Clemson *Means*

BUSINESS

Chapter 3
Relational Model

Instructor: He Li

Department of MANAGEMENT Clemson University

Components of Relational Model

- **Data Structure**
  (*tables, rows, columns*)
- **Data Integrity**
  (*specify business rules to maintain data integrity*)
- **Data Manipulation**
  (*operations to manipulate data, typically using SQL language*)

RELATION – *two-dimensional (columns and rows) table of data*

EMPLOYEE1

EmpID	Name	DeptName	Salary
100	Margaret Simpson	Marketing	48,000
140	Allen Beeton	Accounting	52,000
110	Chris Lucero	Info Systems	43,000
190	Lorenzo Davis	Finance	55,000
150	Susan Martin	Marketing	42,000

EMPLOYEE1(EmpID, Name, DeptName, Salary)

Relation → **Table**

Each row needs to have a primary key

E-R MODEL	RELATIONAL MODEL
Entity Types M:M Relationship Types	**Relations (Tables)**
Entity Instances M:M Relationship Instances	**Rows**
Attributes	**Columns**
Identifier	**Primary Key** (*identifiers of the relation*) **Foreign Key** (*identifiers that enable a dependent relation to refer to its parent relation*) e.g., EMPLOYEE1(<u>EmpID</u>, Name, <u>DeptName</u>) DEPARTMENT(<u>DeptName</u>, Location, Fax)

INTEGRITY CONSTRAINTS

Domain Constraints

Allowable values for an attributes (i.e., data types of value restrictions)

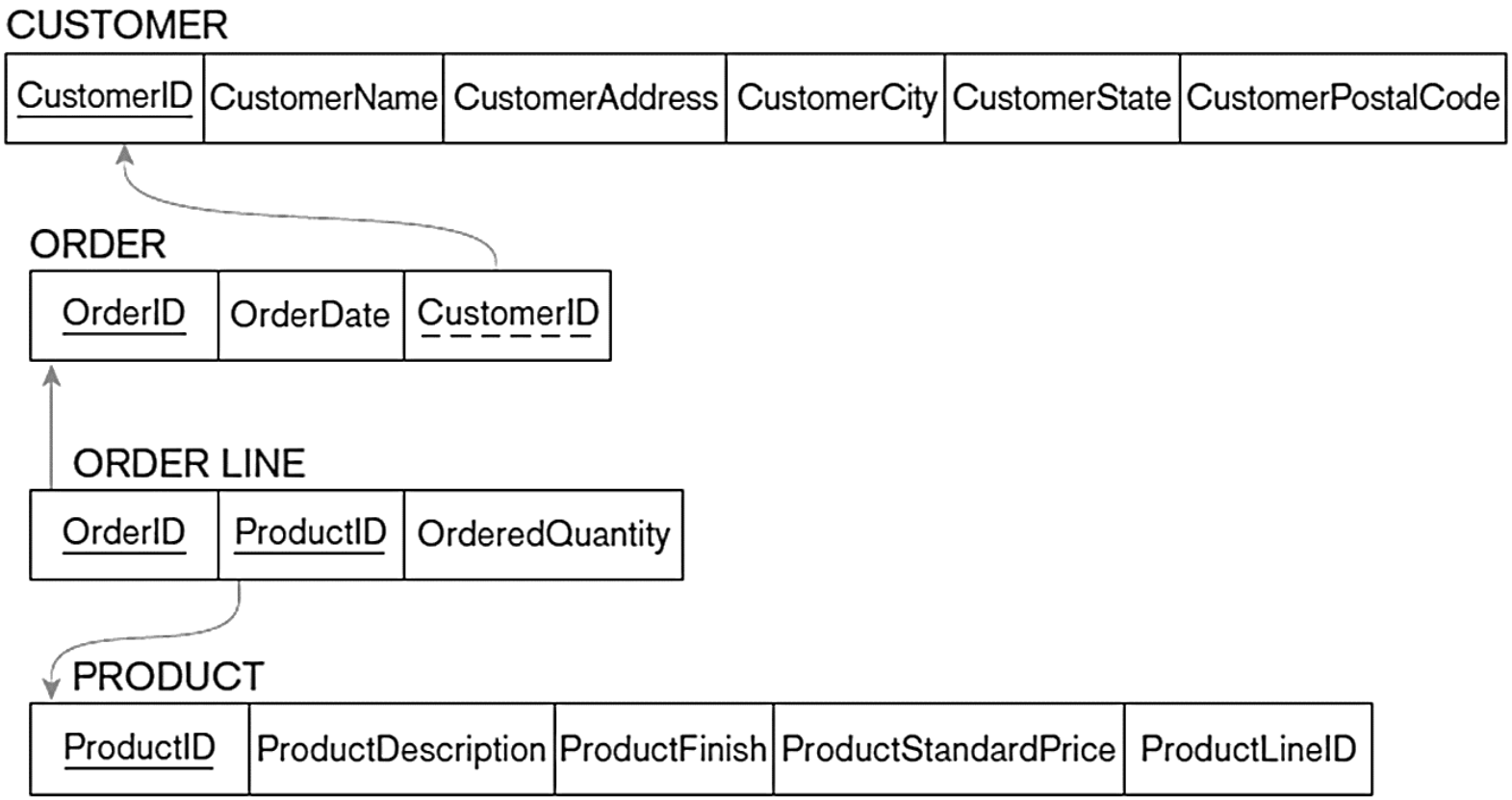
Entity Integrity

Primary key is NOT NULL

Referential Integrity

Foreign key MUST match a primary key OR NULL

Example of Referential Integrity Constraints

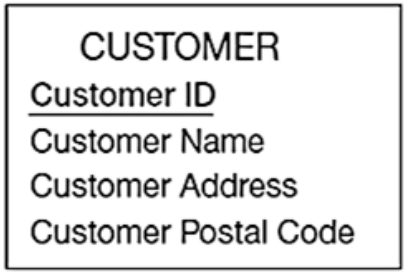


Converting EER to Relations

1. Mapping Regular Entities to Relations
   - 1.1 Simple attributes
   - 1.2 Composite attributes
   - 1.3 Multivalued attributes
2. Mapping Weak Entities to Relations
3. Mapping Binary Relationships to Relations
   - 3.1 One-to-Many
   - 3.2 Many-to-Many
   - 3.3 One-to-One
4. Mapping Associative Entities to Relations
   - 4.1 Identifier not assigned
   - 4.2 Identifier assigned
5. Mapping Unary Relationships to Relations
   - 5.1 One-to-Many
   - 5.2 Many-to-Many
6. Mapping Ternary (and n-ary) Relationships to Relations

Step 1.1: Regular Entity with Simple Attributes

Rule of Thumb: E-R attributes map directly onto the relation



CUSTOMER

CustomerID	CustomerName	CustomerAddress	CustomerPostalCode

Step 1.2: Regular Entity with a Composite Attribute

Rule of Thumb: Use only their simple, component attributes

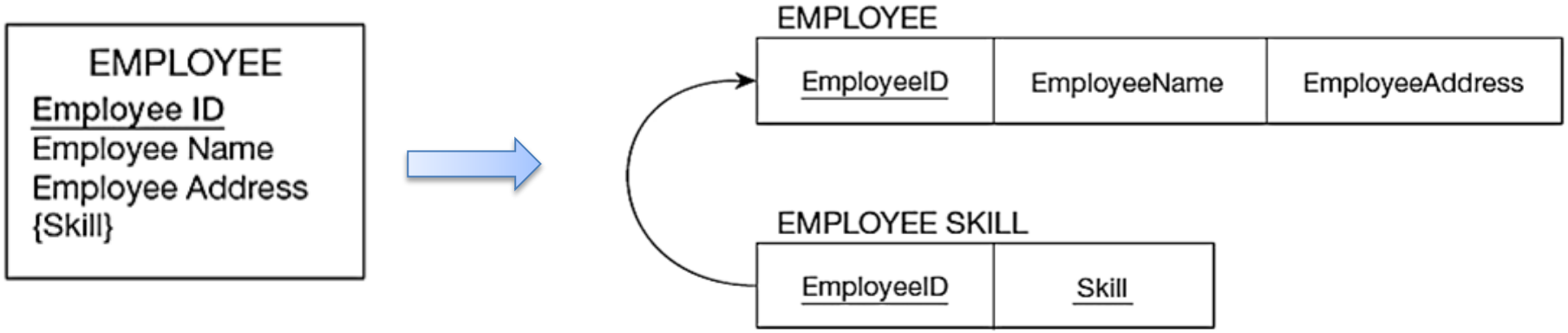
CUSTOMER
Customer ID
Customer Name
Customer Address
(Customer Street, Customer City, Customer State)
Customer Postal Code

CUSTOMER

CustomerID	CustomerName	CustomerStreet	CustomerCity	CustomerState	CustomerPostalCode

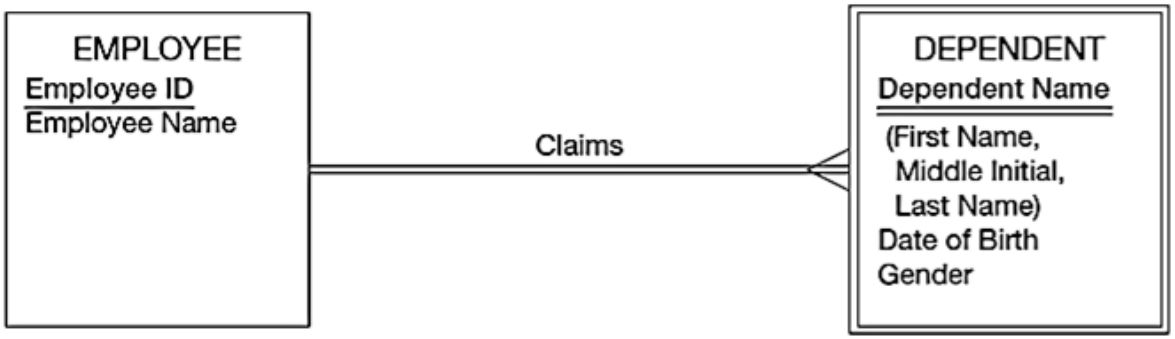
Step 1.3: Regular Entity with a Multivalued Attribute

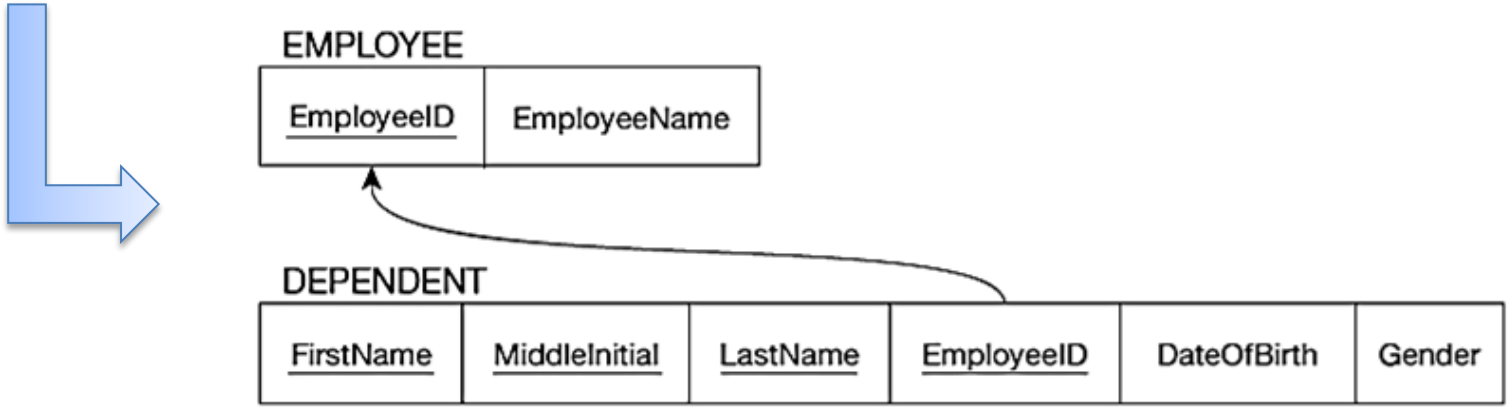
Rule of Thumb: Become a separate relation with a foreign key taken from the superior entity



Step 2: Weak Entities

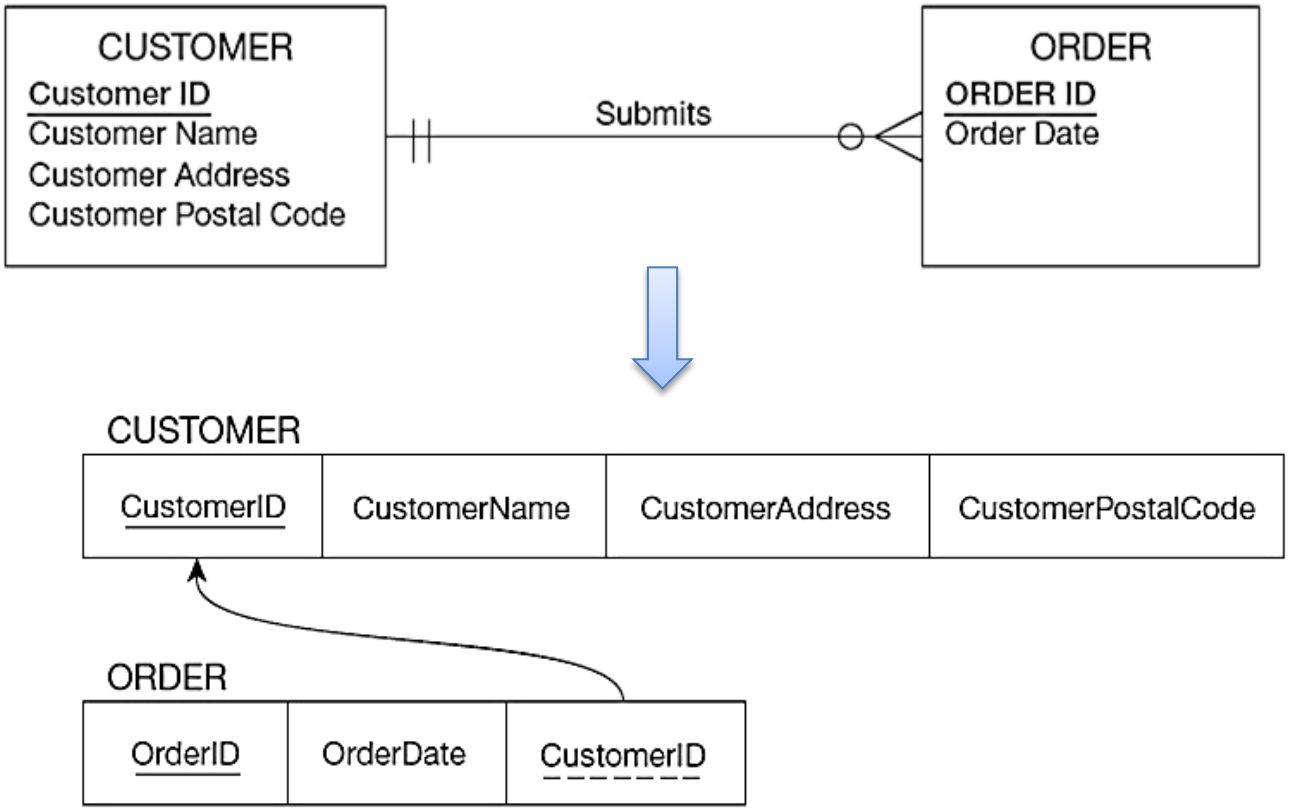
Rule of Thumb: (1) Becomes a separate relation with a foreign key taken from the superior entity; and (2) Primary key composed of: partial identifier of weak entity and primary key of identifying relation (strong entity)





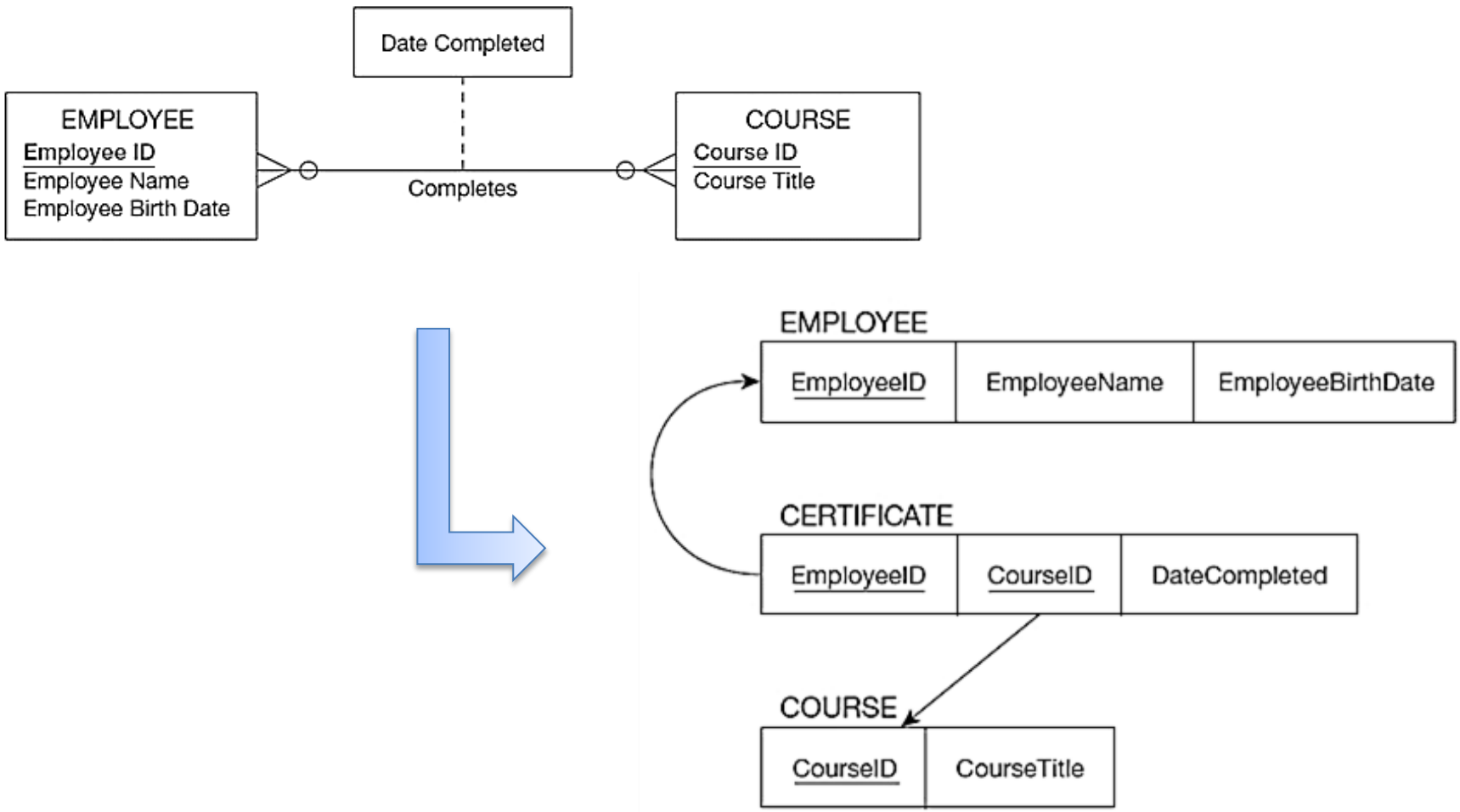
Step 3.1: Binary One-to-Many Relationship

Rule of Thumb: Primary key on the one side becomes a foreign key on the many side



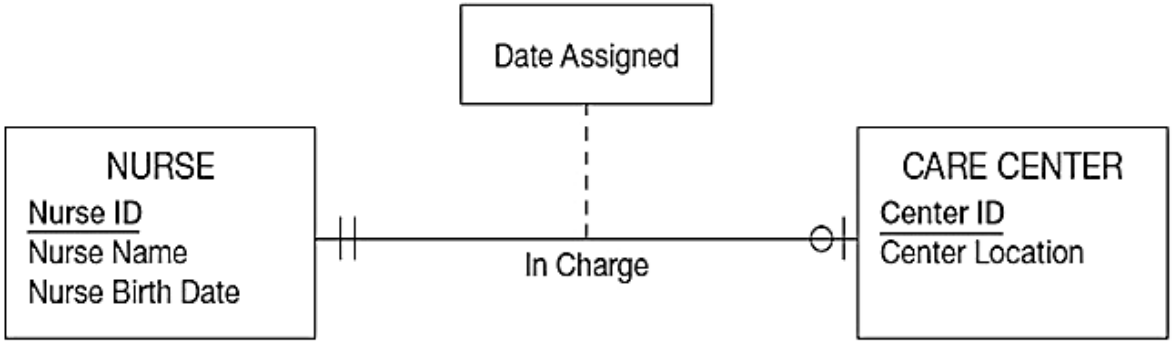
Step 3.2: Binary Many-to-Many Relationship

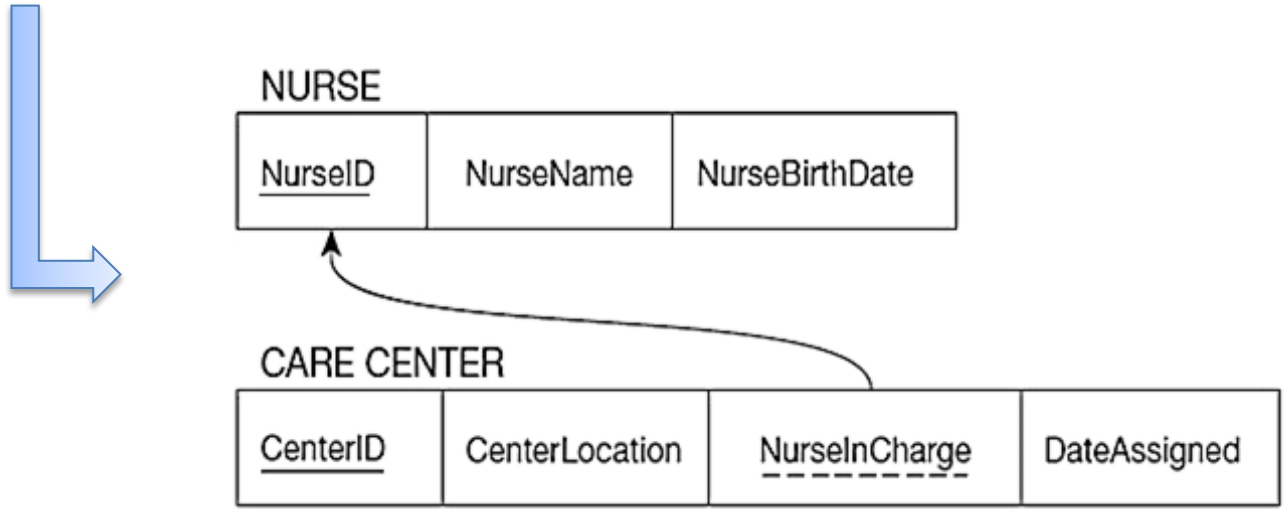
Rule of Thumb: Create a **new relation** with the primary keys of the two entities as its primary key



Step 3.3: Binary One-to-One Relationship

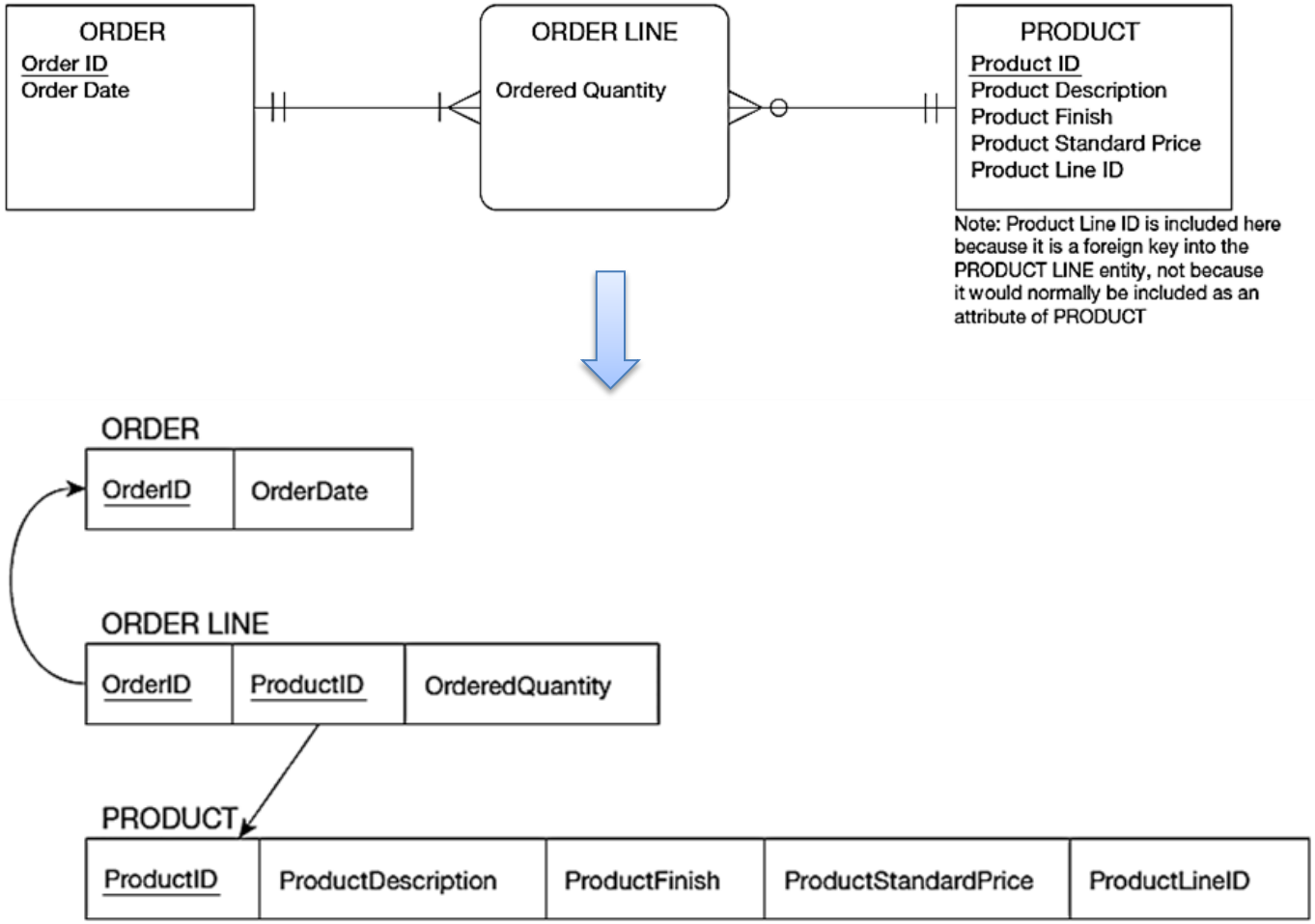
Rule of Thumb: Primary key on mandatory side becomes a foreign key on optional side





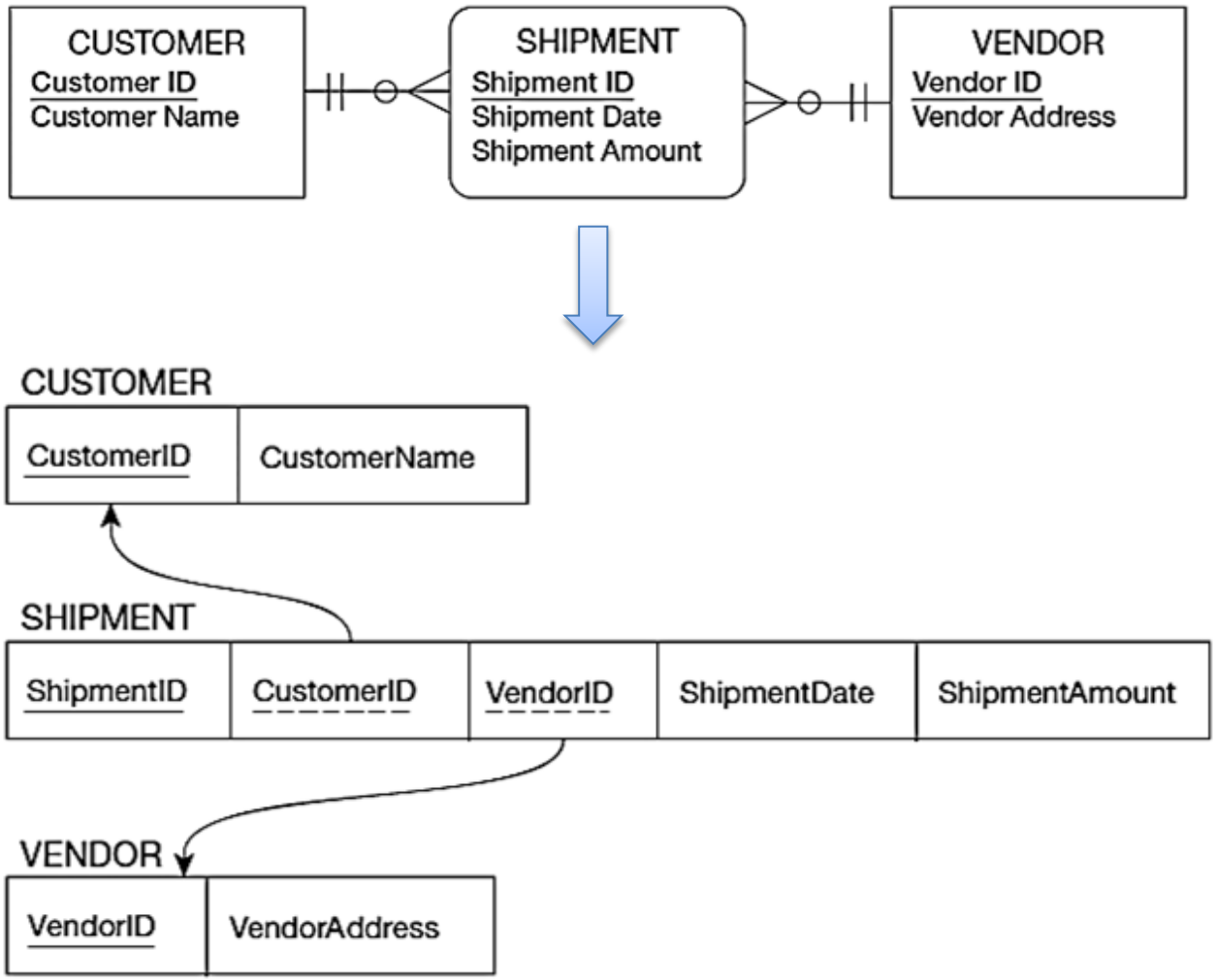
Step 4.1: Associative Entity (Identifier Not Assigned)

Rule of Thumb: Default primary key for the association relation is composed of the primary keys of the two entities (as in M:N relationship)



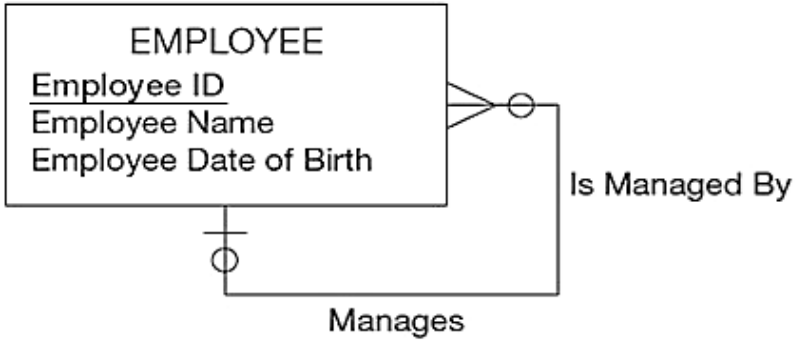
Step 4.2: Associative Entity (Identifier Assigned)

Rule of Thumb:
Use primary keys of associated entities as foreign keys in the associative entity.



Step 5.1: Unary One-to-Many Relationship

Rule of Thumb: Recursive foreign key in the same relation

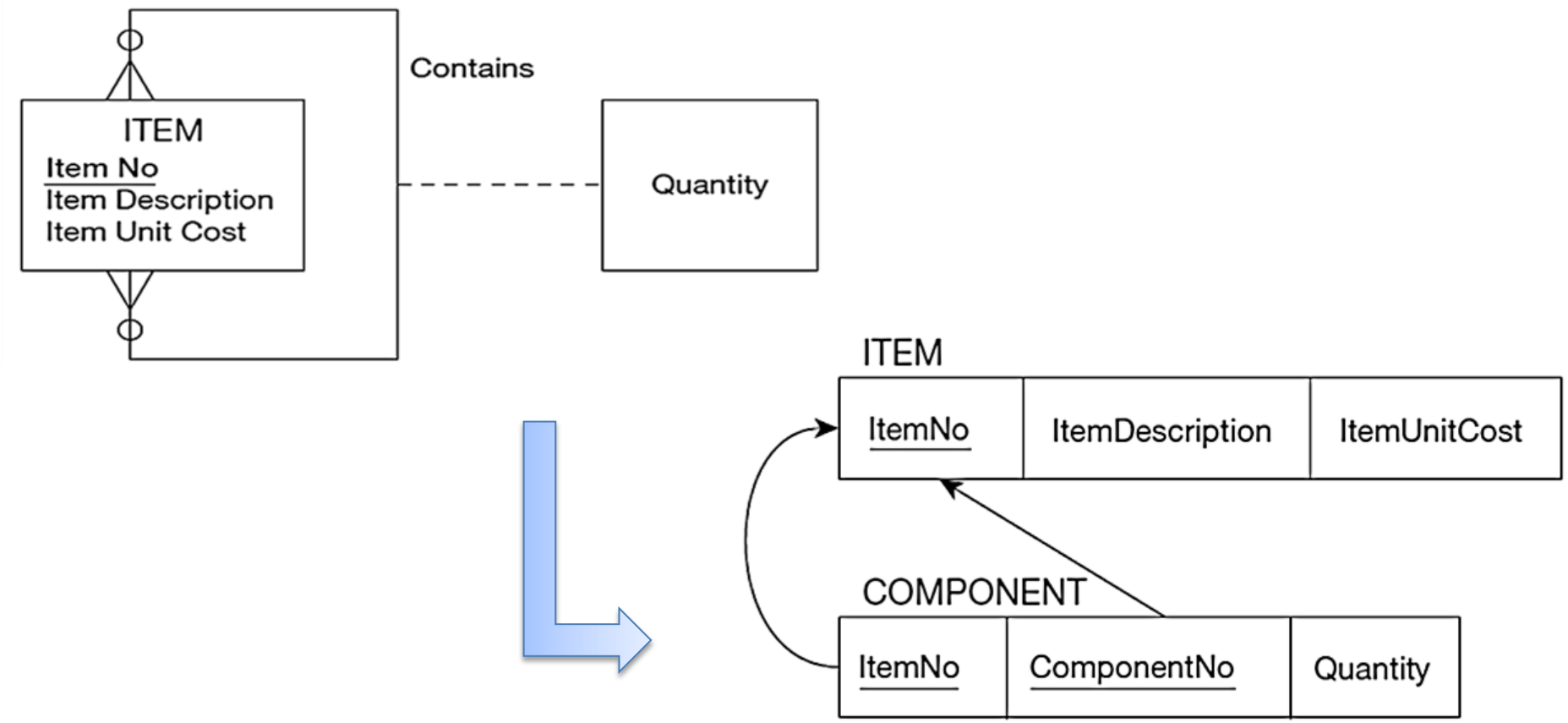


EMPLOYEE

EmployeeID	EmployeeName	EmployeeDateOfBirth	ManagerID

Step 5.2: Unary Many-to-Many Relationship

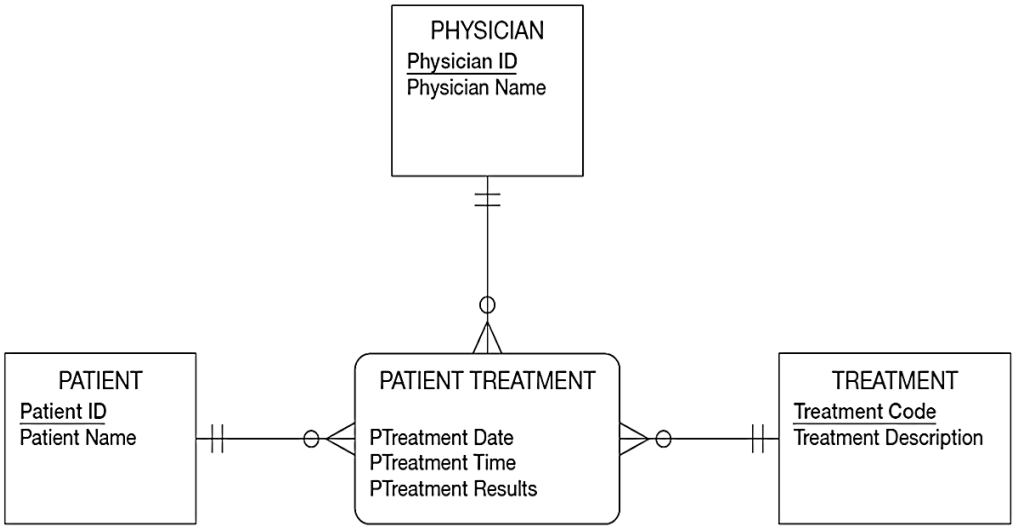
Rule of Thumb: Use two relations: (1) one for the entity type; and (2) one for an associative relation in which the primary key has two attributes, both taken from the primary key of the entity

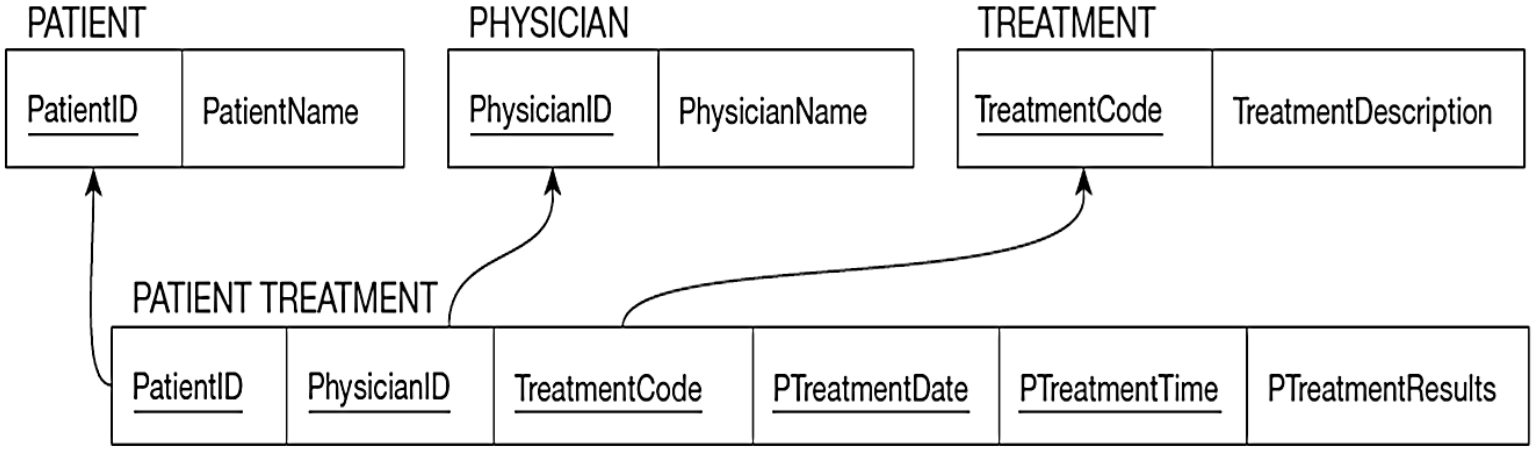


Step 6: Ternary Relationship

Rule of Thumb:

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





Summary: Converting EER to Relations

Department of MANAGEMENT Clemson University

EER Structure	Relational Representation (Sample Figure)
Regular entity	Create a relation with primary key and nonkey attributes
Composite attribute	Each component of a composite attribute becomes a separate attribute in the target relation
Multivalued attribute	Create a separate relation for multivalued attribute with composite primary key, including the primary key of the entity
Weak entity	Create a relation with a composite primary key (which includes the primary key of the entity on which this entity depends) and nonkey attributes
Binary or unary 1:*M* relationship	Place the primary key of the entity on the one side of the relationship as a foreign key in the relation for the entity on the many side
Binary or unary *M*:*N* relationship or associative entity without its own key	Create a relation with a composite primary key using the primary keys of the related entities plus any nonkey attributes of the relationship or associative entity
Binary or unary 1:1 relationship	Place the primary key of either entity in the relation for the other entity; if one side of the relationship is optional, place the foreign key of the entity on the mandatory side in the relation for the entity on the optional side
Binary or unary *M*:*N* relationship or associative entity with its own key	Create a relation with the primary key associated with the associative entity plus any nonkey attributes of the associative entity and the primary keys of the related entities as foreign keys
Ternary and *n*-ary relationships	Same as binary *M*:*N* relationships above; without its own key, include as part of primary key of relation for the relationship or associative entity the primary keys from all related entities; with its own surrogate key, the primary keys of the associated entities are included as foreign keys in the relation for the relationship or associative entity
Supertype/subtype relationship	Create a relation for the superclass, which contains the primary and all nonkey attributes in common with all subclasses, plus create a separate relation for each subclass with the same primary key (with the same or local name) but with only the nonkey attributes related to that subclass

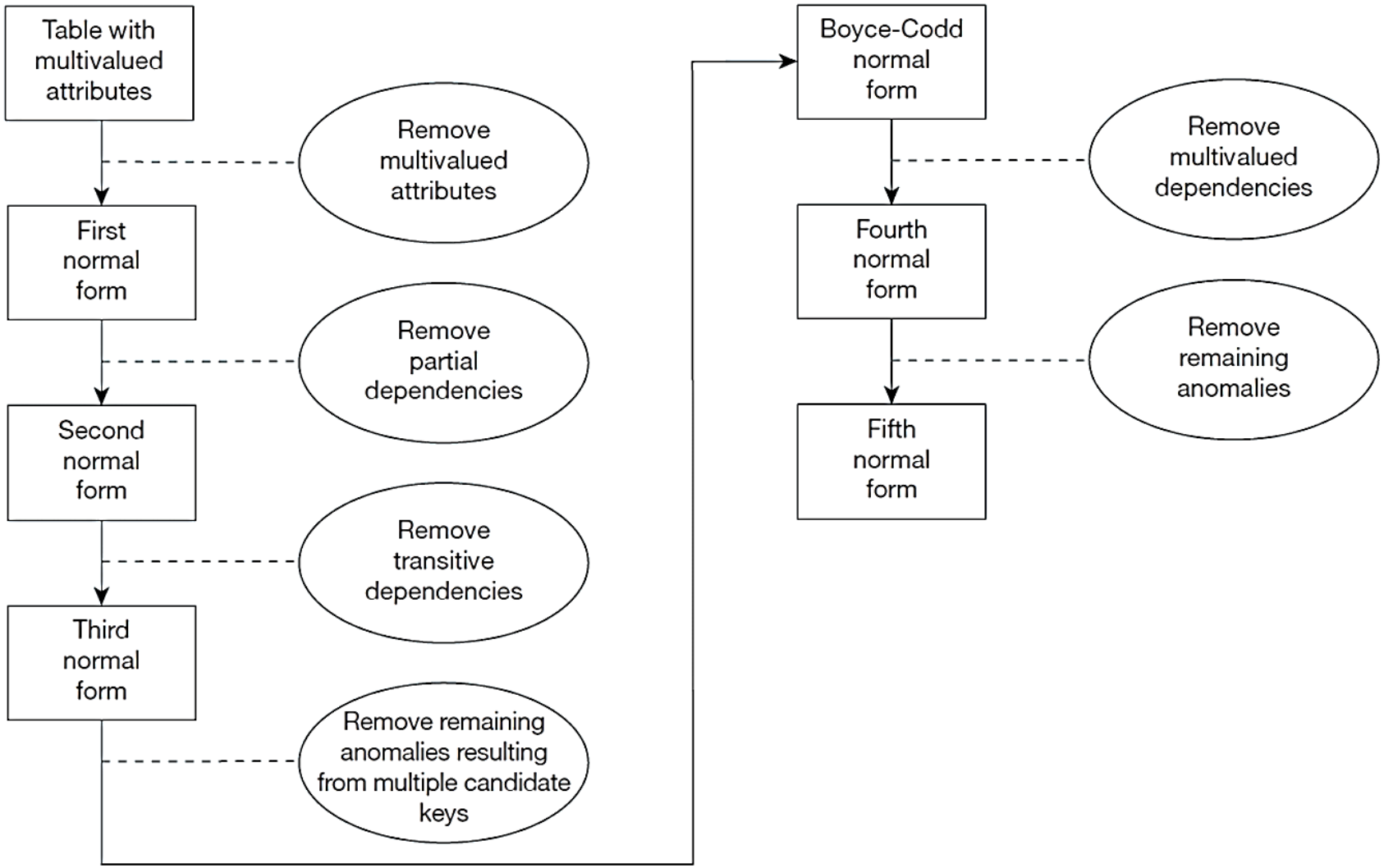
Normalization – *a tool to validate and improve a logical design*

Department of MANAGEMENT Clemson University

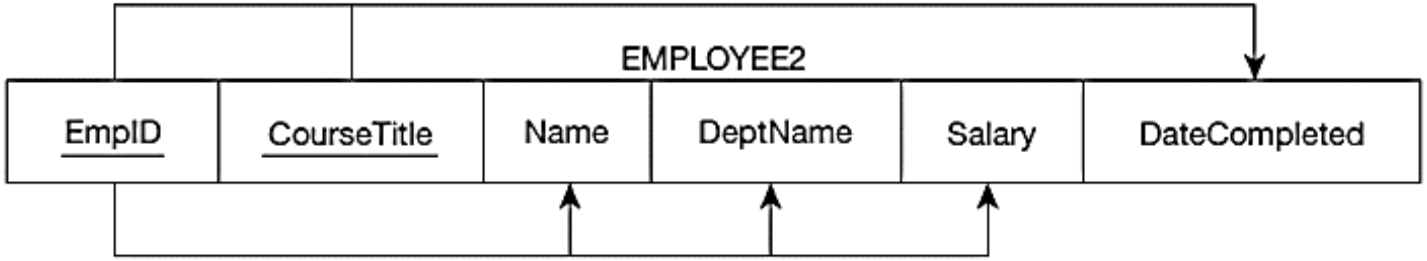
GOAL ***produce well-structured relations***

- **Contain minimal data redundancy**
- **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

Steps in Normalization



Functional Dependencies – The value of one attribute determines the value of another attribute.



Two Types of Functional Dependencies:

1. EmpID → Name, DeptName, Salary
2. EmpID, CourseTitle → DateCompleted

Candidate Key = a unique identifier

- One of the candidate keys will become the primary key.
- Each non-key field is functionally dependent on every candidate key.

Step 1.1: Converting to First Normal Form (1NF) – remove repeating groups

OrderID	Order Date	Customer ID	Customer Name	Customer Address	ProductID	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

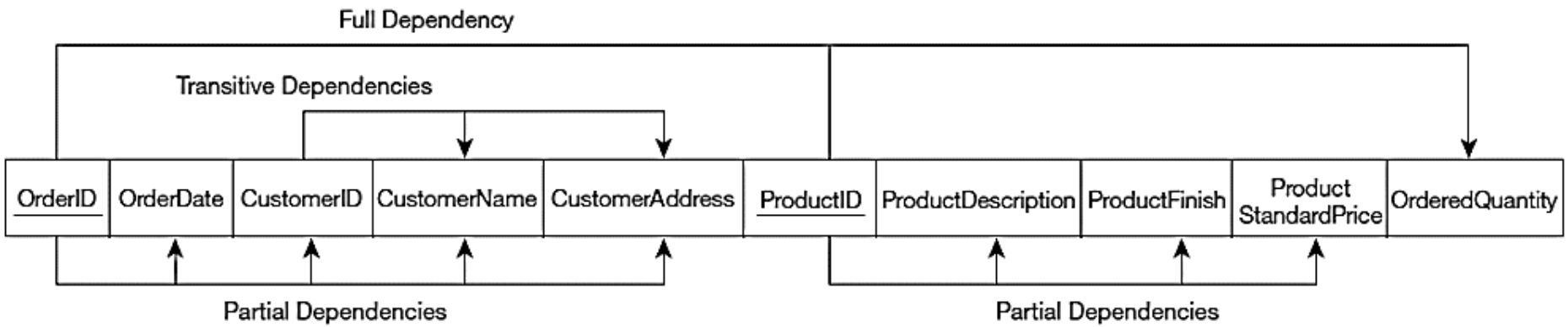
Remove repeating groups

OrderID	Order Date	Customer ID	Customer Name	Customer Address	ProductID	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
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

Step 1.2: Converting to First Normal Form (1NF) – identify primary keys

In this example, there are four determinants and functions dependencies:

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

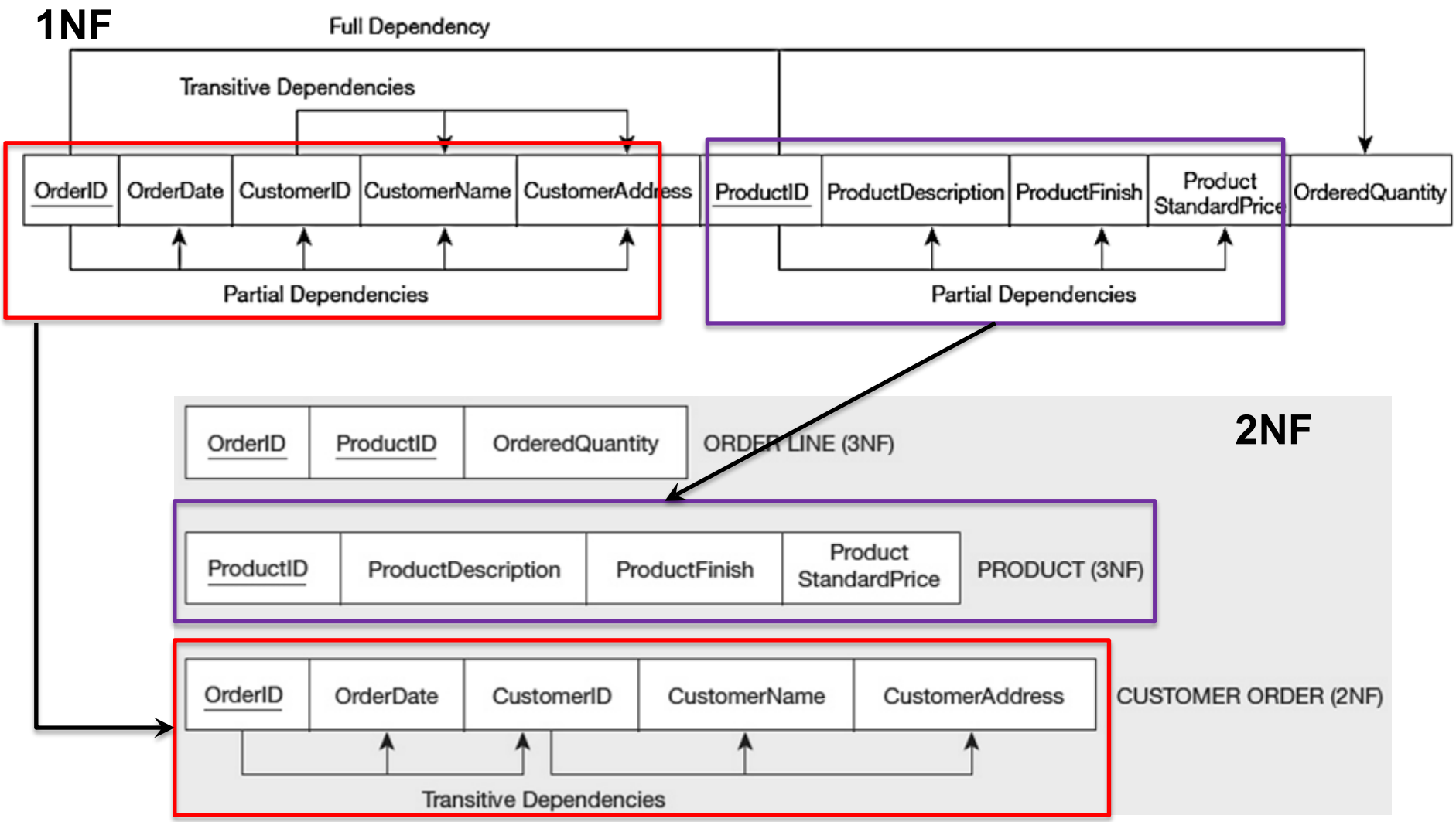


Step 2: Converting to Second Normal Form (2NF)

- 1NF + **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

- To convert a relation with partial dependencies to second normal form, the following steps are required:
  1. Create a new relation for each primary key attribute (or combination of attributes) that is a determinant in a partial dependency. That attribute is the primary key in the new relation.
  2. Move the non-key attributes that are only dependent on this primary key attribute (or attributes) from the old relation to the new relation.

Step 2: Converting to Second Normal Form (2NF)



Step 3: Converting to Third Normal Form (3NF)

- 2NF + **no transitive dependencies** (functional dependencies on non-primary-key attributes)
- **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