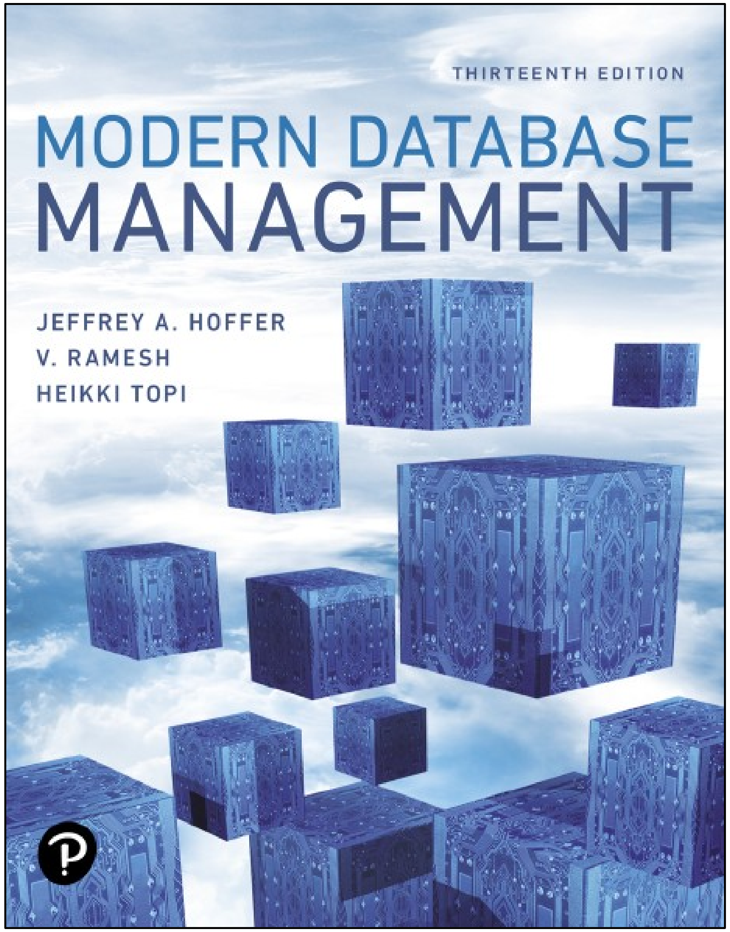
Modern Database Management

Thirteenth Edition



Chapter 4

Logical Database Design and the Relational Model

Learning Objectives

4.1 Define terms

4.2 List five properties of relations

4.3 State two properties of candidate keys

4.4 Define first, second, and third normal form

4.5 Describe problems from merging relations

4.6 Transform E-R and EER diagrams to relations

4.7 Create tables with entity and relational integrity constraints

4.8 Use normalization to decompose anomalous relations to well-structured relations

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

Relation

- A relation is a named, two-dimensional table of data.
- Consists of rows (records) and columns (attribute or field)
- Requirements for a table to qualify as a relation:
  - It must have a unique name.
  - Every attribute value must be atomic (not multivalued, not composite).
  - Every row must be unique (can't have two rows with exactly the same values for all their fields).
  - Attributes (columns) in tables must have unique names.
  - The order of the columns must be irrelevant.
  - The order of the rows must be irrelevant.

Note: All **relations** are in **1st Normal form.**

Correspondence with E-R Model

- Relations (tables) correspond with entity types and with many-to-many relationship types.
- Rows correspond with entity instances and with many-to-many relationship instances.
- Columns correspond with attributes.
- **Note**: The word **relation** (in relational database) is **not** the same as the word **relationship** (in E-R model).

Key Fields

- Keys are special fields that serve two main purposes:
  - **Primary keys** are **unique** identifiers of the relation. Examples include employee numbers, social security numbers, etc. **This guarantees that all rows are unique.**
  - **Foreign keys** are identifiers that enable a **dependent** relation (on the many side of a relationship) to refer to its **parent** relation (on the one side of the relationship).
- Keys can be **simple** (a single field) or **composite** (more than one field).
- Keys are usually used as indexes to speed up the response to user queries.

Figure 4-3 Schema for Four Relations (Pine Valley Furniture Company

a) EER notation

CUSTOMER

CustomerID	CustomerName	CustomerAddress	CustomerCity*	CustomerState*	CustomerPostalCode

ORDER

OrderID	OrderDate	CustomerID

ORDER LINE

OrderID	ProductID	OrderedQuantity

PRODUCT

ProductID	ProductDescription	ProductFinish	ProductStandardPrice	ProductLineID

Integrity Constraints (1 of 2)

- Domain Constraints
  - Allowable values for an attribute (includes data types and restrictions on values)
- Entity Integrity
  - No primary key attribute may be null. All primary key fields **MUST** contain data values.
- Referential Integrity
  - Rules that maintain consistency between the rows of two related tables.

Integrity Constraints (2 of 2)

- **Referential Integrity** – rule states that any foreign key value (on the relation of the many side) MUST match a primary key value in the relation of the one side.(Or the foreign key can be null.)
  - For 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

Figure 4-5 Referential Integrity Constraints (Pine Valley Furniture)

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

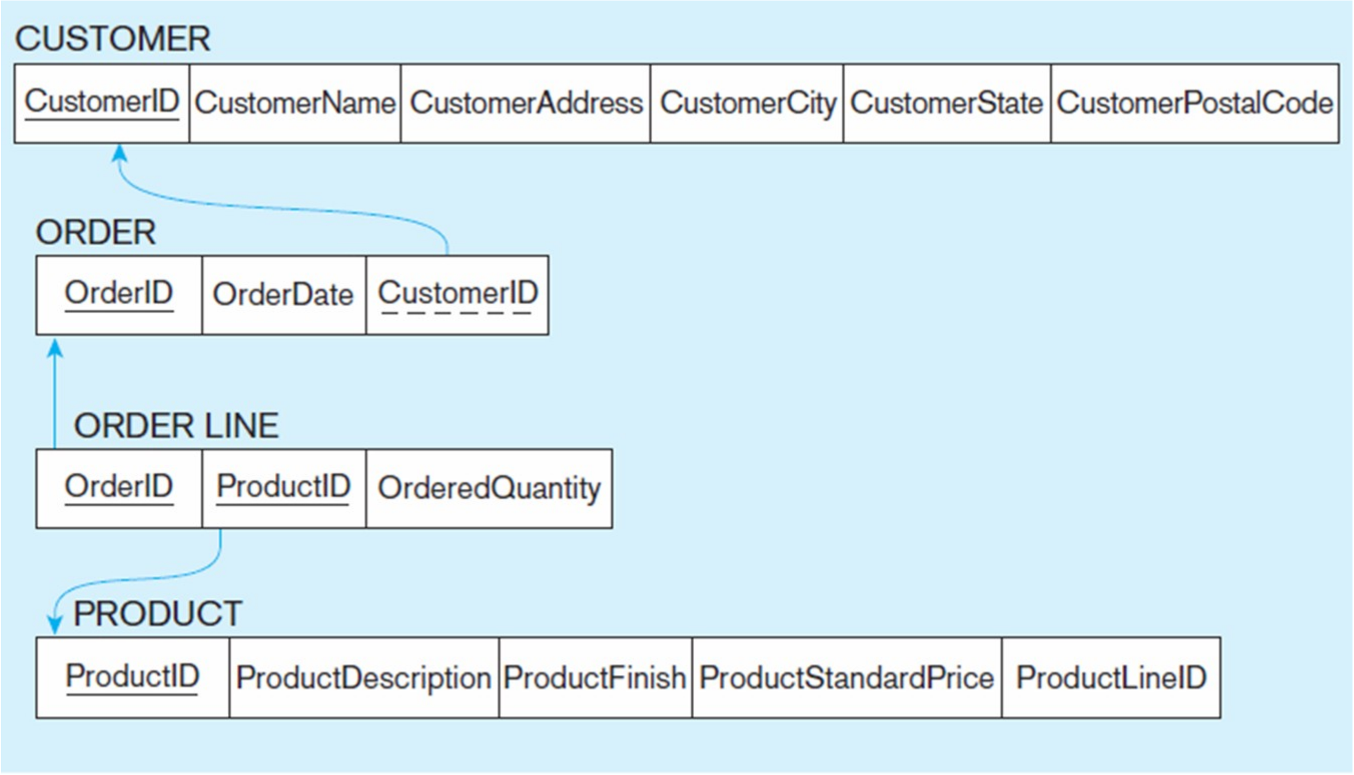


Figure 4-6 SQL Table Definitions

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

```
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));
```

Transforming EER Diagrams into Relations (1 of 7)

- Mapping Regular Entities to Relations
  - Simple attributes: E-R attributes map directly onto the relation
  - Composite attributes: Use only their simple, component attributes
  - Multivalued attributes: Become a separate relation with a foreign key taken from the superior entity

Figure 4-8 Example of Mapping a Regular Entity

a) CUSTOMER entity type

b) CUSTOMER relation

CUSTOMER

<u>CustomerID</u>	CustomerName	CustomerAddress	CustomerPostalCode

Figure 4-9 Mapping a Composite Attribute

a) CUSTOMER entity type with composite attribute

CUSTOMER

Customer ID

Customer Name

Customer Address

(Customer Street, Customer City, Customer State)

Customer Postal Code

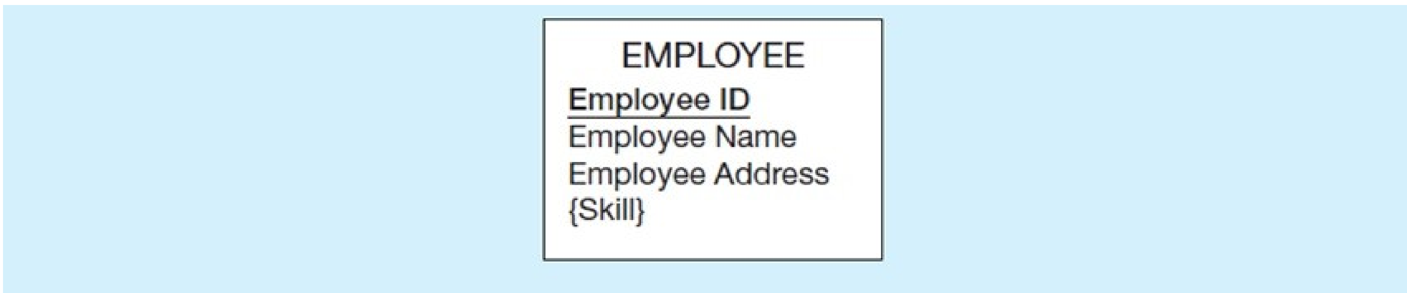
b) CUSTOMER relation with address detail

CUSTOMER

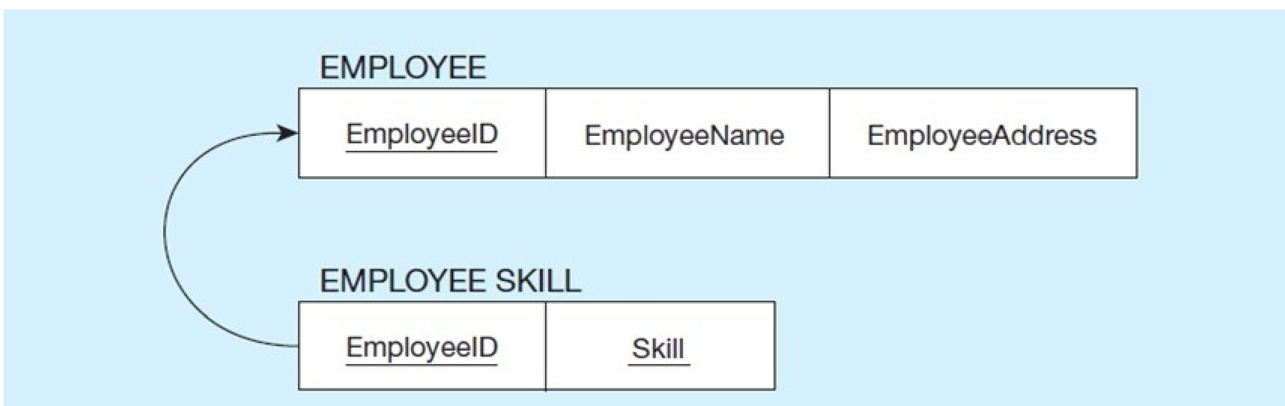
CustomerID	CustomerName	CustomerStreet	CustomerCity	CustomerState	CustomerPostalCode

Figure 4-10 Mapping an Entity with a Multivalued Attribute

a) EMPLOYEE entity type with multivalued attribute



b) EMPLOYEE and EMPLOYEE SKILL relations

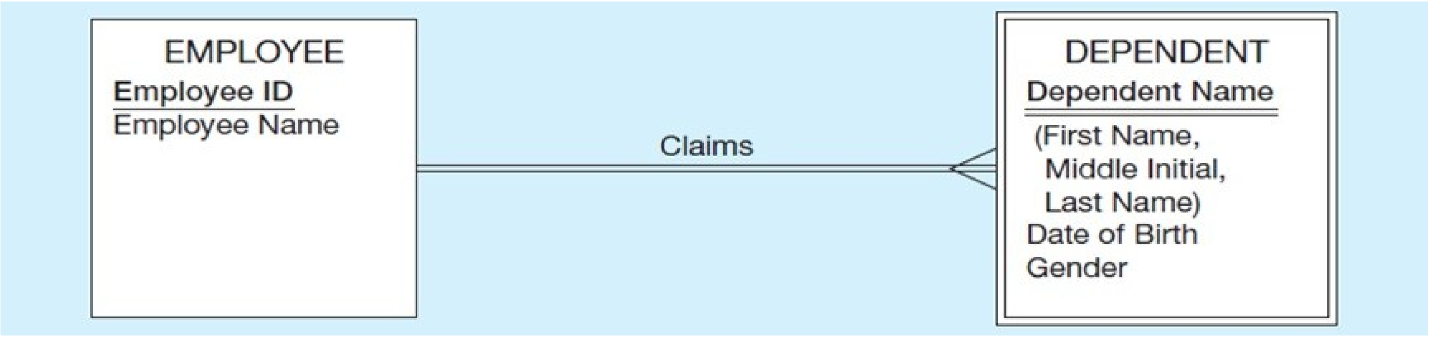


Transforming EER Diagrams into Relations (2 of 7)

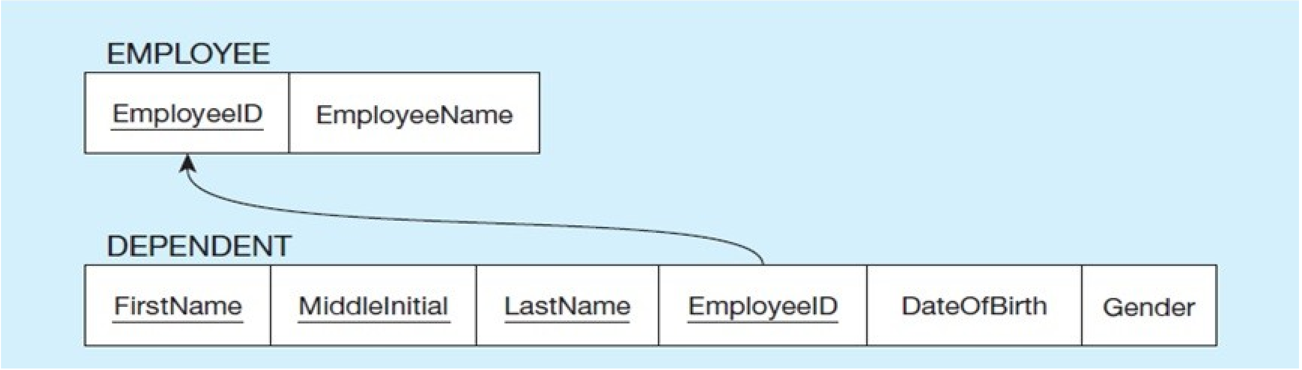
- Mapping Weak Entities
  - Becomes a separate relation with a foreign key taken from the superior entity
  - Primary key composed of:
    - Partial identifier of weak entity
    - Primary key of identifying relation (strong entity)

Figure 4-11 Example of Mapping a Weak Entity

a) Weak entity DEPENDENT



b) Relations resulting from weak entity

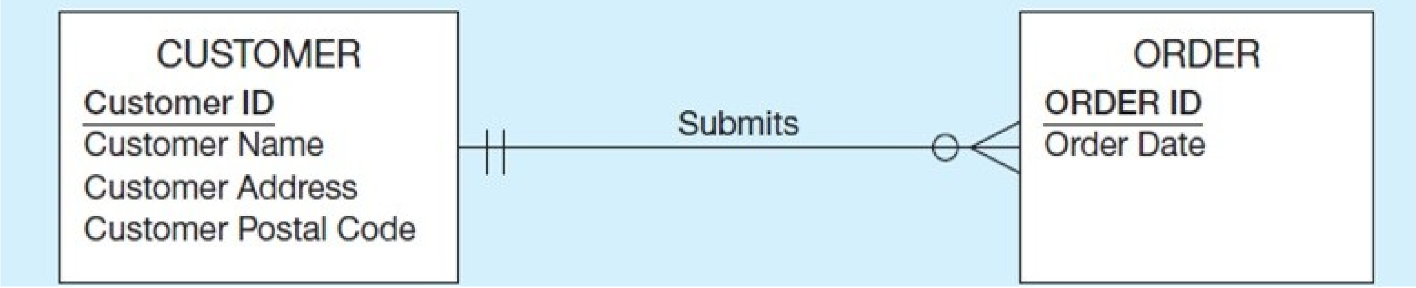


Transforming EER Diagrams into Relations (3 of 7)

- Mapping Binary Relationships
  - **One-to-Many** – Primary key on the one side becomes a foreign key on the many side
  - **Many-to-Many** – Create a **new relation** with the primary keys of the two entities as its primary key
  - **One-to-One** – Primary key on mandatory side becomes a foreign key on optional side

Figure 4-12 Example of Mapping a 1:N Relationship

a) Relationship between customers and orders



b) CUSTOMER and ORDER relations with a foreign key in ORDER

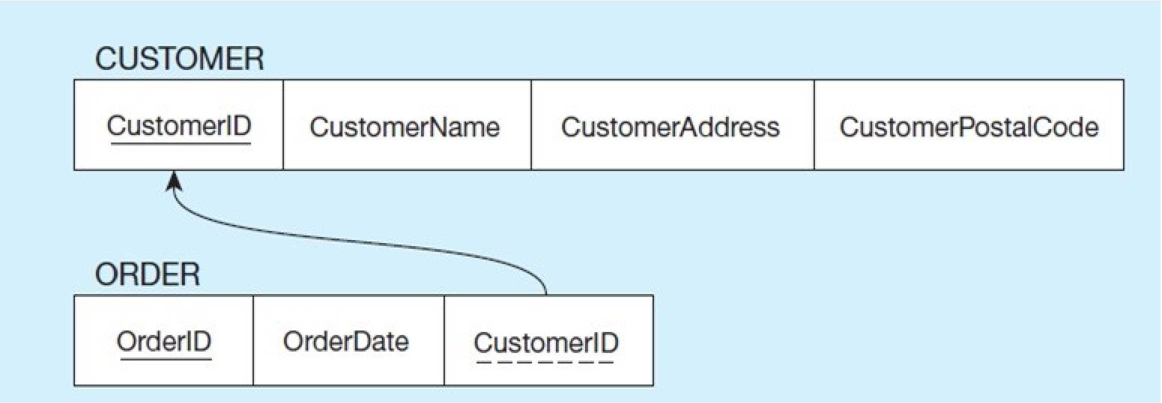
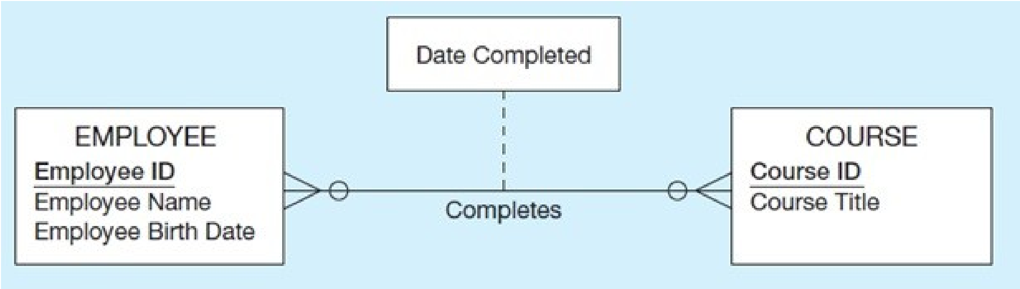


Figure 4-13 Example of Mapping an M:N Relationship

a) Completes relationship (M:N)



b) Three resulting relations

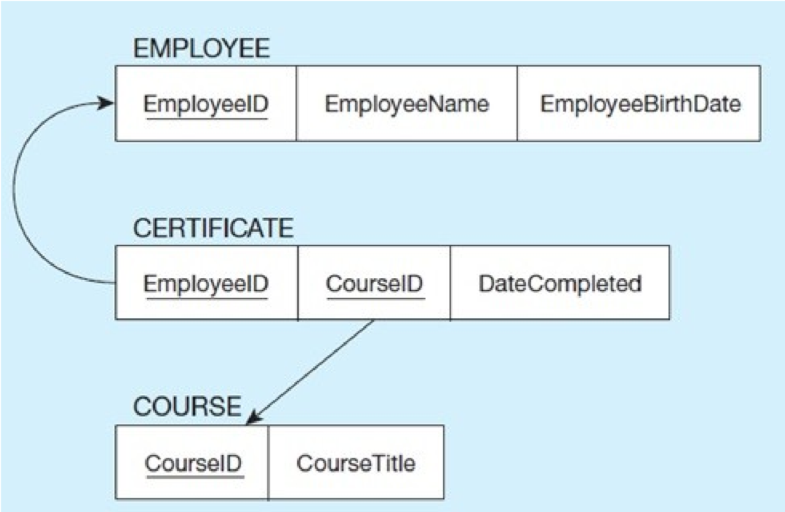
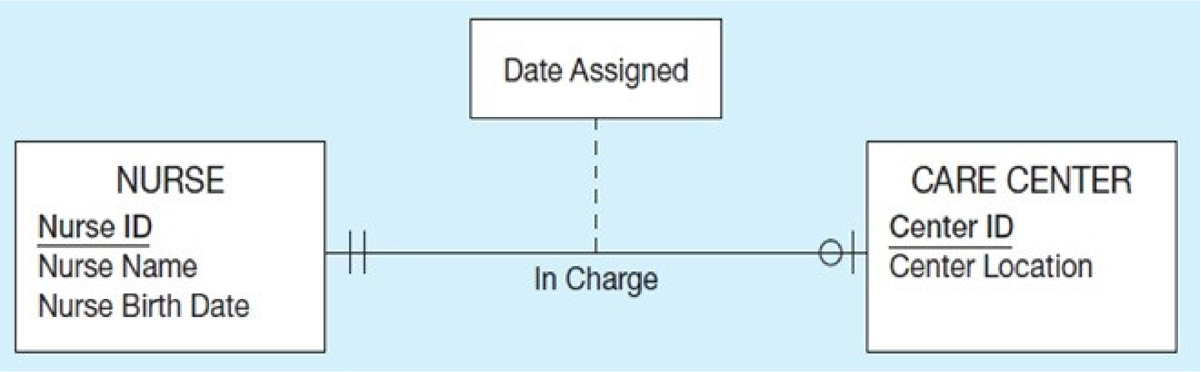
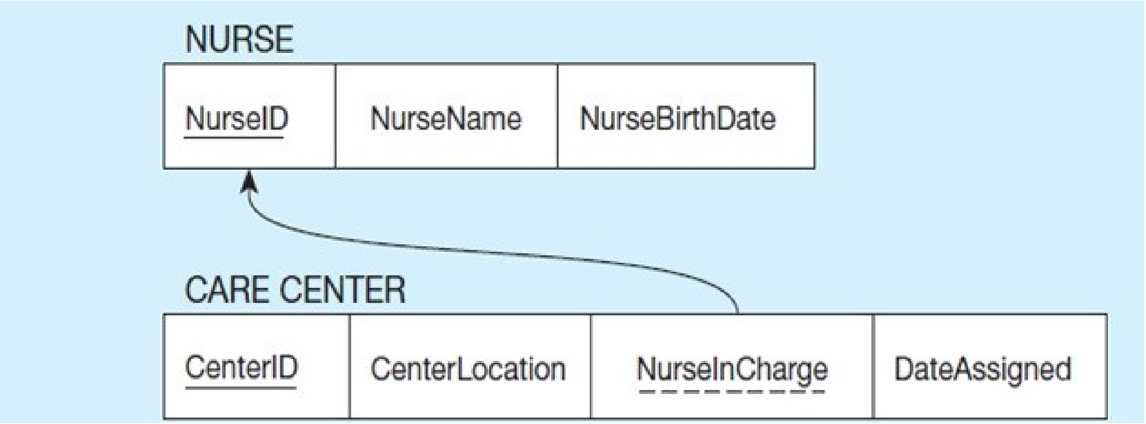


Figure 4-14 Example of Mapping a Binary 1:1 Relationship

a) In charge relationship (binary 1 : 1)



b) Resulting relations



Transforming EER Diagrams into Relations (4 of 7)

- 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 Example of Mapping an Associative Entity (1 of 2)

a) An associative entity

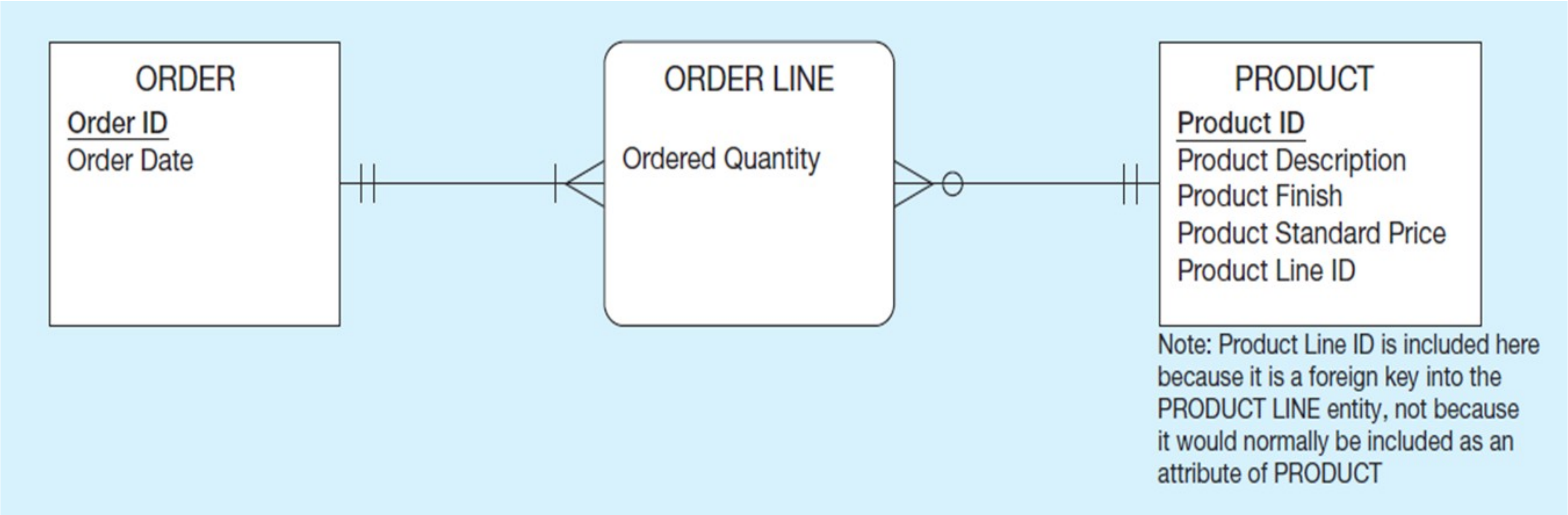
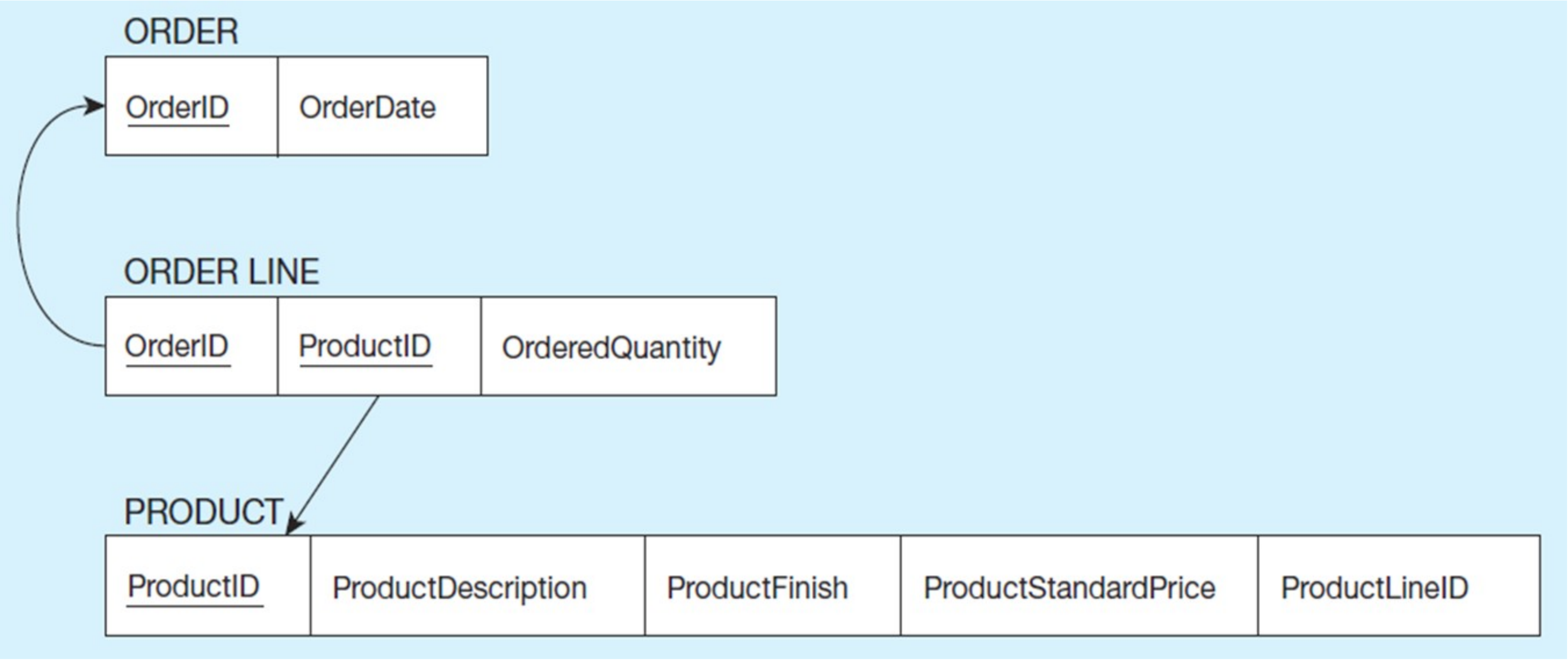


Figure 4-15 Example of Mapping an Associative Entity (2 of 2)

b) Three resulting relations



Composite primary key formed from the two foreign keys

Figure 4-16 Example of Mapping an Associative Entity with an Identifier (1 of 2)

a) SHIPMENT associative entity

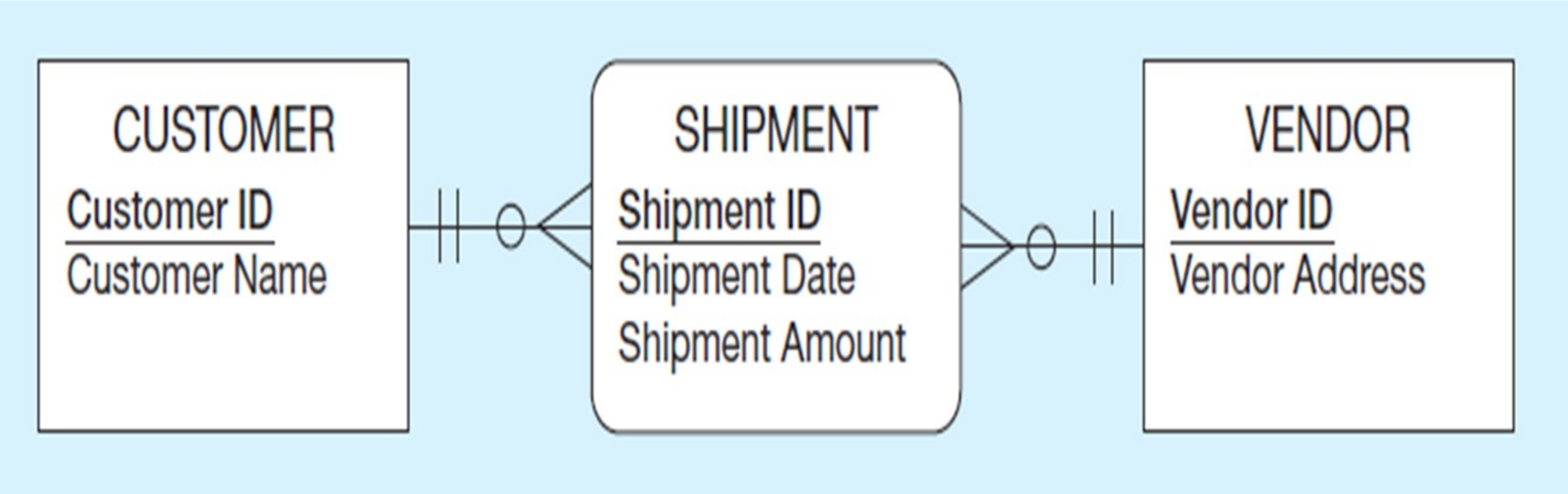
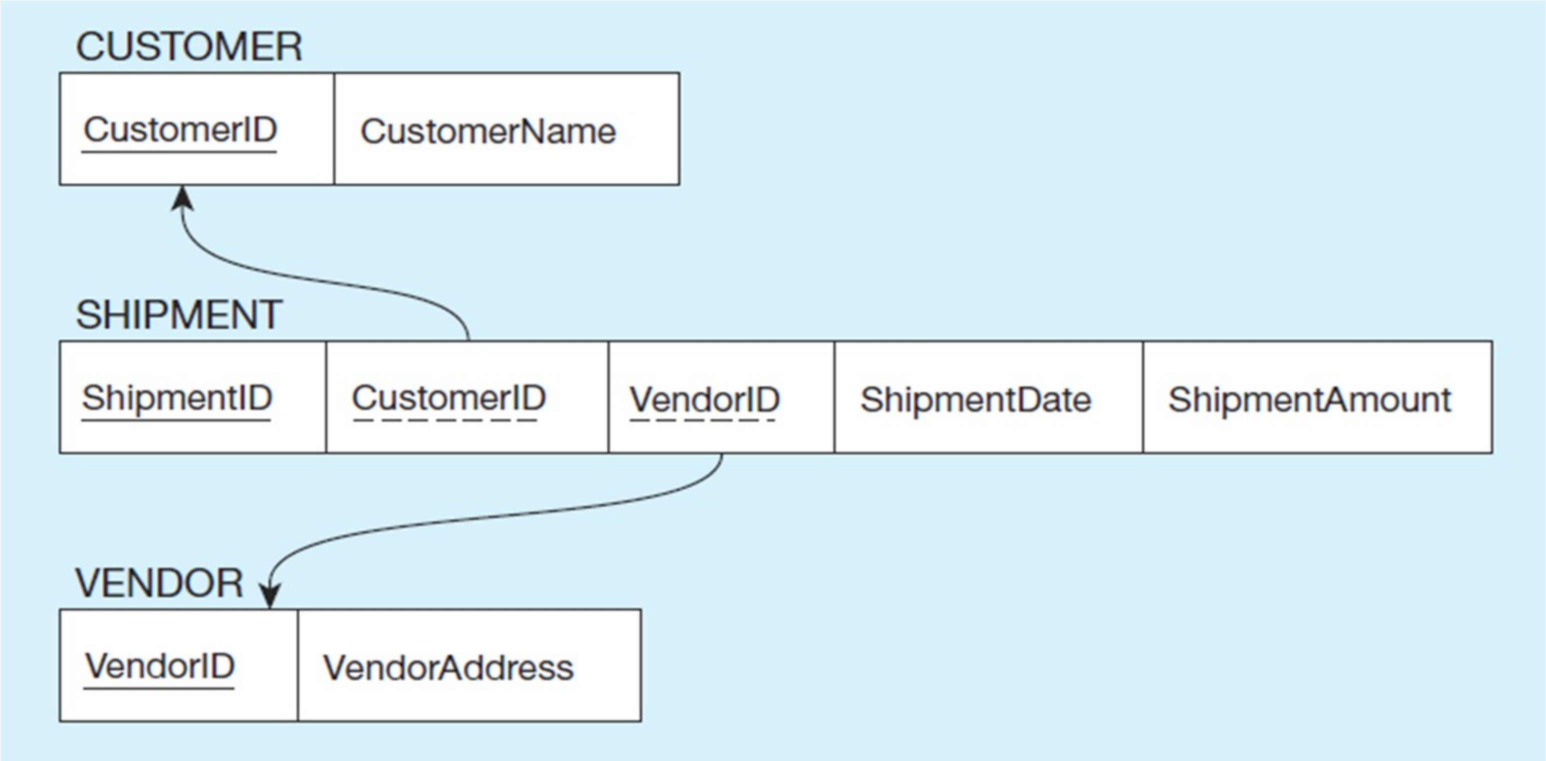


Figure 4-16 Example of Mapping an Associative Entity with an Identifier (2 of 2)

b) Three resulting relations



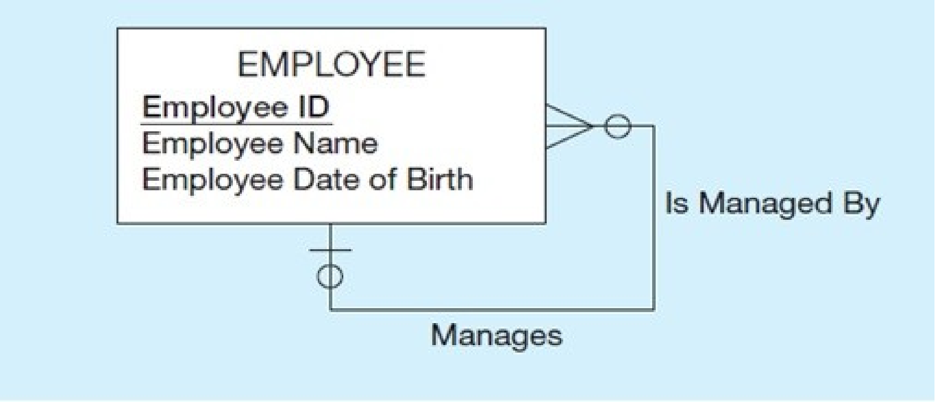
Primary key differs from foreign keys

Transforming EER Diagrams into Relations (5 of 7)

- Mapping Unary Relationships
  - One-to-Many – Recursive foreign key in the same relation
  - Many-to-Many – Two relations:
    - One for the entity type
    - One for an associative relation in which the primary key has two attributes, both taken from the primary key of the entity

Figure 4-17 Example of Mapping a Unary 1:N Relationship

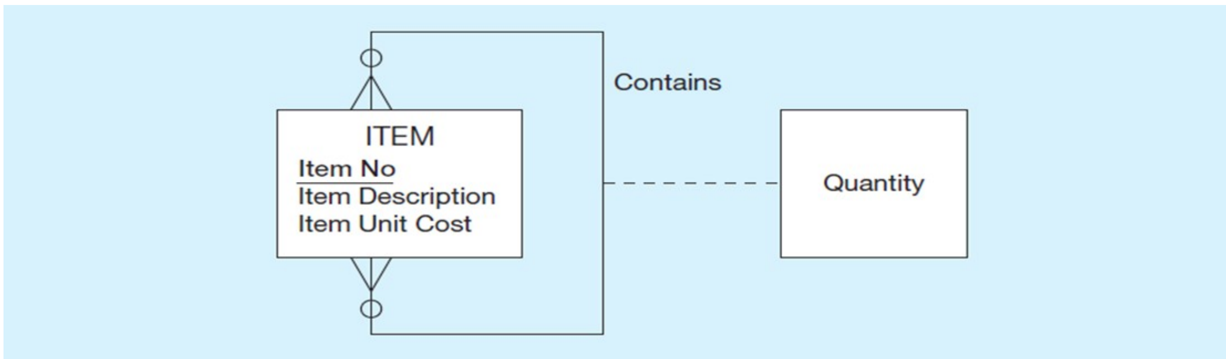
a) EMPLOYEE entity with unary relationship



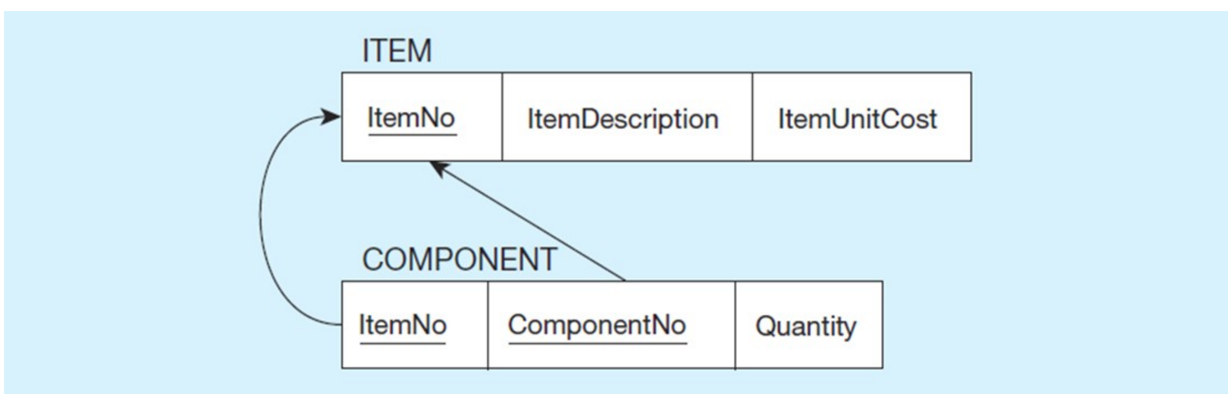
b) EMPLOYEE relation with recursive foreign key

Figure 4-18 Example of Mapping a Unary M:N Relationship

a) EMPLOYEE entity with unary relationship



b) EMPLOYEE relation with recursive foreign key



Transforming EER Diagrams into Relations (6 of 7)

- Mapping Ternary (and n-ary) Relationships
  - One relation for each entity and one for the associative entity
  - Associative entity has foreign keys to each entity in the relationship

Figure 4-18 Example of Mapping a Ternary Relationship (1 of 2)

a) PATIENT TREATMENT ternary relationship with associative entity

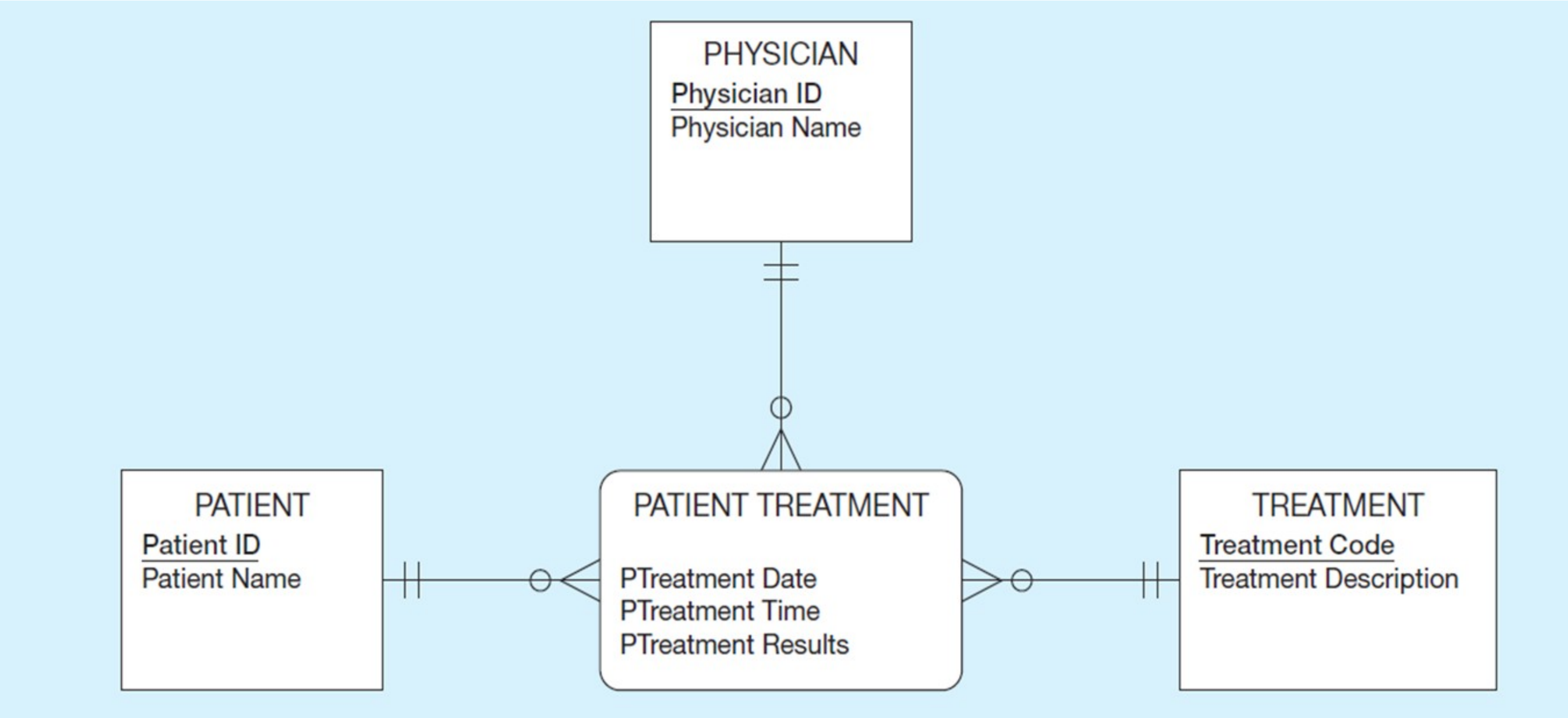
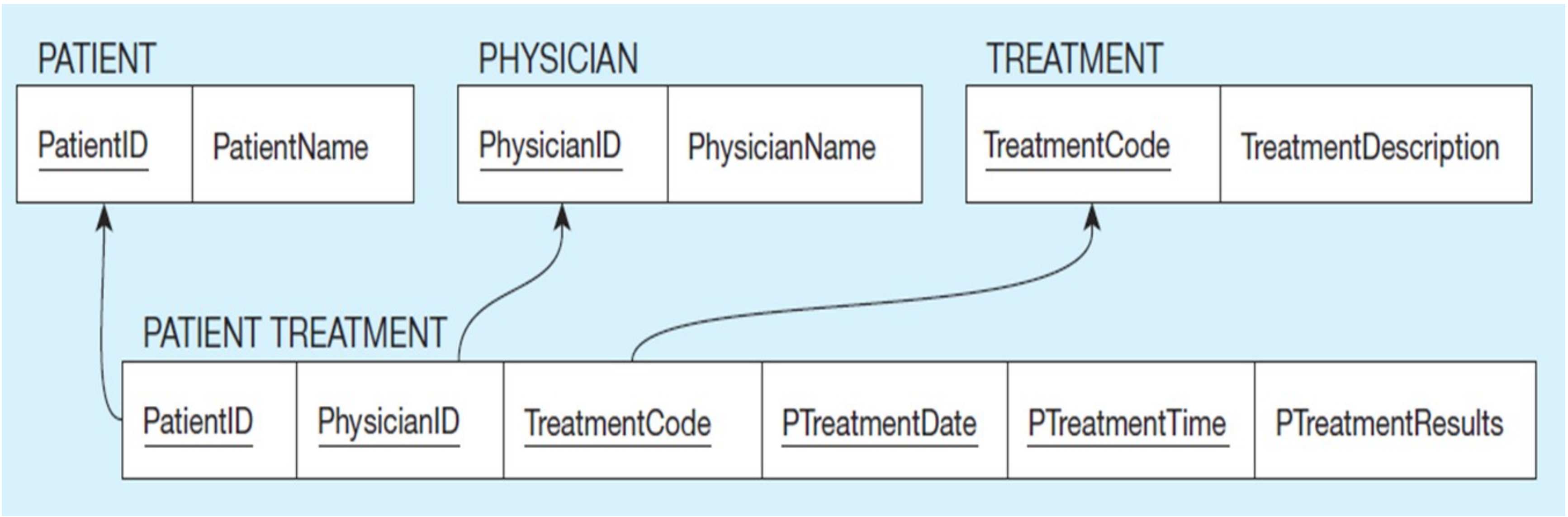


Figure 4-18 Example of Mapping a Ternary Relationship (2 of 2)

b) Four resulting relations



Transforming EER Diagrams into Relations (7 of 7)

- 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 Relationships

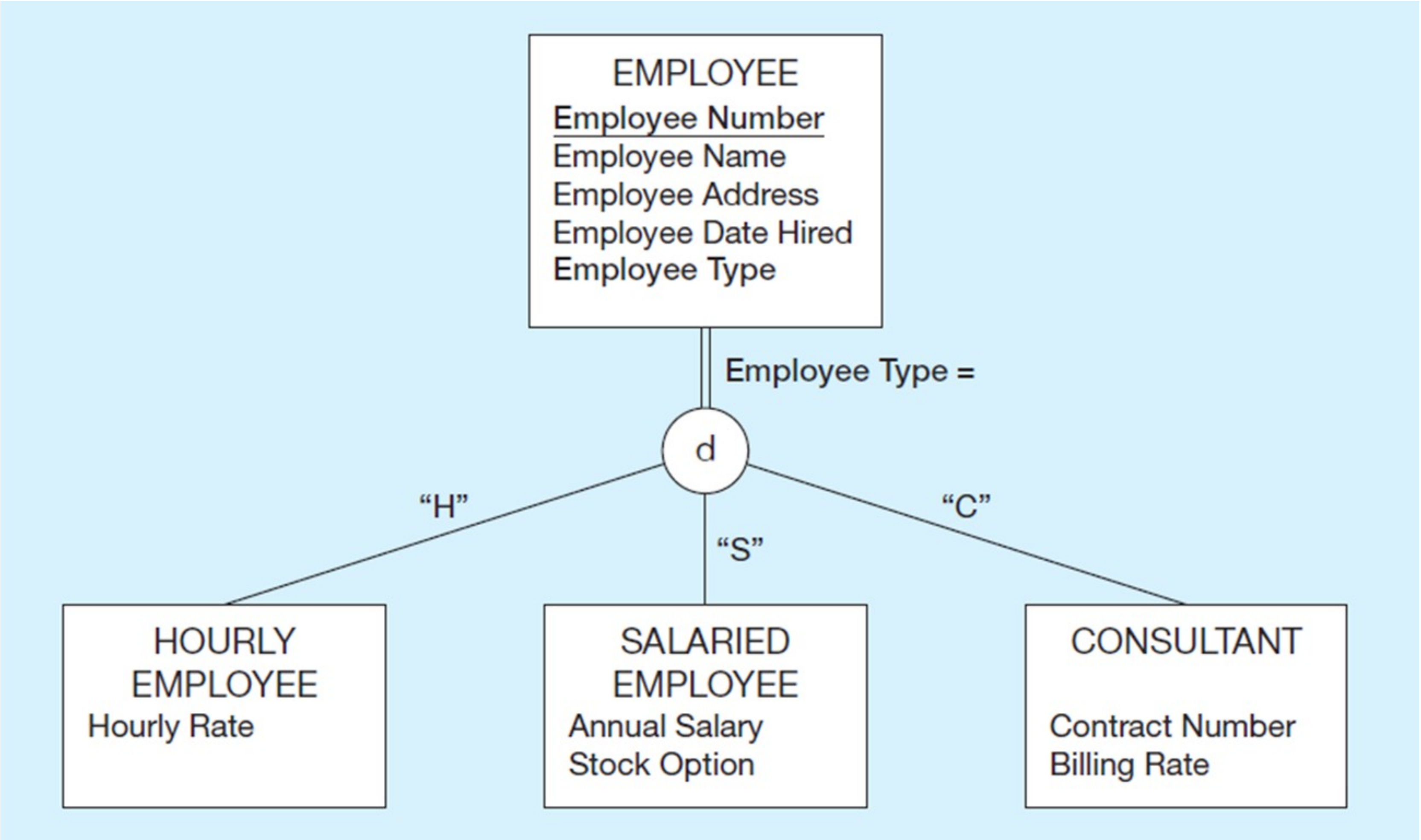
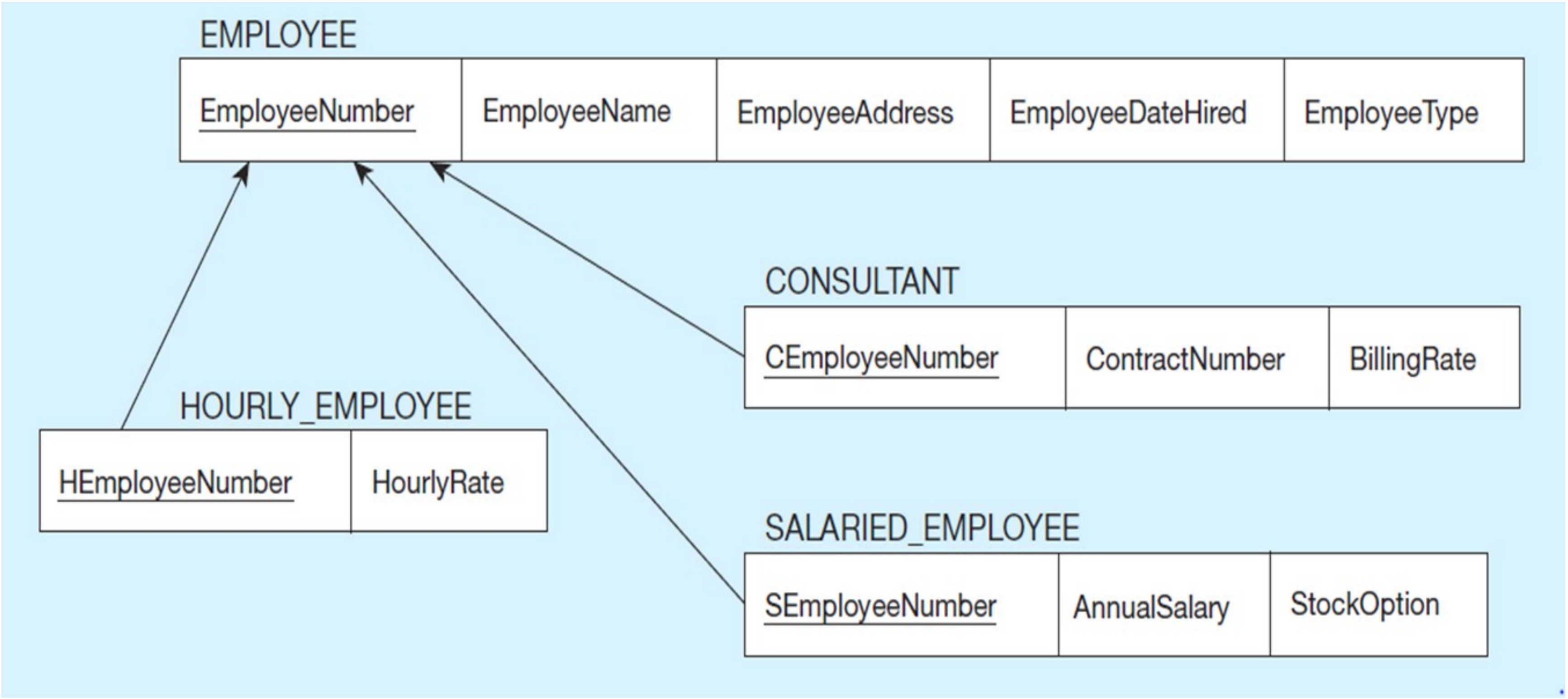


Figure 4-21 Example of Mapping Supertype/Subtype Relationships to Relations



These are implemented as one-to-one relationships.

Data Normalization

- Primarily a tool to validate and improve a logical design so that it satisfies certain constraints that **avoid unnecessary duplication of data**
- The process of decomposing relations with anomalies to produce smaller, **well-structured relations**

Well-Structured Relations

- Relations that contain minimal data redundancy and allow users to insert, delete, and update rows without causing data inconsistencies
- Goal is to avoid anomalies
  - **Insertion Anomaly** – adding new rows forces user to create duplicate data
  - **Deletion Anomaly** – deleting rows may cause a loss of data that would be needed for other future rows
  - **Modification Anomaly** – changing data in a row forces changes to other rows because of duplication

Example–Figure 4-2b

EMPLOYEE2

EmpID	Name	DeptName	Salary	CourseTitle	DateCompleted
100	Margaret Simpson	Marketing	48,000	SPSS	6/19/2018
100	Margaret Simpson	Marketing	48,000	Surveys	10/7/2018
140	Alan Beeton	Accounting	52,000	Tax Acc	12/8/2018
110	Chris Lucero	Info Systems	43,000	Visual Basic	1/12/2018
110	Chris Lucero	Info Systems	43,000	C++	4/22/2018
190	Lorenzo Davis	Finance	55,000		
150	Susan Martin	Marketing	42,000	SPSS	6/19/2018
150	Susan Martin	Marketing	42,000	Java	8/12/2018

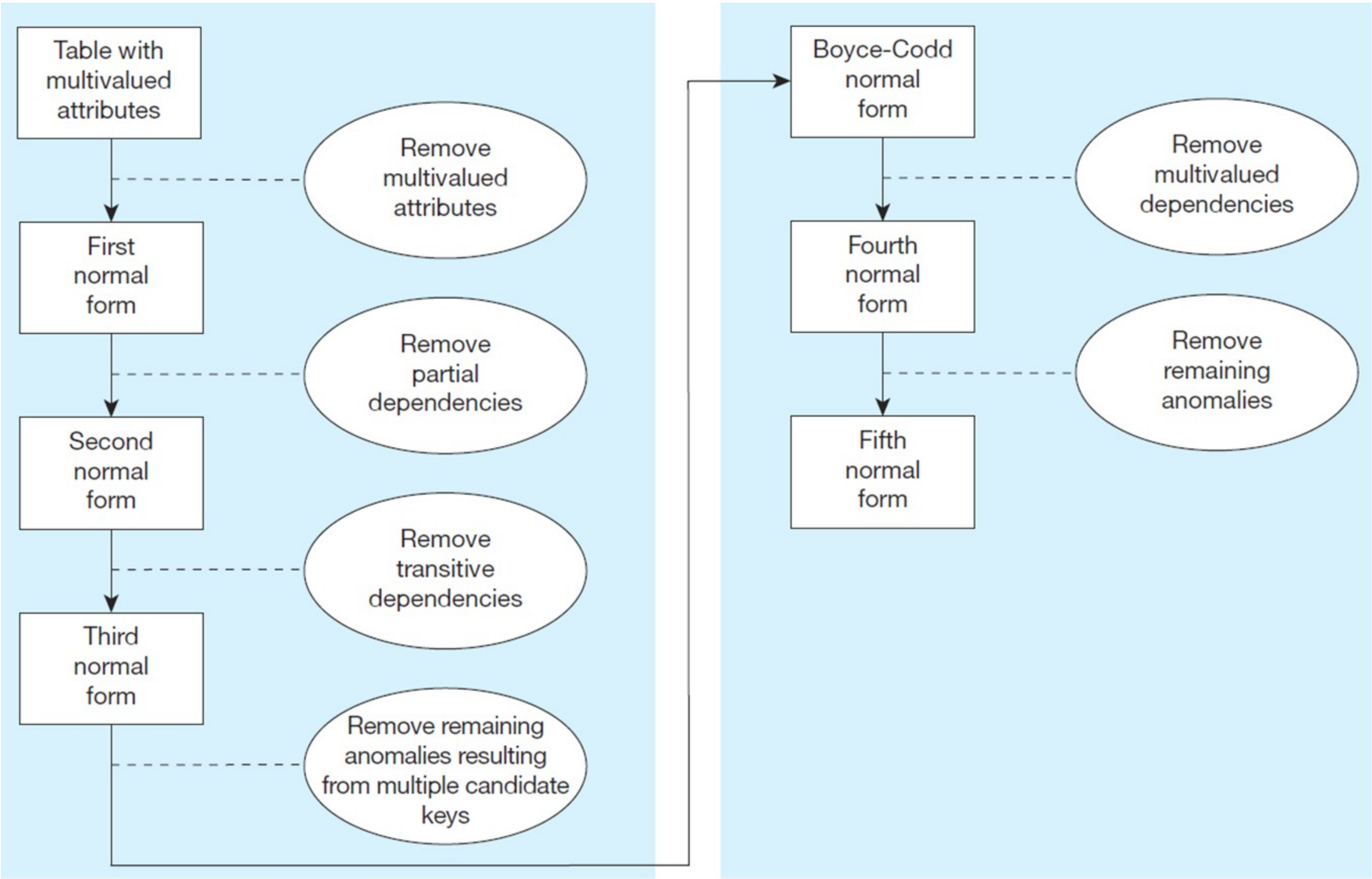
Question: Is this a relation? Answer: Yes; unique rows and no multivalued attributes

Question: What's the primary key? Answer: Composite — EmpID, CourseTitle

Anomalies in This Relation (1 of 2)

- **Insertion** – can’t enter a new employee without having the employee take a class (or at least empty fields of class information)
- **Deletion** – if we remove employee 140, we lose information about the existence of a Tax Acc class
- **Modification** – giving a salary increase to employee 100 forces us to update multiple records

Figure 4.22 Steps in Normalization



Functional Dependencies and Keys

- Functional Dependency: The value of one attribute (the **determinant**) determines the value of another attribute
- Candidate Key:
  - A unique identifier. One of the candidate keys will become the primary key
    - E.g., perhaps there is both credit card number and SS# in a table…in this case both are candidate keys.
  - Each non-key field is functionally dependent on every candidate key.

First Normal Form

- No multivalued attributes
- Every attribute value is atomic
- Fig. 4-25 **is not** in 1st Normal Form (multivalued attributes) → it is not a relation.
- Fig. 4-26 *is* in 1st Normal form.
- **All relations are in 1st Normal Form.**

Figure 4.25 Invoice Data (Pine Valley Furniture Company)

OrderID	Order Date	Customer ID	Customer Name	Customer Address	Product ID	Product Description	Product Finish	Product StandardPrice	Ordered Quantity
1006	10/24/2018	2	Value Furniture	Plano, TX	7	Dining Table	Natural Ash	800.00	2
					5	Writer's Desk	Cherry	325.00	2
					4	Entertainment Center	Natural Maple	650.00	1
1007	10/25/2018	6	Furniture Gallery	Boulder, CO	11	4–Dr Dresser	Oak	500.00	4
					4	Entertainment Center	Natural Maple	650.00	3

Table with multivalued attributes, not in 1st normal form.

This is **not** a relation.

Figure 4.26 INVOICE Relation (1NF) (Pine Valley Furniture Company)

OrderID	Order Date	Customer ID	Customer Name	Customer Address	ProductID	Product Description	Product Finish	Product Standard Price	Ordered Quantity
1006	10/24/2018	2	Value Furniture	Plano, TX	7	Dining Table	Natural Ash	800.00	2
1006	10/24/2018	2	Value Furniture	Plano, TX	5	Writer's Desk	Cherry	325.00	2
1006	10/24/2018	2	Value Furniture	Plano, TX	4	Entertainment Center	Natural Maple	650.00	1
1007	10/25/2018	6	Furniture Gallery	Boulder, CO	11	4–Dr Dresser	Oak	500.00	4
1007	10/25/2018	6	Furniture Gallery	Boulder, CO	4	Entertainment Center	Natural Maple	650.00	3

This is a relation, but not a well-structured one.

Pearson

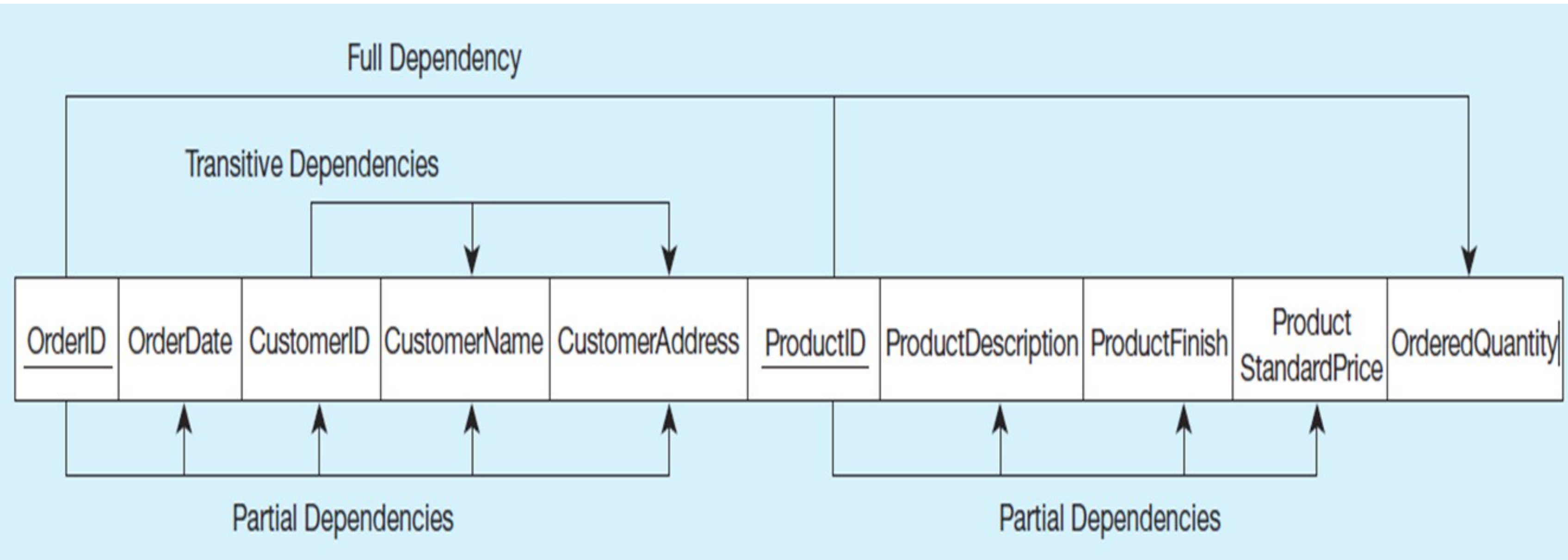
Anomalies in This Relation (2 of 2)

- **Insertion** – if new product is ordered for order 1007 of existing customer, customer data must be re-entered, causing duplication
- **Deletion** – if we delete the Dining Table from Order 1006, we lose information concerning this item's finish and price
- **Update** – changing the price of product ID 4 requires update in multiple records

Second Normal Form

- 1NF plus **every non-key attribute is fully functionally dependent on the ENTIRE primary key**
  - Every non-key attribute must be defined by the entire key, not by only part of the key
  - No partial functional dependencies

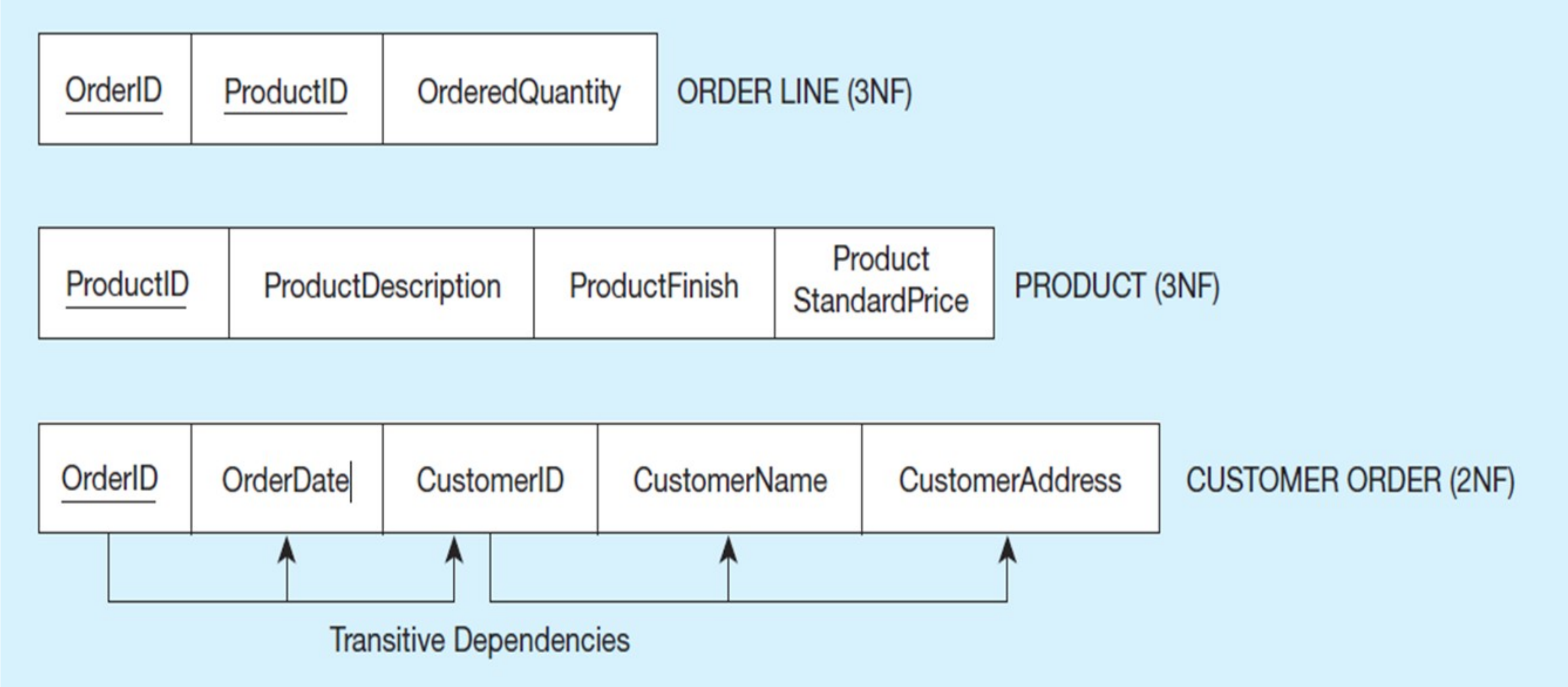
Figure 4-27 Functional Dependency Diagram for Invoice



OrderID → OrderDate, CustomerID, CustomerName, CustomerAddress CustomerID → CustomerName, CustomerAddress ProductID → ProductDescription, ProductFinish, ProductStandardPrice OrderID, ProductID → OrderQuantity

Therefore, **not** in 2nd Normal Form

Figure 4-28 Removing Partial Dependencies



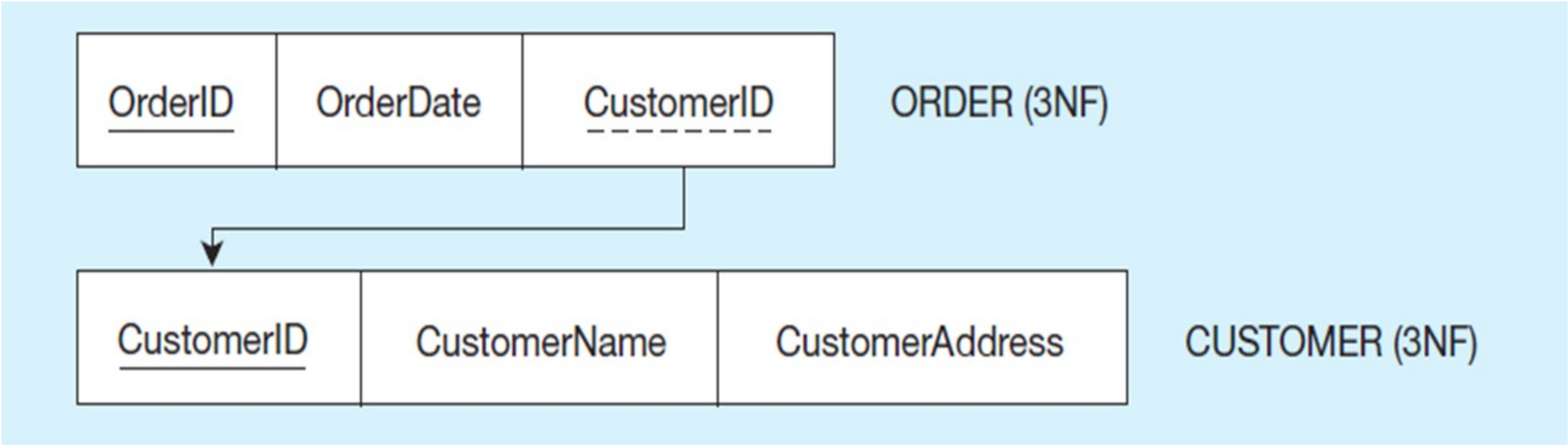
Getting it into Second Normal Form

Partial dependencies are removed, but there are still transitive dependencies

Third Normal Form

- 2NF PLUS **no transitive dependencies** (functional dependencies on non-primary-key attributes)
- Note: This is called transitive, because the primary key is a determinant for another attribute, which in turn is a determinant for a third
- Solution: Non-key determinant with transitive dependencies go into a new table; non-key determinant becomes primary key in the new table and stays as foreign key in the old table

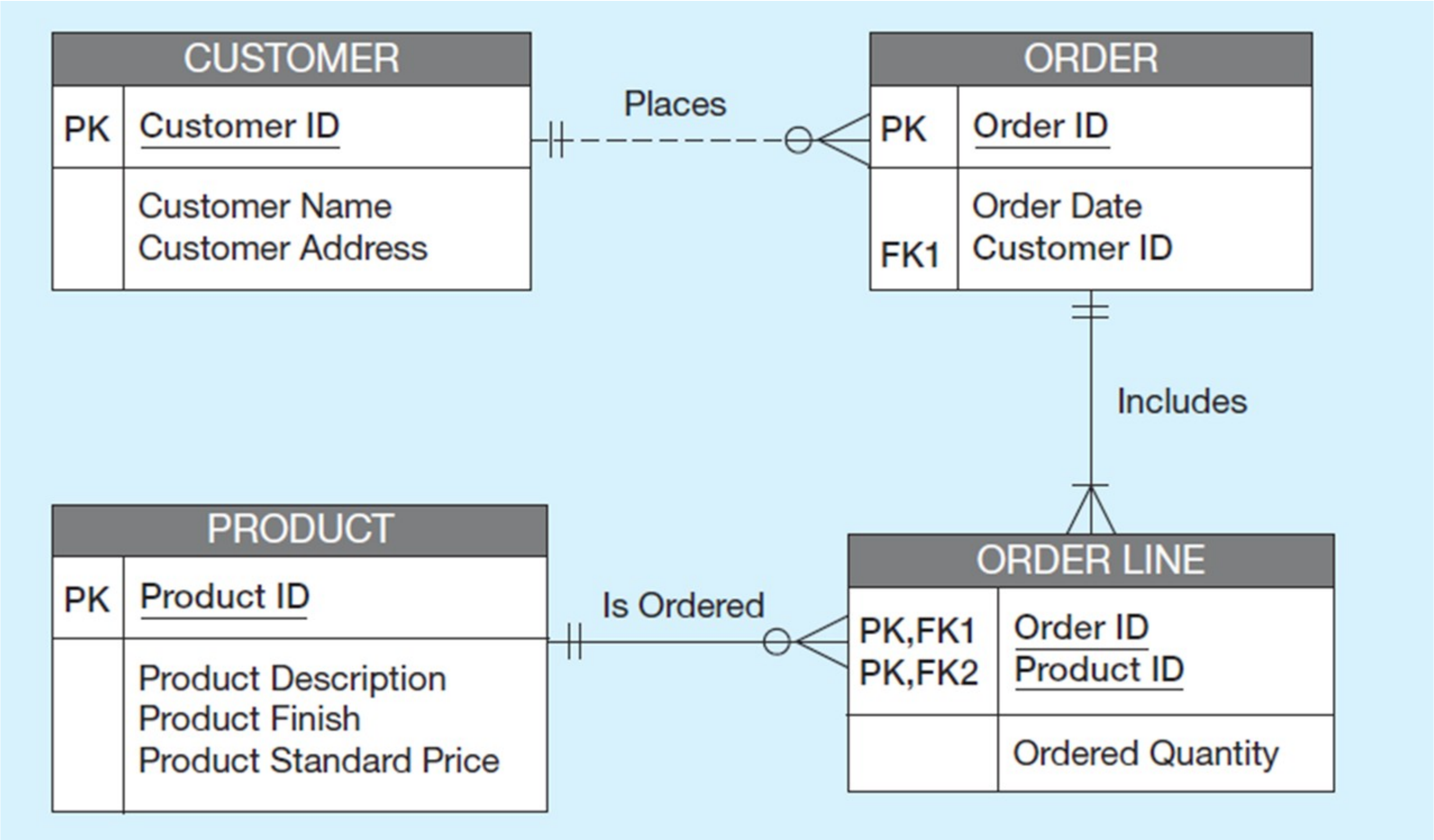
Figure 4-29 Removing Transitive Dependencies



Getting it into Third Normal Form

Transitive dependencies are removed.

Figure 4-30 Relational Schema for INVOICE Data (Microsoft Visio Notation)



Merging Relations

- View Integration – Combining entities from multiple E-R models into common relations
- Issues to watch out for when merging entities from different E-R models:
  - Synonyms – two or more attributes with different names but same meaning
  - Homonyms – attributes with same name but different meanings
  - Transitive dependencies – even if relations are in 3NF prior to merging, they may not be after merging
  - Supertype/subtype relationships – may be hidden prior to merging

Pearson

Figure 4-31 Enterprise Keys

a) Relations with enterprise key

OBJECT (<u>OID</u>, ObjectType)
EMPLOYEE (<u>OID</u>, EmpID, EmpName, DeptName, Salary)
CUSTOMER (<u>OID</u>, CustID, CustName, Address)

b) Sample data with enterprise key

OBJECT

<u>OID</u>	ObjectType
1	EMPLOYEE
2	CUSTOMER
3	CUSTOMER
4	EMPLOYEE
5	EMPLOYEE
6	CUSTOMER
7	CUSTOMER

EMPLOYEE

<u>OID</u>	EmpID	EmpName	DeptName	Salary
1	100	Jennings, Fred	Marketing	50000
4	101	Hopkins, Dan	Purchasing	45000
5	102	Huber, Ike	Accounting	45000

CUSTOMER

<u>OID</u>	CustID	CustName	Address
2	100	Fred's Warehouse	Greensboro, NC
3	101	Bargain Bonanza	Moscow, ID
6	102	Jasper's	Tallahassee, FL
7	103	Desks 'R Us	Kettering, OH

Copyright