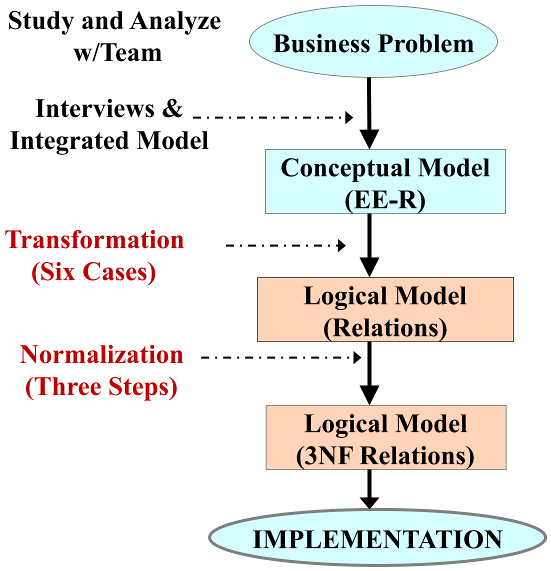
Chapter 4
Logical Database Design and the Relational Model

Objectives

- Define terms for the relational data model
- Transform EE-R diagrams to relations
- Create tables with entity and relational integrity constraints

Steps in Database Problem Solving



Study and Analyze w/Team

Business Problem

Interviews & Integrated Model

Conceptual Model (EE-R)

Transformation (Six Cases)

Logical Model (Relations)

Normalization (Three Steps)

Logical Model (3NF Relations)

IMPLEMENTATION

Logical Model: Relational Model

- Can represent all kinds of information
- Based on Math (relations)
- Natural to people
- Relatively simple
- We know how to implement it fast

Components of Relational Model

- Data structure
  – Tables (relations), rows, columns
- Data manipulation
  – Powerful SQL operations for retrieving and modifying data
- Data integrity
  – Mechanisms for implementing business rules that maintain integrity of manipulated data

Motivating Example

- Make a list of students in the class, keeping their ID, name and phone number

Motivating Example

- Make a list of students in the class, keeping their ID, name and phone number
- You'd probably come up with something like this:

ID	Name	Phone
xx	Mike	111
yy	Elisa	222

Tuple
(Record, Row)

- This is the basic structure of the relational model, a table or relation

Extra Assumptions

- You would not repeat the same row twice
- No two rows have the same ID, but they may have the same name and phone number

ID	Name	Phone
xx	Mike	111
yy	Elisa	222

SET
(no duplicates)

- ID would be the *PRIMARY KEY (PK).*

Now add emails … (many!)

- Now you need to add the emails of each student, but you do not know how many emails
- Can you come up with a solution? Try it …

Many Fields

- Could come up with something like this

ID	Name	Phone	Email1	Email2
xx	Mike	111	bad	idea ☺
yy	Elisa	222	bad	idea ☺

- Above would not work very well. How many fields?
  - Wasted space
  - What if a student has more emails?
  - How to access the emails?

Un-Normalized

- Could also try this:

ID	Name	Phone	Email
xx	Mike	111	mk@ad.com
xx	Mike	111	mk@vu.edu
yy	Elisa	222	eli@vu.edu

- Problem is duplication, we are repeating the name and phone number in the second row
  - What if Mike changes his phone?

- Later we will study normalization to solve this.

Now add emails … (many!)

- A much better way:

Student

ID	Name	Phone
xx	Mike	111
yy	Elisa	222

Email

StudentID	Email
xx	mk@ad.com
xx	mk@vu.edu
yy	eli@vu.edu

- Every StudentID on the second table needs a matching ID on the first table: StudentID is a *FOREIGN KEY*
- In a way, StudentID in the second table is a *pointer* or *reference* to the first table

Formalizing: Relations

- Definition: A ***relation*** is a **named table** of data
  - – Table is made up of rows (records or tuples), and columns (attributes or fields)
- Requirements for a table to be a relation:
  1. Has a unique name.
  2. Every attribute value is atomic (not multivalued or composite)
  3. Every row is unique
  4. Attributes (columns) in tables have unique names
  5. The order of the columns is irrelevant
  6. The order of the rows is irrelevant

By definition, all relations are in ***1st Normal Form (1NF).***

13

Correspondence with ER Model

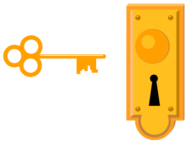
- Relations (tables) correspond to entity types and to many-to-many relationship types
- Rows correspond to entity instances and to many-to-many relationship instances
- Columns correspond to attributes

- NOTE: The word ***relation*** (in relational database) is NOT the same as the word ***relationship*** (in ER model)

14

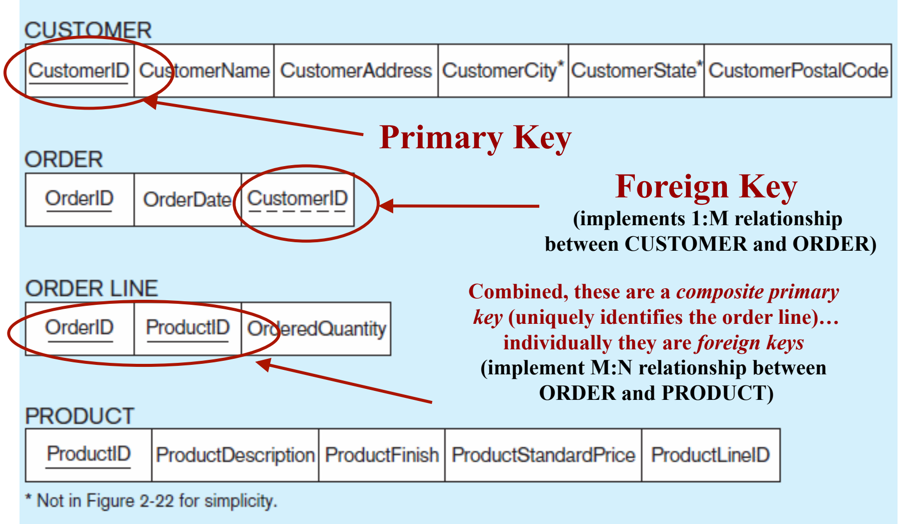
Formalizing Key Fields

- Primary key (PK)
  - Minimal set of attributes that uniquely identifies a row, chosen for referencing
  - This is how we can guarantee that all rows are unique
- Foreign key (FK)
  - Set of attributes in a table that serves as a reference to the primary key of another table



- Keys can be simple or composite
- Used as indexes to speed up queries

Figure 4-3 Schema for four relations (Pine Valley Furniture Company)



CUSTOMER

CustomerID	CustomerName	CustomerAddress	CustomerCity*	CustomerState*	CustomerPostalCode

ORDER

OrderID	OrderDate	CustomerID

ORDER LINE

OrderID	ProductID	OrderedQuantity

PRODUCT

ProductID	ProductDescription	ProductFinish	ProductStandardPrice	ProductLineID

* Not in Figure 2-22 for simplicity.

Primary Key

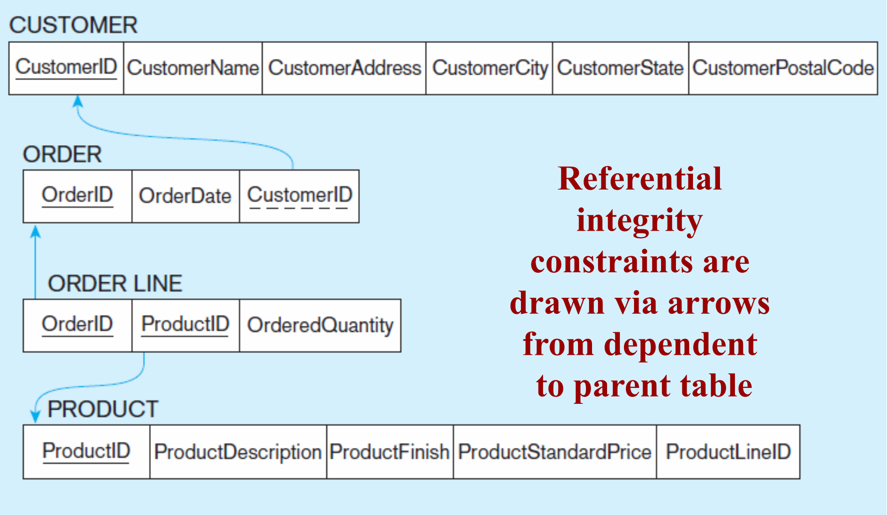
Foreign Key
(implements 1:M relationship between CUSTOMER and ORDER)

Combined, these are a *composite primary key* (uniquely identifies the order line)… individually they are *foreign keys* (implement M:N relationship between ORDER and PRODUCT)

Key Constraints

- Entity Integrity Constraint
  - No attribute of the PK may be null
- Referential Integrity Constraint
  - For a FK, either all attributes are null, or the values appear in the PK of a row of the referred table

Figure 4-5
Referential integrity constraints (Pine Valley Furniture)



CUSTOMER

CustomerID	CustomerName	CustomerAddress	CustomerCity	CustomerState	CustomerPostalCode

ORDER

OrderID	OrderDate	CustomerID

ORDER LINE

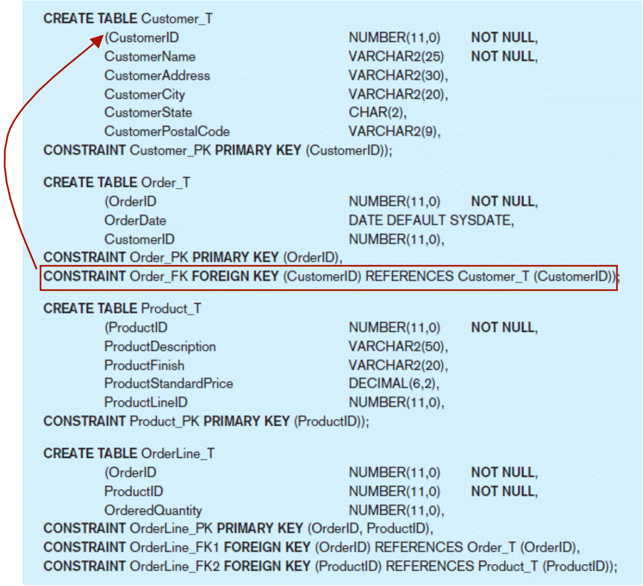
OrderID	ProductID	OrderedQuantity

PRODUCT

ProductID	ProductDescription	ProductFinish	ProductStandardPrice	ProductLineID

Referential integrity constraints are drawn via arrows from dependent to parent table

Figure 4-6 SQL table definitions



```
CREATE TABLE Customer_T
          (CustomerID                        NUMBER(11,0)     NOT NULL,
          CustomerName                       VARCHAR2(25)     NOT NULL,
          CustomerAddress                    VARCHAR2(30),
          CustomerCity                       VARCHAR2(20),
          CustomerState                      CHAR(2),
          CustomerPostalCode                 VARCHAR2(9),
CONSTRAINT Customer_PK PRIMARY KEY (CustomerID));

CREATE TABLE Order_T
          (OrderID                           NUMBER(11,0)     NOT NULL,
          OrderDate                          DATE DEFAULT SYSDATE,
          CustomerID                         NUMBER(11,0),
CONSTRAINT Order_PK PRIMARY KEY (OrderID),
CONSTRAINT Order_FK FOREIGN KEY (CustomerID) REFERENCES Customer_T (CustomerID));

CREATE TABLE Product_T
          (ProductID                         NUMBER(11,0)     NOT NULL,
          ProductDescription                 VARCHAR2(50),
          ProductFinish                      VARCHAR2(20),
          ProductStandardPrice               DECIMAL(6,2),
          ProductLineID                      NUMBER(11,0),
CONSTRAINT Product_PK PRIMARY KEY (ProductID));

CREATE TABLE OrderLine_T
          (OrderID                           NUMBER(11,0)     NOT NULL,
          ProductID                          NUMBER(11,0)     NOT NULL,
          OrderedQuantity                    NUMBER(11,0),
CONSTRAINT OrderLine_PK PRIMARY KEY (OrderID, ProductID),
CONSTRAINT OrderLine_FK1 FOREIGN KEY (OrderID) REFERENCES Order_T (OrderID),
CONSTRAINT OrderLine_FK2 FOREIGN KEY (ProductID) REFERENCES Product_T (ProductID));
```

Referential integrity constraints are implemented with foreign key to primary key references.

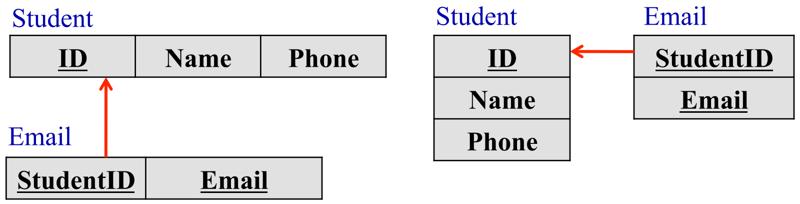
Key Constraints – Example

- Delete Rules
  - **Restrict** – don't allow delete of "parent" side if related rows exist in "dependent" side
  - **Cascade** – automatically delete "dependent" side rows that correspond with the "parent" side row to be deleted
  - **Set-to-Null** – set the foreign key in the dependent side to null if deleting from the parent side → not allowed for weak entities

From E-R Diagrams to Relations (Tables)

Transforming E-R Into Relations

- Use a rectangle for each entity (table), with attributes inside rectangles, too
  - Can be vertical or horizontal
  - Primary key is underlined
- Use arrows from Foreign key to Primary key



E-R vs. Relational

- Entities are represented by tables
  - But tables may also represent relationships, or multivalued attributes
- Foreign Keys used to relate table rows
  - Similar to relationships in E-R, but lower level
- Relational model is more concrete, lower level
  - Usually many more tables than entities
  - Harder to understand by non-technical people
  - Directly implementable

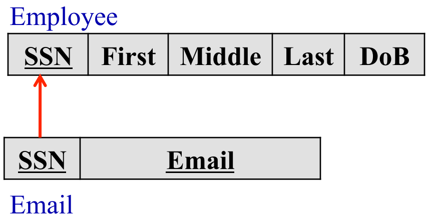
Six Cases of Transforming E-R Diagrams into Relations

1. Map Regular Entities
2. Map Binary Relationships
3. Map Weak Entities
4. Map Associative Entities
5. Map Unary Relationships
6. Map Ternary (and n-ary) Relationships

1. Mapping Regular Entities

- Create a new table for each entity
- Remember to underline the identifier
- For composite attributes, map only the basic pieces
- Derived attributes disappear
- For multivalued attributes we need a new table
- We may need to create several tables for independent multivalued attributes

EMPLOYEE
SSN
Name (First, Middle, Last)
{Emails}
Date of Birth
[Age]



Employee

SSN	**First**	**Middle**	**Last**	**DoB**

SSN	**Email**

Email

You Try …

BOOK
ISBN
Title
{Authors}
Format (Binding, NumPages, Dimensions, [Weight])

Six Cases of Transforming E-R Diagrams into Relations

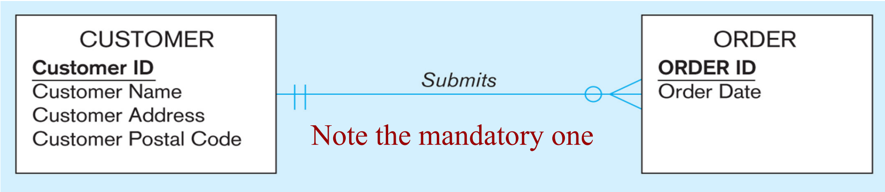
1. Map Regular Entities
2. Map Binary Relationships
3. Map Weak Entities
4. Map Associative Entities
5. Map Unary Relationships
6. Map Ternary (and n-ary) Relationships

Mapping Binary Relationships

- One-to-Many
  - Primary key on the one side becomes a foreign key on the many side (Fig. 4-12).
- One-to-One
  - Primary key on the mandatory side becomes a foreign key on the optional side (Fig. 4-14).
- Many-to-Many
  - Create a ***new relation*** with the primary keys of the two entities as its primary key (Fig. 4-13).

Fig. 4-12: Example of mapping a 1:M relationship

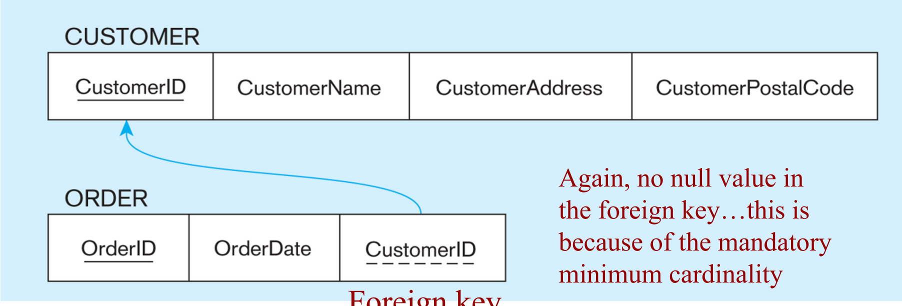
(a) Relationship between customers and orders



Note the mandatory one

Fig. 4-12: (b) Mapping the relationship

[Primary key on the one side becomes a foreign key on the many side]



CUSTOMER

<u>CustomerID</u>	CustomerName	CustomerAddress	CustomerPostalCode

ORDER

<u>OrderID</u>	OrderDate	CustomerID

Again, no null value in the foreign key…this is because of the mandatory minimum cardinality

Foreign key

You Try – 1:M Relationship

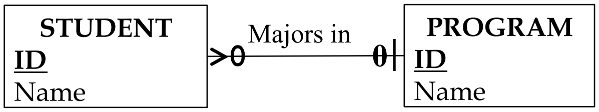
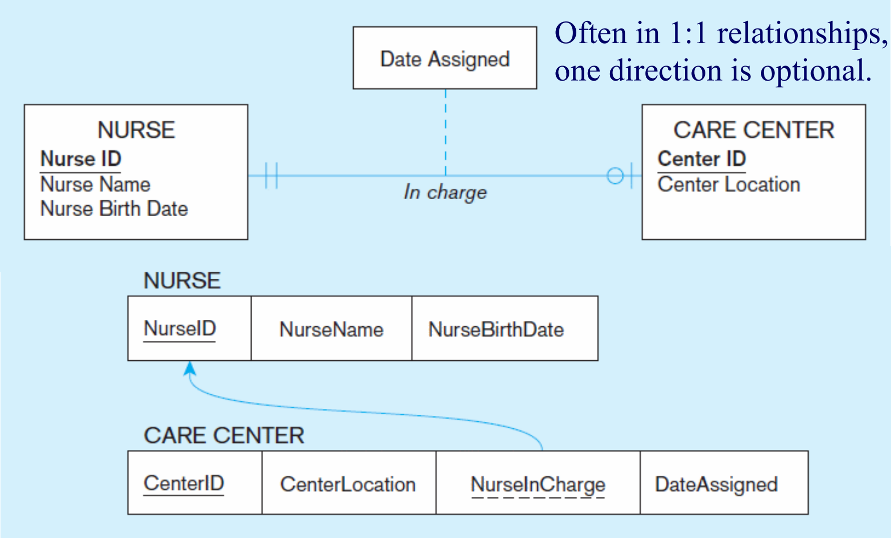


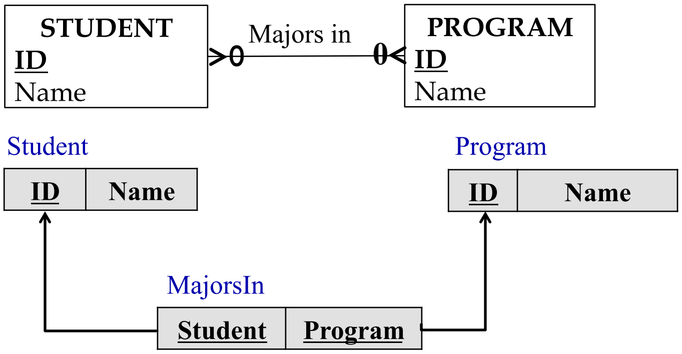
Figure 4-14 Example of mapping a binary 1:1 relationship



Often in 1:1 relationships, one direction is optional.

Foreign key goes in the relation on the optional side, matching the primary key on the mandatory side

Many-to-Many Relationship



STUDENT
ID
Name

Majors in

PROGRAM
ID
Name

Student

ID	Name

Program

ID	Name

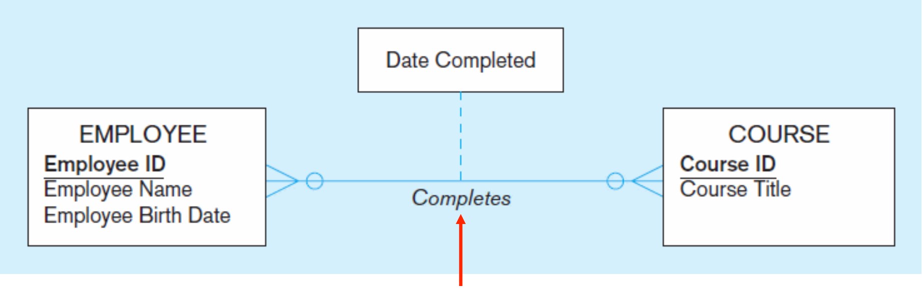
MajorsIn

Student	Program

- For a many-to-many, we need a new table representing the relationship.
- This table has Foreign Keys to both entities.

Figure 4-13 Example of mapping an M:N relationship

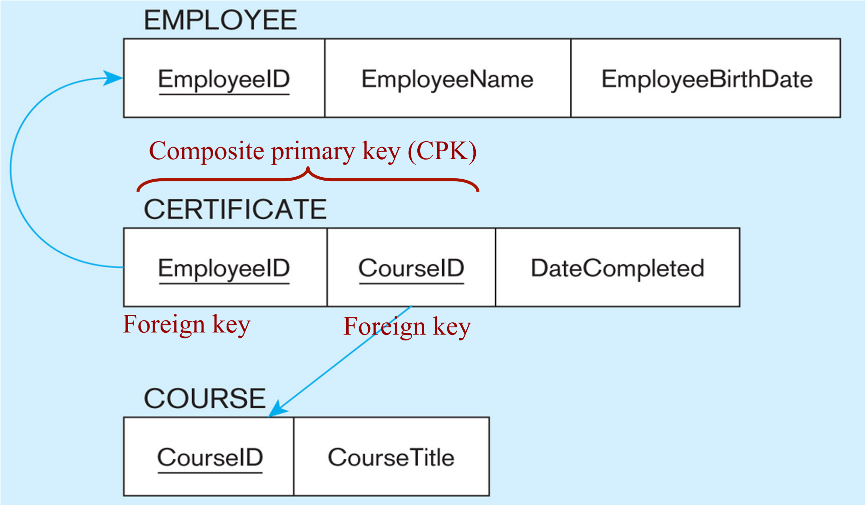
a) Completes relationship (M:N)



The *Completes* relationship will need to become a separate relation

Figure 4-13 Example of mapping an M:N relationship (cont.)

b) Three resulting relations



EMPLOYEE

EmployeeID	EmployeeName	EmployeeBirthDate

Composite primary key (CPK)

CERTIFICATE

EmployeeID	CourseID	DateCompleted

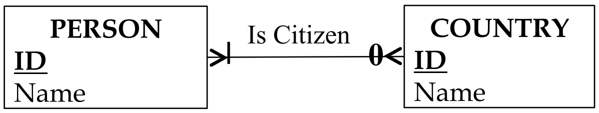
Foreign key

Foreign key

COURSE

CourseID	CourseTitle

You Try …



PERSON

ID

Name

Is Citizen

COUNTRY

ID

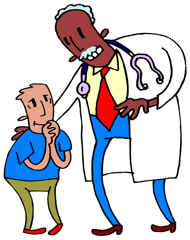
Name

Six Cases of Transforming E-R Diagrams into Relations

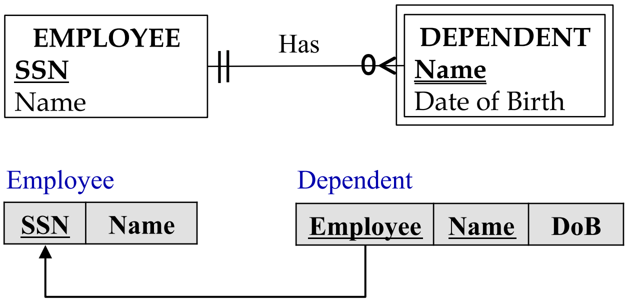
1. Map Regular Entities
2. Map Binary Relationships
3. Map Weak Entities
4. Map Associative Entities
5. Map Unary Relationships
6. Map Ternary (and n-ary) Relationships

3. Mapping Weak Entities

- A weak entity becomes a separate relation with a foreign key taken from the strong entity
- Primary key composed of:
  - Partial identifier of weak entity
  - Primary key of identifying relation (strong entity)



Weak Entities



EMPOYEE
<u>SSN</u>
Name

Has

DEPENDENT
<u>Name</u>
Date of Birth

Employee

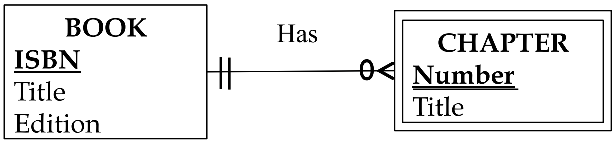
<u>SSN</u>	**Name**

Dependent

<u>Employee</u>	**<u>Name</u>**	**DoB**

- Transform the strong entity normally
- For the weak entity, the PK becomes the identifier, plus the PK of the identifying entity

You Try …



43

Six Cases of Transforming E-R Diagrams into Relations

1. Map Regular Entities
2. Map Binary Relationships
3. Map Weak Entities
4. Map Associative Entities
5. Map Unary Relationships
6. Map Ternary (and n-ary) Relationships

45

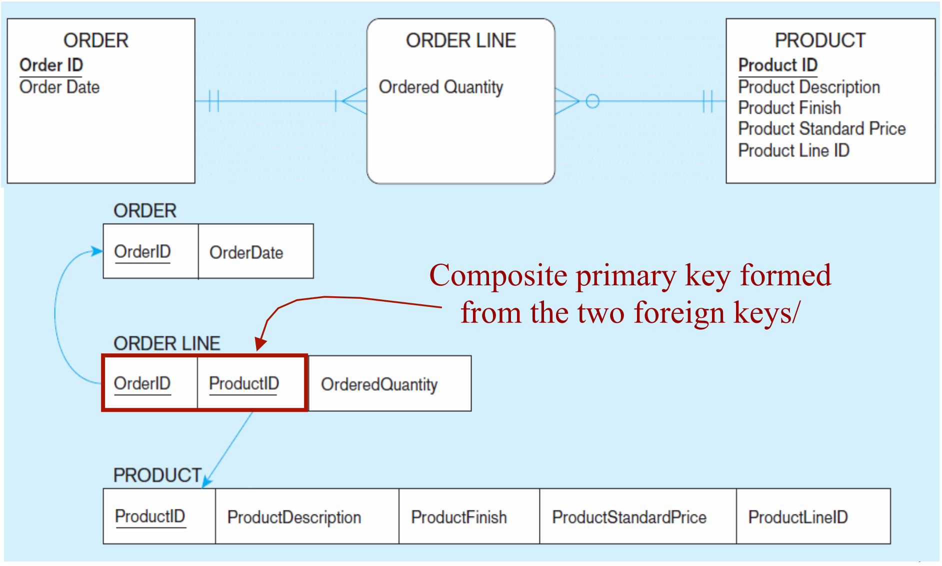
4. Mapping Associative Entities

- Identifier Not Assigned
  - Default primary key for the association relation is composed of the primary keys of the two entities (as in M:N relationship)
- Identifier Assigned
  - It is natural and familiar to end-users
  - Default identifier may not be unique

Figure 4-15: Mapping an associative entity

(a) Associative entity (ORDER LINE)

[Default primary key for the association relation is NOT assigned]



ORDER

<u>OrderID</u>	OrderDate

ORDER LINE

<u>OrderID</u>	<u>ProductID</u>	OrderedQuantity

PRODUCT

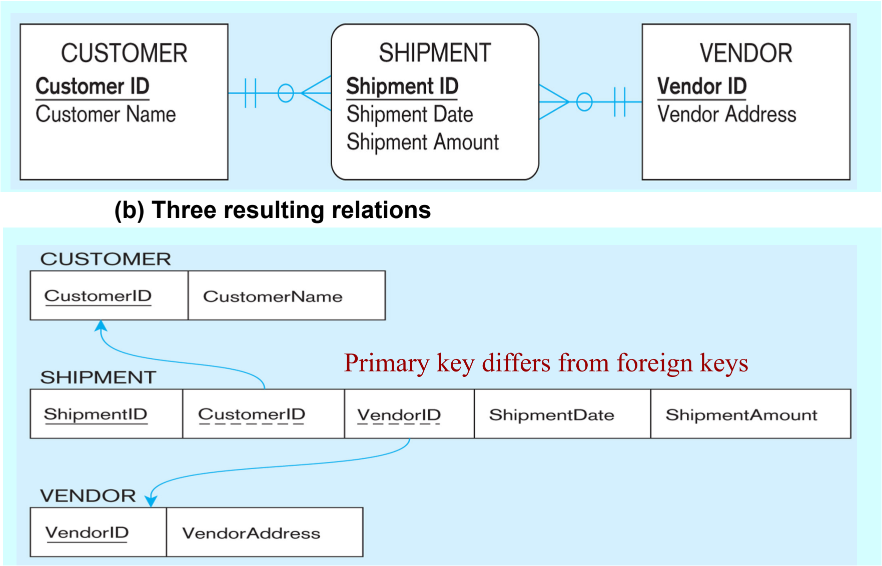
<u>ProductID</u>	ProductDescription	ProductFinish	ProductStandardPrice	ProductLineID

Composite primary key formed from the two foreign keys/

Figure 4-16: Mapping an associative entity

[Default primary key for the association relation is assigned]

(a) Associative entity (SHIPMENT)



CUSTOMER
Customer ID
Customer Name

SHIPMENT
Shipment ID
Shipment Date
Shipment Amount

VENDOR
Vendor ID
Vendor Address

(b) Three resulting relations

CUSTOMER

CustomerID	CustomerName

SHIPMENT

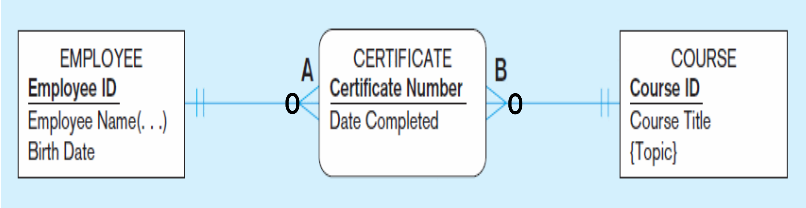
ShipmentID	CustomerID	VendorID	ShipmentDate	ShipmentAmount

VENDOR

VendorID	VendorAddress

Primary key differs from foreign keys

You Try …



EMPLOYEE
Employee ID
Employee Name(. . .)
Birth Date

CERTIFICATE
Certificate Number
Date Completed

COURSE
Course ID
Course Title
{Topic}

Six Cases of Transforming E-R Diagrams into Relations

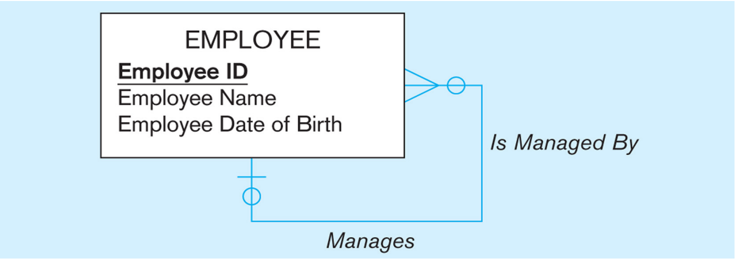
1. Map Regular Entities
2. Map Binary Relationships
3. Map Weak Entities
4. Map Associative Entities
5. Map Unary Relationships
6. Map Ternary (and n-ary) Relationships

5. Mapping Unary Relationships

- Same as other relationships, except that the FK may go to the same table.
- For one-to-many, the table has a reference to other rows of the same table.
- For many-to-many, an extra table has two FKs, both to the same table (Fig. 4-18).

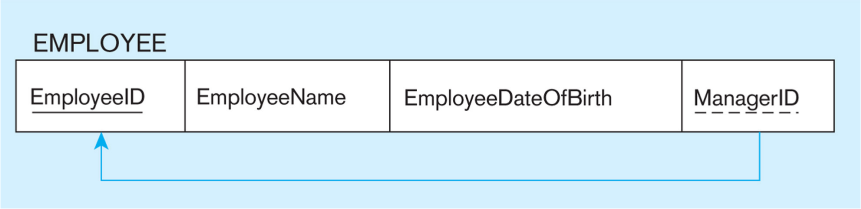
Figure 4-17 Mapping a unary 1:N relationship

(a) EMPLOYEE entity with unary relationship



A recursive FK is a FK in a relation that references the PK values of that same relation.

(b) EMPLOYEE relation with recursive foreign key

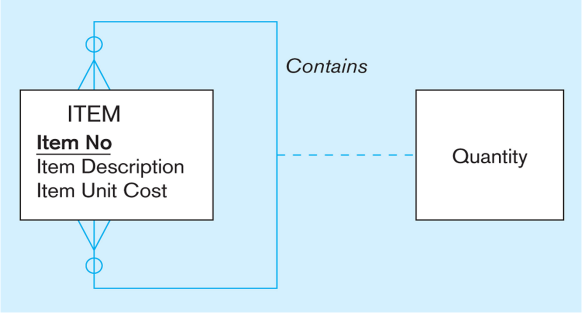


EMPLOYEE

EmployeeID	EmployeeName	EmployeeDateOfBirth	ManagerID

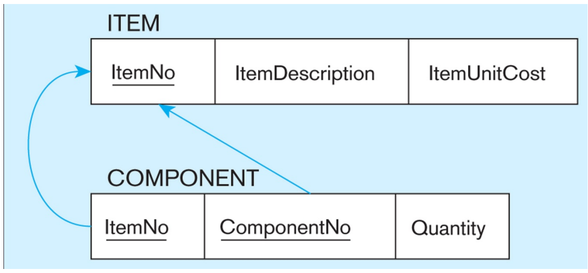
Figure 4-18: Mapping a unary M:N relationship

(a) Bill-of-materials relationships (M:N)



One table for the entity type.

(b) ITEM and COMPONENT relations



ITEM

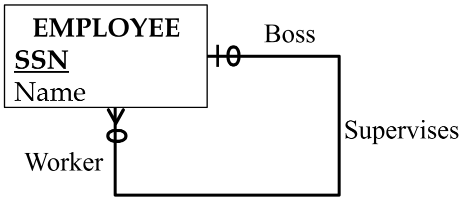
ItemNo	ItemDescription	ItemUnitCost

COMPONENT

ItemNo	ComponentNo	Quantity

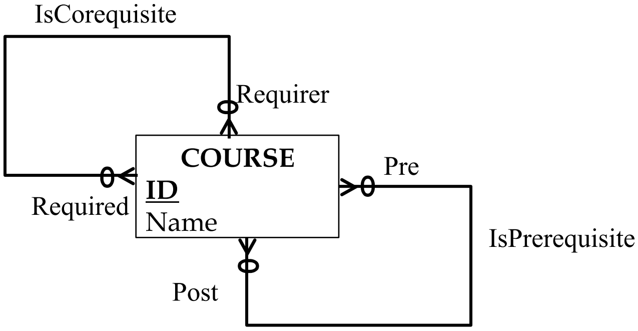
One table for an associative relation in which the primary key has two attributes, both taken from the primary key of the entity.

You Try – 1:M Unary



55

You Try – M:N Unary



57

Six Cases of Transforming E-R Diagrams into Relations

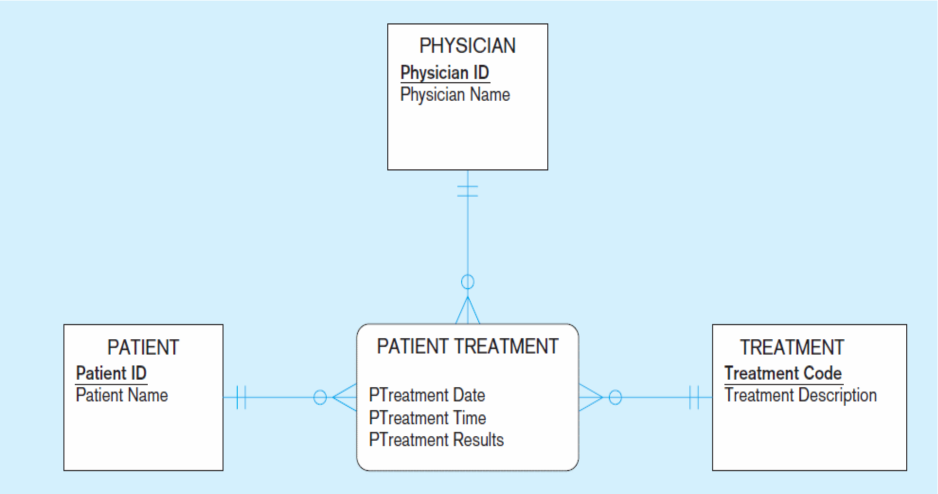
1. Map Regular Entities
2. Map Binary Relationships
3. Map Weak Entities
4. Map Associative Entities
5. Map Unary Relationships
6. Map Ternary (and n-ary) Relationships

6. Mapping Ternary Relationships

- One relation for each entity and one for the associative entity.
- Associative entity has foreign keys to each entity in the relationship

Figure 4-19 Mapping a ternary relationship

a) PATIENT TREATMENT Ternary relationship with associative entity



PHYSICIAN
Physician ID
Physician Name

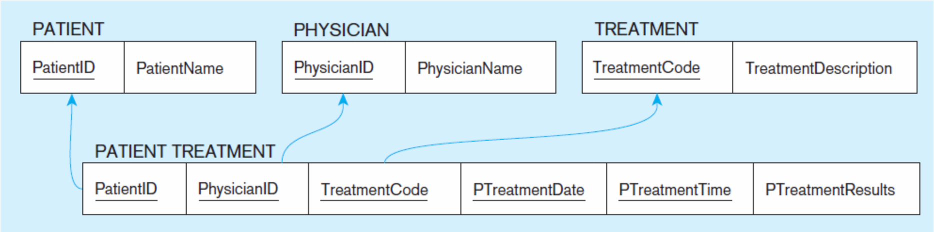
PATIENT
Patient ID
Patient Name

PATIENT TREATMENT
PTreatment Date
PTreatment Time
PTreatment Results

TREATMENT
Treatment Code
Treatment Description

Figure 4-19 Mapping a ternary relationship (cont.)

b) Mapping the ternary relationship PATIENT TREATMENT



PATIENT

PatientID	PatientName

PHYSICIAN

PhysicianID	PhysicianName

TREATMENT

TreatmentCode	TreatmentDescription

PATIENT TREATMENT

PatientID	PhysicianID	TreatmentCode	PTreatmentDate	PTreatmentTime	PTreatmentResults

Remember that the primary key MUST be unique.

This is why treatment date and time are included in the composite primary key.

But this makes a very cumbersome key…

It would be better to create a surrogate key like Treatment#.

(A patient may receive a treatment once in the morning, then the same treatment in the afternoon.)

From EE-R Diagrams
to
Relations (Tables)

EE-R to Relations

- Mapping Supertype/Subtype Relationships
- One relation for supertype and for each subtype
- Supertype attributes (including identifier and subtype discriminator) go into supertype relation
- Subtype attributes go into each subtype; primary key of supertype relation also becomes primary key of subtype relation
- 1:1 relationship established between supertype and each subtype, with supertype as primary table

Figure 4-20 Supertype/subtype relationship

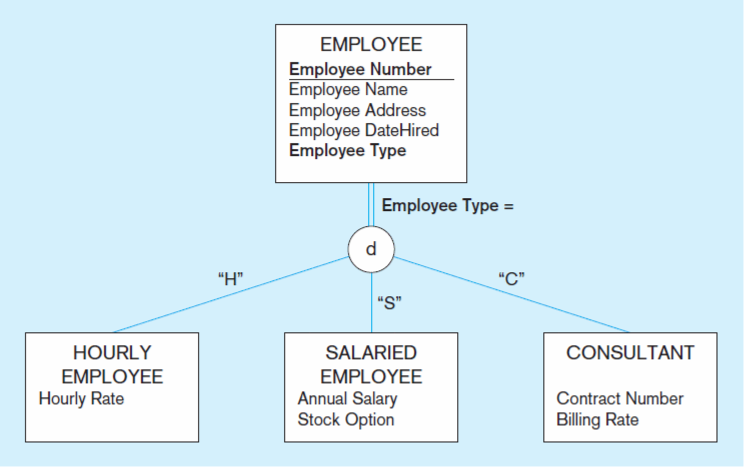
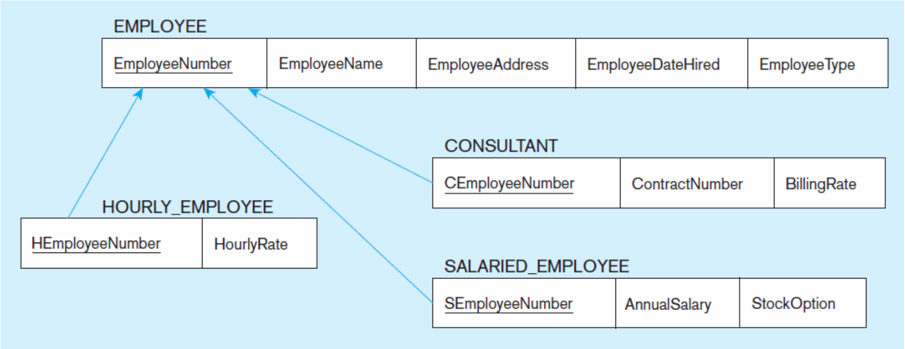


Figure 4-21
Mapping supertype/subtype relationships to relations



EMPLOYEE

EmployeeNumber	EmployeeName	EmployeeAddress	EmployeeDateHired	EmployeeType

CONSULTANT

CEmployeeNumber	ContractNumber	BillingRate

HOURLY_EMPLOYEE

HEmployeeNumber	HourlyRate

SALARIED_EMPLOYEE

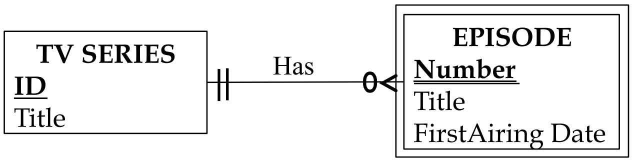
SEmployeeNumber	AnnualSalary	StockOption

These are implemented as one-to-one relationships.

Relational Model

Practice Exercises

#1



TV SERIES

ID

Title

Has

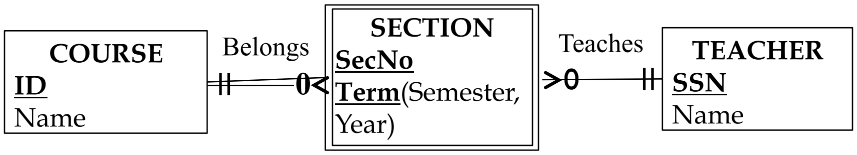
EPISODE

Number

Title

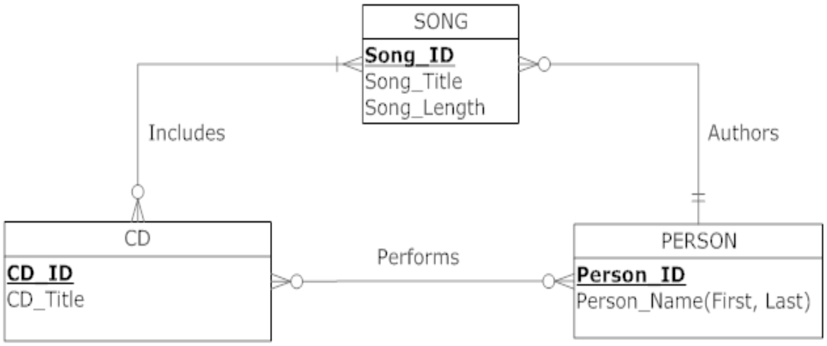
FirstAiring Date

#2


COURSE
ID
Name
Belongs
SECTION
SecNo
Term(Semester, Year)
Teaches
TEACHER
SSN
Name

70

#3


SONG
Song_ID
Song_Title
Song_Length
Includes
Authors
CD
CD_ID
CD_Title
Performs
PERSON
Person_ID
Person_Name(First, Last)

72

Next Topic

- Next topic is considered the most important theory in database management.
- What is it?
- **Normalization**

Table 4-6: Preview of Normalization

Below is a list of parking tickets issued by the Public Safety office to vehicles parked illegally on campus.

How would you organize such data into relations?

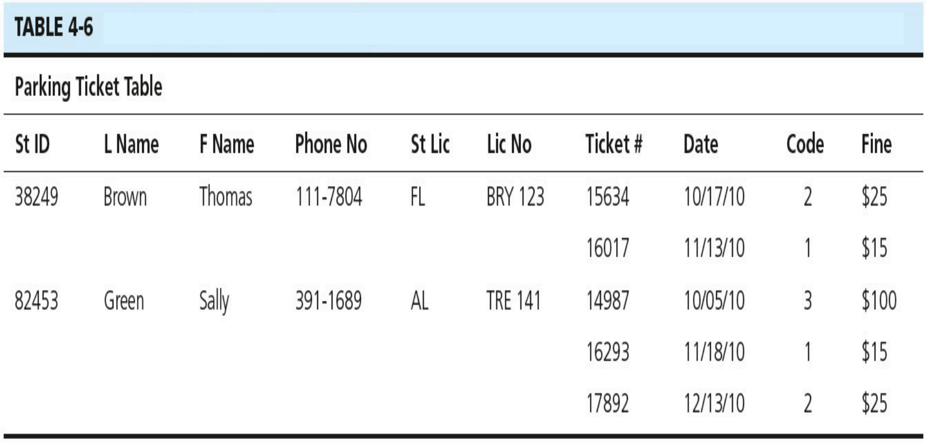


TABLE 4-6

Parking Ticket Table

St ID	L Name	F Name	Phone No	St Lic	Lic No	Ticket #	Date	Code	Fine
38249	Brown	Thomas	111-7804	FL	BRY 123	15634	10/17/10	2	$25
						16017	11/13/10	1	$15
82453	Green	Sally	391-1689	AL	TRE 141	14987	10/05/10	3	$100
						16293	11/18/10	1	$15
						17892	12/13/10	2	$25