **weak entity**
  - In Crow's foot these are represented by the combination of the solid line of a strong relationship and the the PK/FK designation
- These are not very common and are up to the database designer

Weak Entity Example

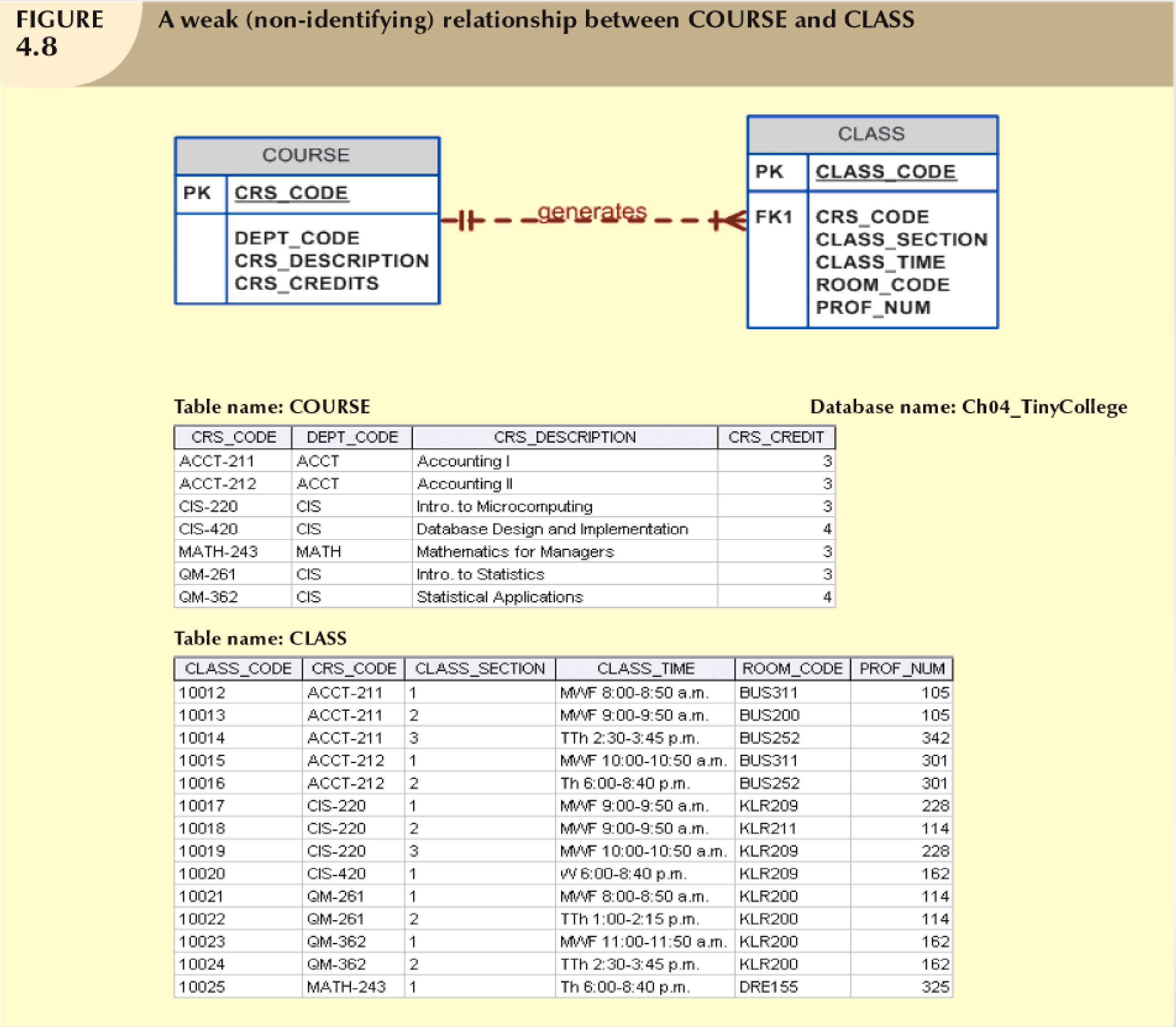


FIGURE 4.8 A weak (non-identifying) relationship between COURSE and CLASS

COURSE
PK CRS_CODE
DEPT_CODE
CRS_DESCRIPTION
CRS_CREDITS

generates

CLASS
PK CLASS_CODE
FK1 CRS_CODE
CLASS_SECTION
CLASS_TIME
ROOM_CODE
PROF_NUM

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

Participation

- Entity relationships can be of two types in relation to whether there is a related entity for every parent
- To determine relationship participation you have to look at it both ways and determine its categorization *for each direction*
- The easiest way to determine these is to examine cardinality

Participation Types

- **Optional** → When the 'parent' does not require a 'child' to be present
  - In Crow's Foot a circle is added to the related entity's end
- **Mandatory** → There is an entity required on each end for the relationship to make sense
  - In Crow's Foot no circle is assumed mandatory
  - The cardinality can also help, there can be no 0 in a mandatory relationship

How to 'read' the diagram

- Fig. 4.12
- "Professor teaches 0 to 3 classes" is the relationship
  - To get the partcipation you read backwards: "Class is ***optional*** to professor"
- "A class is taught by 1 professor" is the relationship
  - "Professor is ***mandatory*** to class"

Crow's Foot Diagrams

FIGURE 4.13 CLASS is optional to COURSE

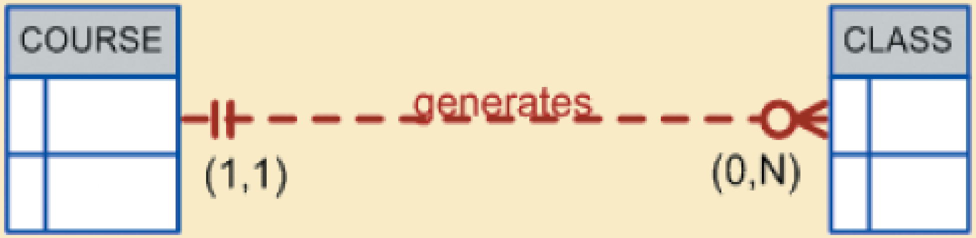
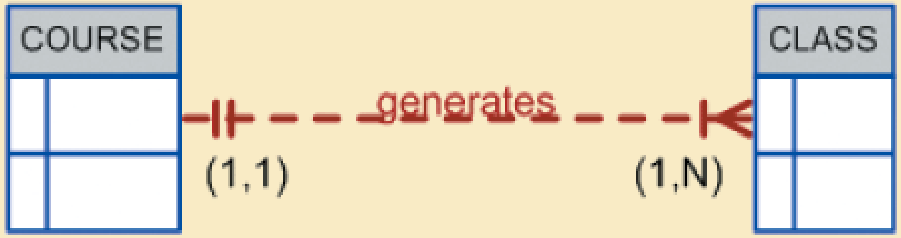


FIGURE 4.14 COURSE and CLASS in a mandatory relationship



Gotcha

- Participation and Strength do not determine each other
- A mandatory relationship does not imply a strong relationship (or vice versa) nor the same for optional/weak
- Relationship *strength* is determined by the composition of the related table's PK
- Relationship *participation* are based on business rules (which may contradict)
- See the previous diagrams

Things to be careful about

- Relationships help determine the order in which you create tables and their rows
  - If you create a class entity with no associated course, a temporary course would have to be created until the official course is approved
- Make sure you understand the semantics of the relationship description!
  - For our examples, keeping course and class straight is important as well as getting the cardinality correct

Crow's Feet Clarification

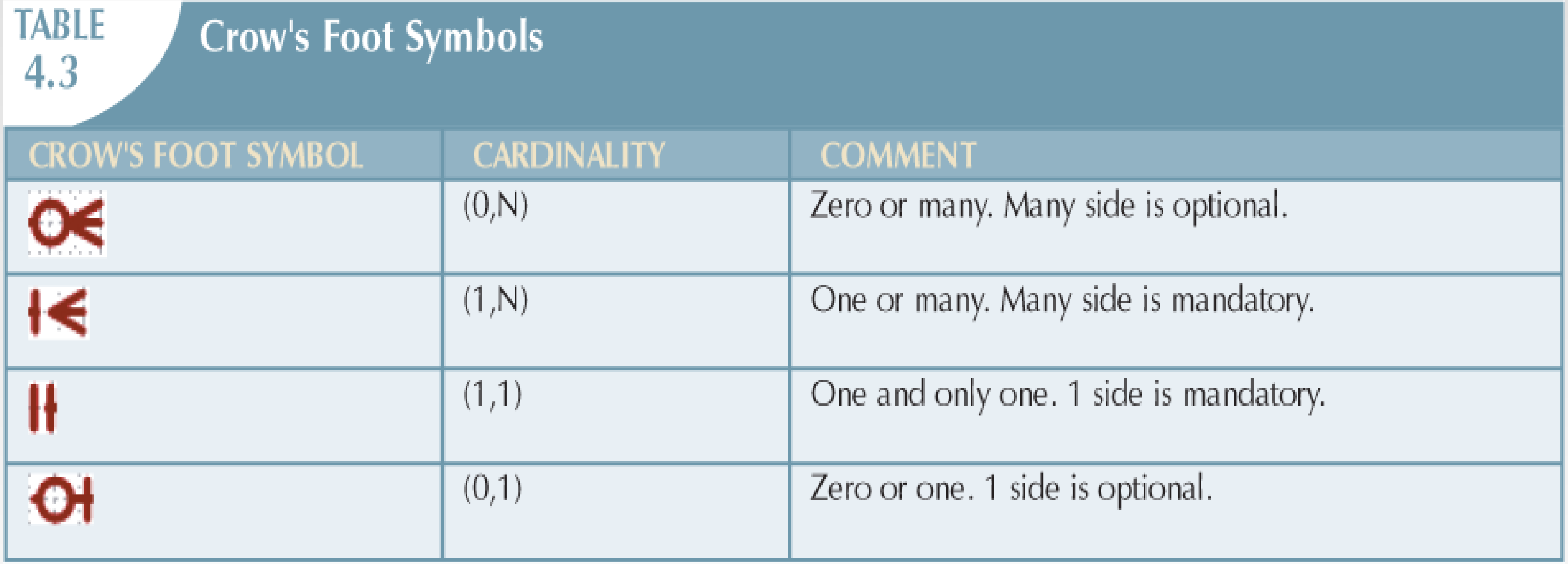


TABLE 4.3 Crow's Foot Symbols

CROW'S FOOT SYMBOL	CARDINALITY	COMMENT
○<	(0,N)	Zero or many. Many side is optional.
ǀ<	(1,N)	One or many. Many side is mandatory.
ǁ	(1,1)	One and only one. 1 side is mandatory.
○ǀ	(0,1)	Zero or one. 1 side is optional.

Associative Entities

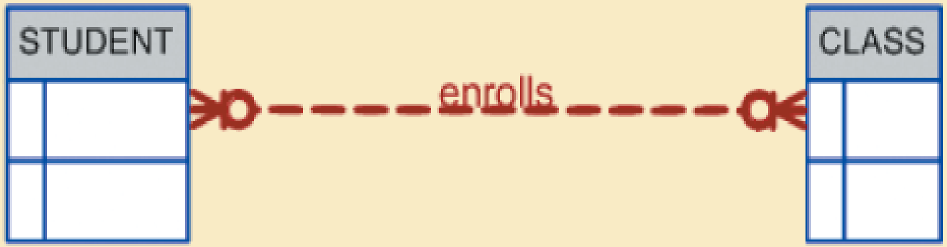
- First heard about these when talking about M:N relationships and how a typical DB model does not allow them
- To overcome the problem, an associative entity is create as a bridge between the two
- The idea is to turn the M:N into two 1:M relationships
- The associative entity will have the PKs from each table as FKs and its own PK
- No special ERD notation, look at the keys and strong/identifying relationships

ERD for Student-Class

- What would it be?
- Think about the relationship between the two, we won't worry about cardinality
  - Optional or mandatory?

The ERD

FIGURE 4.24 The M:N relationship between STUDENT and CLASS



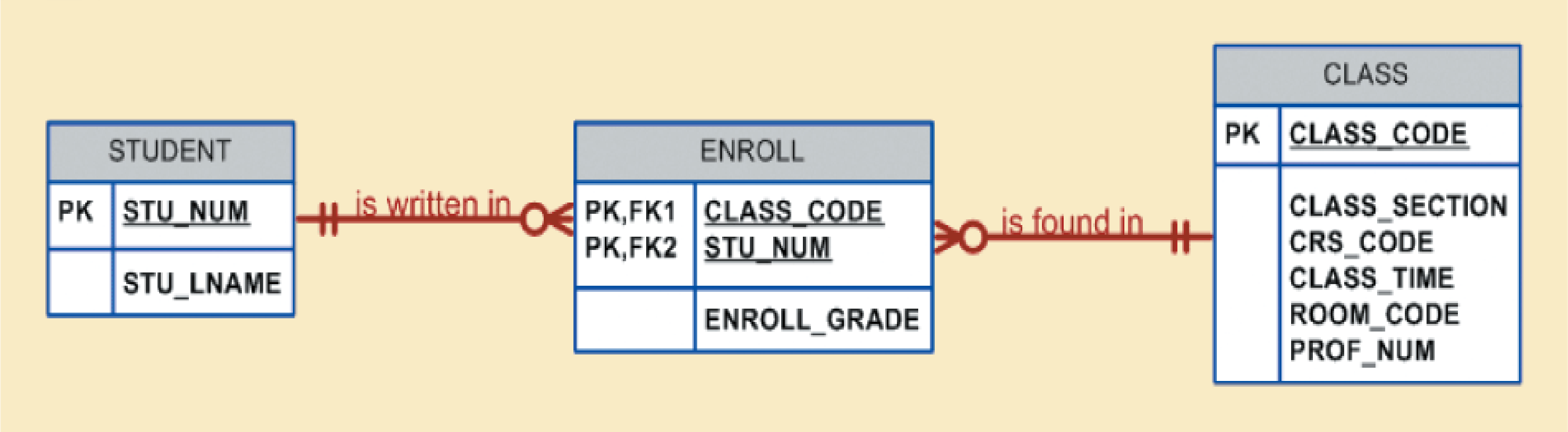
Visio does not permit the definition of a M:N relationship. To make this illustration, two 1:M relationships have been superimposed.

Making the associative entity

- Things to think about:
  - How the relationships change; the optionalities get moved to the new relationships
  - What are the keys?
  - What else needs to be in the new table?

The new ERD

FIGURE 4.25 A composite entity in an ERD



STUDENT	
PK	STU_NUM
	STU_LNAME

ENROLL	
PK,FK1 PK,FK2	CLASS_CODE STU_NUM
	ENROLL_GRADE

CLASS	
PK	CLASS_CODE
	CLASS_SECTION CRS_CODE CLASS_TIME ROOM_CODE PROF_NUM

Developing an ER Diagram

Section 4.2 – Pg 127
A highly recommended read

Homework!

- Review Questions → 2, 3, 4, 5, 8, 9, 10, 15
- Problems → 1, 2, 5, 7