# Modern Systems Analysis and Design

Fourth Edition

Jeffrey A. Hoffer
Joey F. George
Joseph S. Valacich

## Structuring System Data Requirements

# Learning Objectives

- ✓ Define key data modeling terms.
- ✓ Draw entity-relationship (E-R) and class diagrams to represent common business situations.
- ✓ Explain the role of conceptual data modeling in IS analysis and design.
- ✓ Distinguish between unary, binary, and ternary relationships.
- ✓ Define four types of business rules.
- ✓ Compare the capabilities of class diagrams vs. E-R diagrams.
- ✓ Relate data modeling to process and logic modeling.





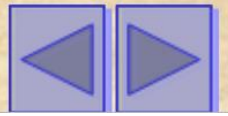


**Figure 9-1** Systems development life cycle with analysis phase highlighted

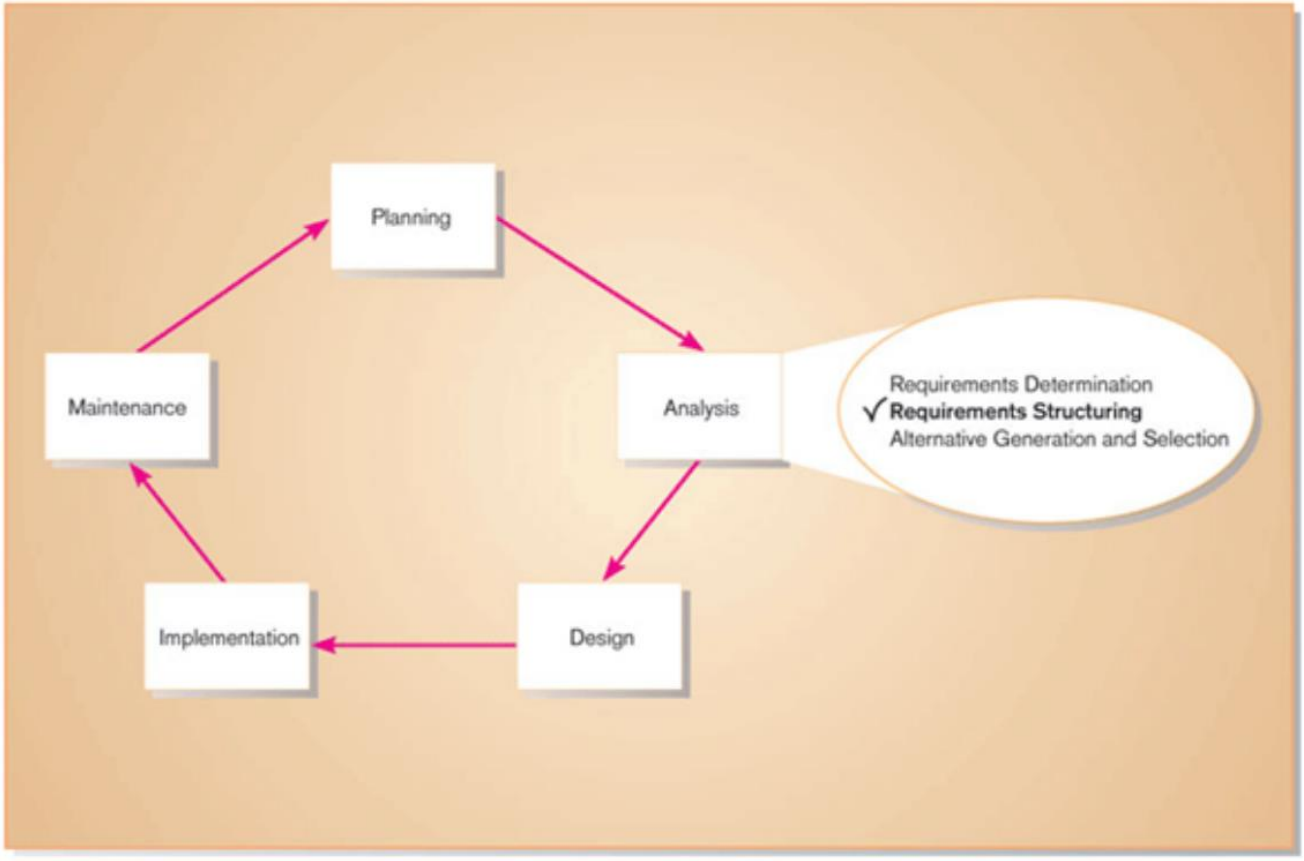
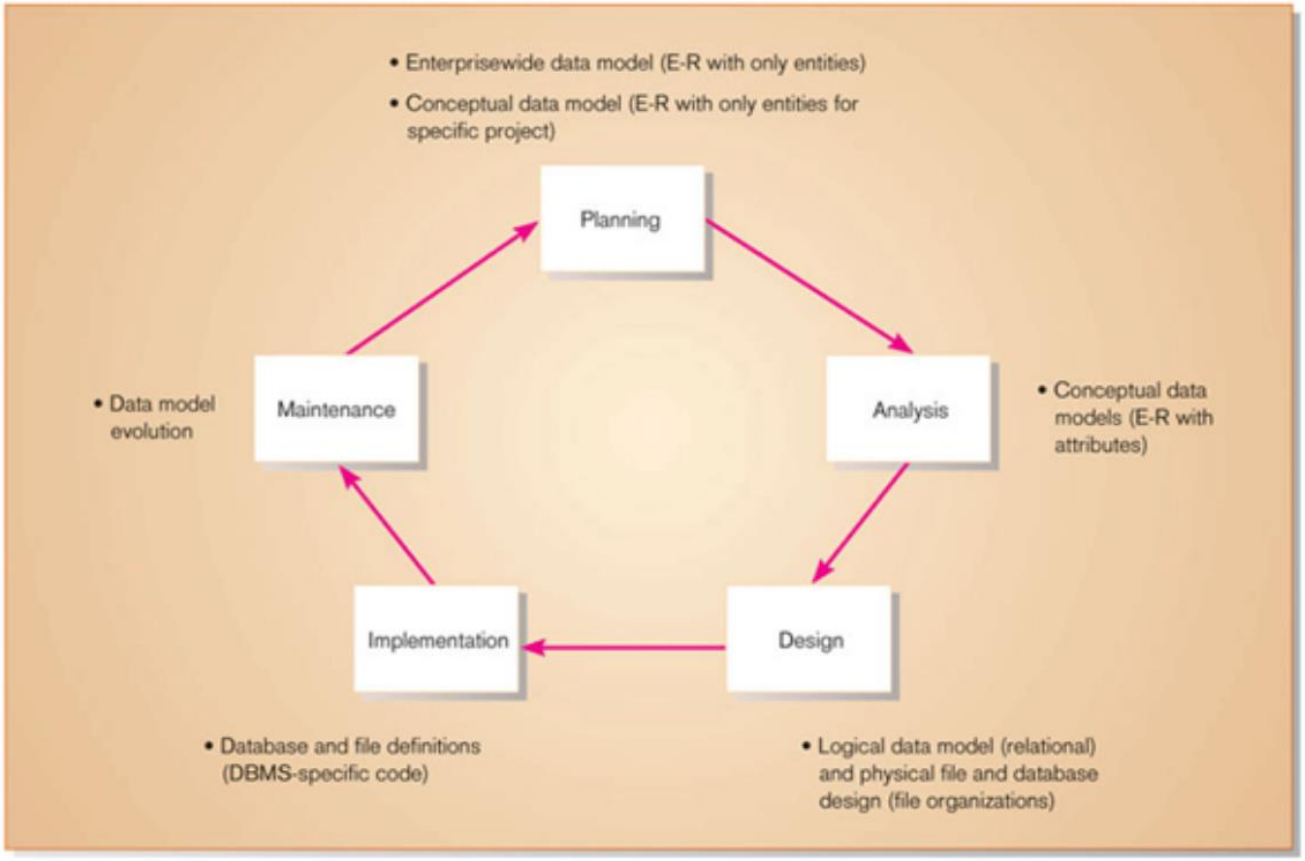


**Figure 9-2** Relationship between data modeling and the systems development life cycle

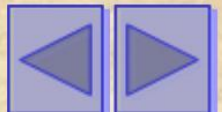


# Conceptual Data Modeling

- A detailed model that captures the overall structure of data in an organization
- Independent of any database management system (DBMS) or other implementation considerations





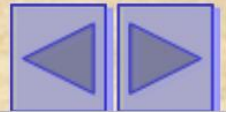


# Process of Conceptual Data Modeling

- Develop a data model for the current system
- Develop a new conceptual data model that includes all requirements of the new system
- In the design stage, the conceptual data model is translated into a physical design
- Project repository links all design and data modeling steps performed during SDLC





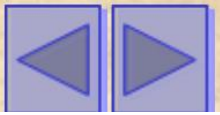


# Deliverables and Outcome

- Primary deliverable is an entity-relationship (E-R) diagram or class diagram
- As many as 4 E-R or class diagrams are produced and analyzed
  - E-R diagram that covers data needed in the project's application
  - E-R diagram for the application being replaced
  - E-R diagram for the whole database from which the new application's data are extracted
  - E-R diagram for the whole database from which data for the application system being replaced is drawn







# Deliverables and Outcome (cont.)

- Second deliverable is a set of entries about data objects to be stored in repository or project dictionary.
  - Repository links data, process, and logic models of an information system.
  - Data elements included in the DFD must appear in the data model and vice versa.
  - Each data store in a process model must relate to business objects represented in the data model.



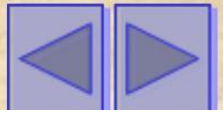
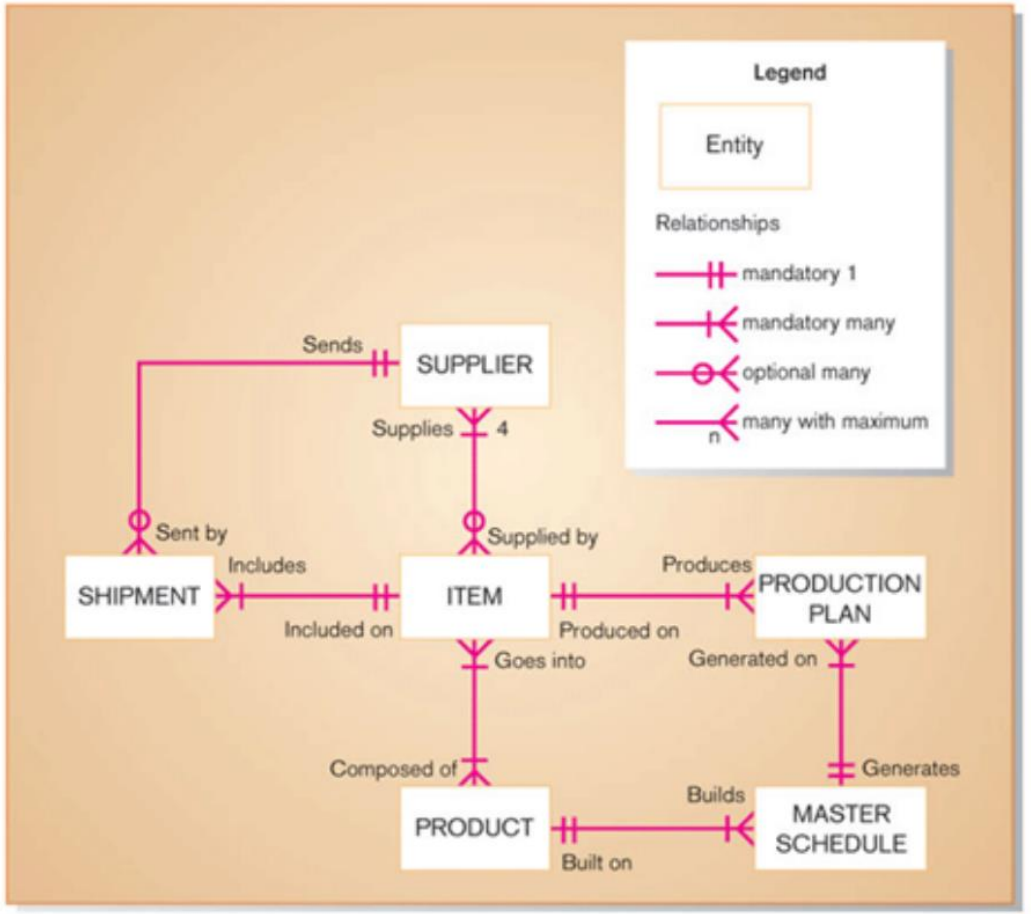



**Figure 9-3** Sample conceptual data model

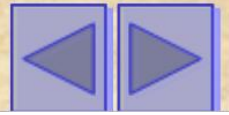


# Gathering Information for Conceptual Data Modeling

- Two perspectives
  - Top-down
    - Data model is derived from an intimate understanding of the business.
  - Bottom-up
    - Data model is derived by reviewing specifications and business documents.





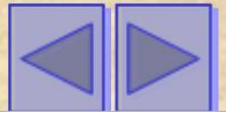


# Requirements Determination Questions for Data Modeling

- What are subjects/objects of the business?
  - ➔ Data entities and descriptions
- What unique characteristics distinguish between subjects/objects of the same type?
  - ➔ Primary keys
- What characteristics describe each subject/object?
  - ➔ Attributes and secondary keys
- How do you use the data?
  - ➔ Security controls and user access privileges





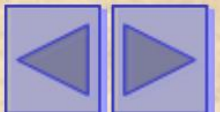


# Requirements Determination Questions for Data Modeling (cont.)

- Over what period of time are you interested in the data?
  - ➔ Cardinality and time dimensions
- Are all instances of each object the same?
  - ➔ Supertypes, subtypes, and aggregations
- What events occur that imply associations between objects?
  - ➔ Relationships and cardinalities
- Are there special circumstances that affect the way events are handled?
  - ➔ Integrity rules, cardinalities, time dimensions





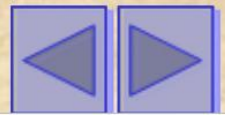


# Introduction to Entity-Relationship (E-R) Modeling

- Entity-Relationship (E-R) Diagram
  - A detailed, logical representation of the entities, associations and data elements for an organization or business
- Notation uses three main constructs
  - Data entities
  - Relationships
  - Attributes







Association between the instances of one or more entity types


Person, place, object, event or concept about which data is to be maintained

**Entity type**: collection of entities with common characteristics

**Entity instance**: single entity

named property or characteristic of an entity

# Identifier Attributes


- Candidate key
  - Attribute (or combination of attributes) that uniquely identifies each instance of an entity type
- Identifier
  - A candidate key that has been selected as the unique identifying characteristic for an entity type

# Identifier Attributes (cont.)



Identifier attribute

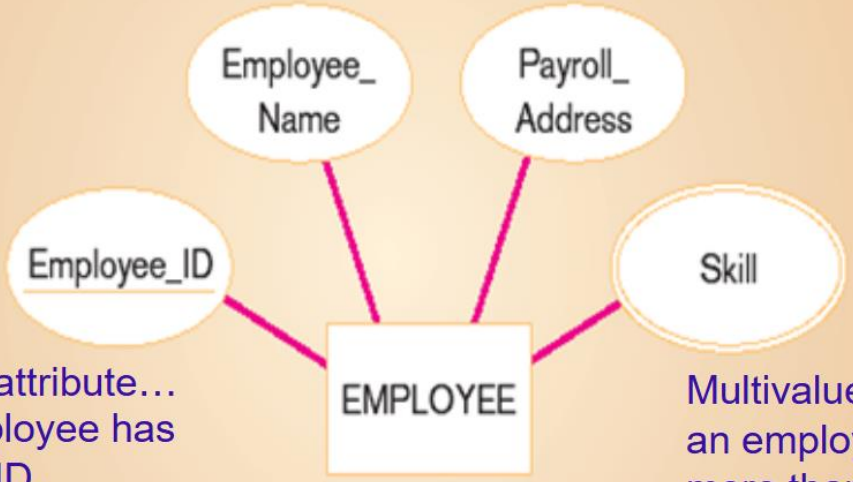
- Selection rules for an identifier
  1. Choose a candidate key that will not change its value.
  2. Choose a candidate key that will never be null.
  3. Avoid using intelligent keys.
  4. Consider substituting single value surrogate keys for large composite keys.

# Multivalued Attributes


- An attribute that may take on more than one value for each entity instance
- Represented on E-R Diagram in two ways:
  - double-lined ellipse
  - weak entity

# Entity and Attribute Example

Simple attributes




Identifier attribute… each employee has a unique ID.

Multivalued attribute… an employee may have more than one skill.

# Degree of Relationship

- Degree: number of entity types that participate in a relationship
- Three cases
  - **Unary:** between two instances of one entity type
  - **Binary:** between the instances of two entity types
  - **Ternary:** among the instances of three entity types

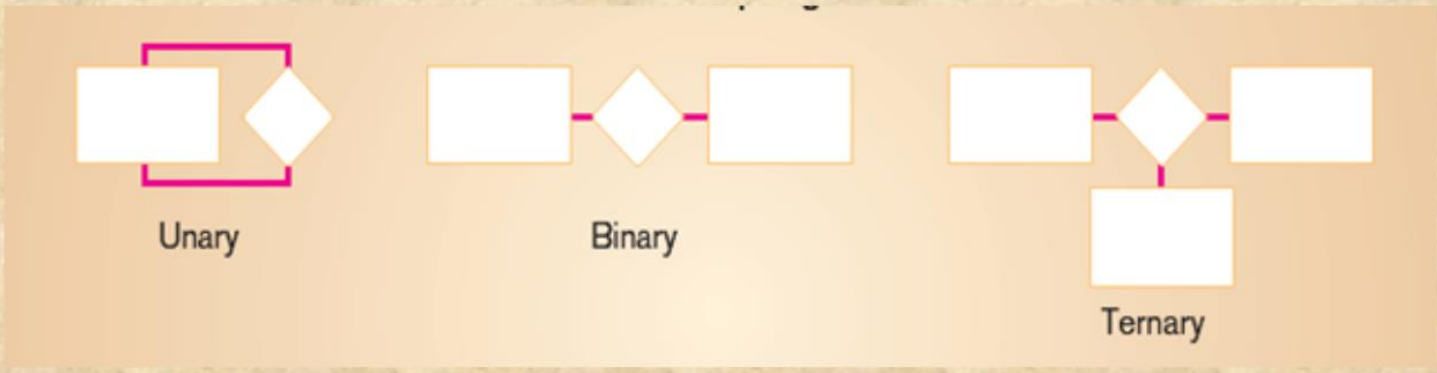
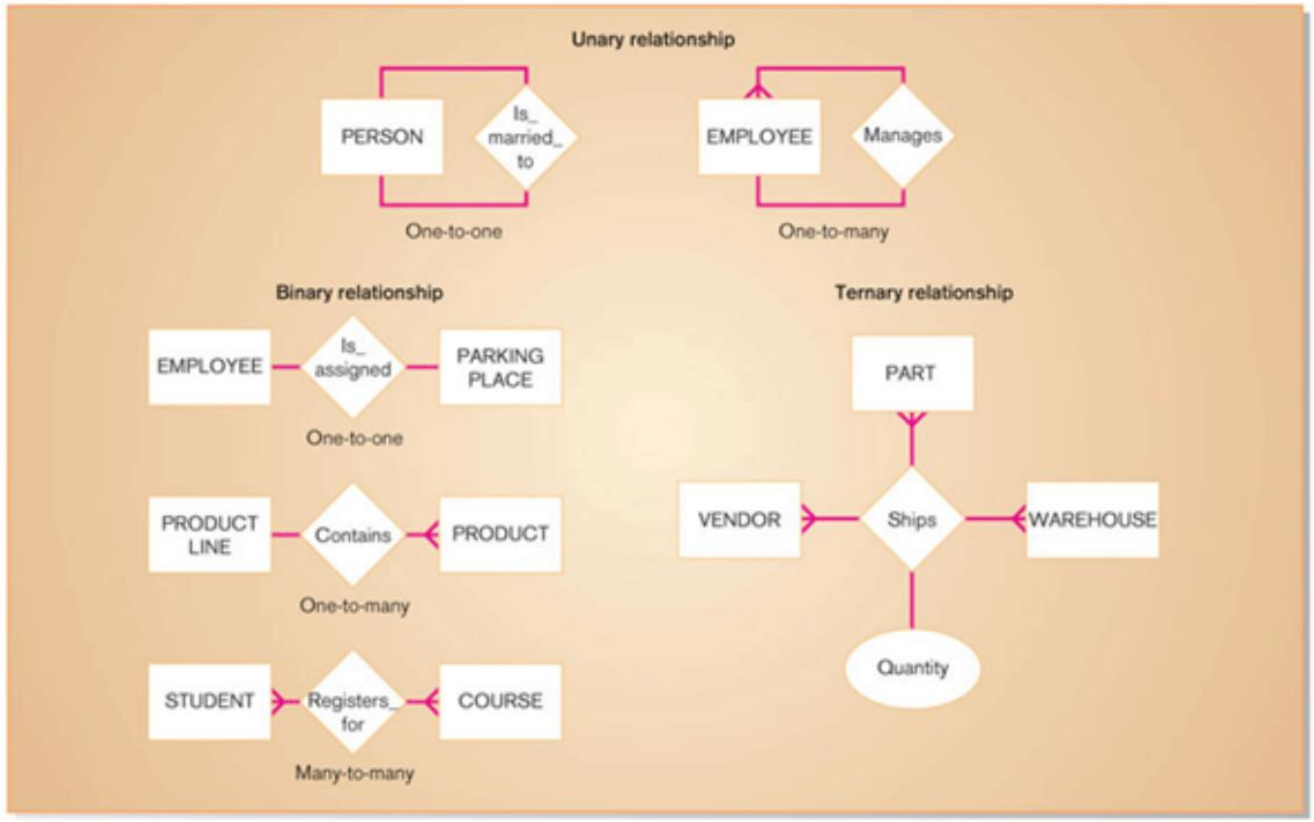


**Figure 9-6** Example relationships of different degrees

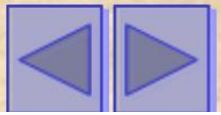


# Cardinality

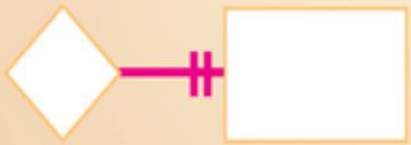
- The number of instances of entity B that can or must be associated with each instance of entity A
- Minimum Cardinality
  - The minimum number of instances of entity B that may be associated with each instance of entity A
- Maximum Cardinality
  - The maximum number of instances of entity B that may be associated with each instance of entity A
- Mandatory vs. Optional Cardinalities
  - Specifies whether an instance must exist or can be absent in the relationship







# Cardinality Symbols



Mandatory 1 cardinality

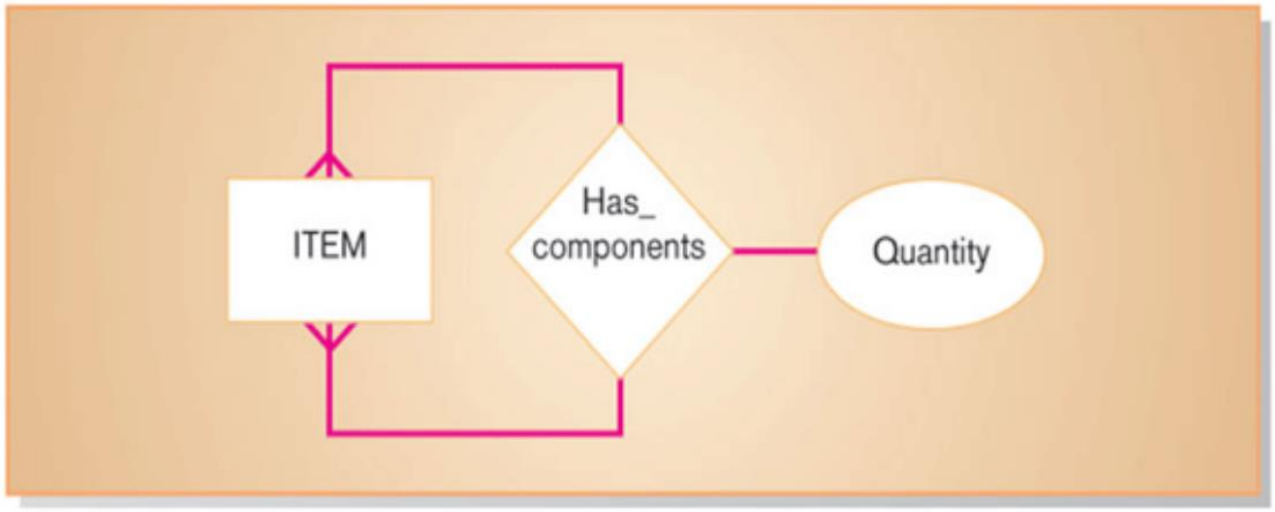

Mandatory many (M) cardinality (1, 2, ..., many)
(*n* is a number for an upper limit, if one exists)

Optional 0 or 1 cardinality


Optional zero-many cardinality (0, 1, 2, ..., many)

# Unary Relationship Example

**Figure 9-7a** Bill-of-materials unary relationship - Many-to-many relationship



# Binary Relationship Examples

**Figure 9-8a** Examples of cardinalities in relationships - Mandatory cardinalities

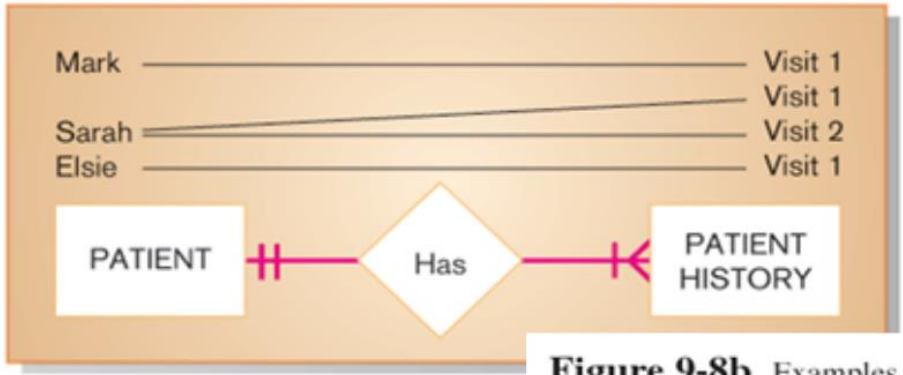
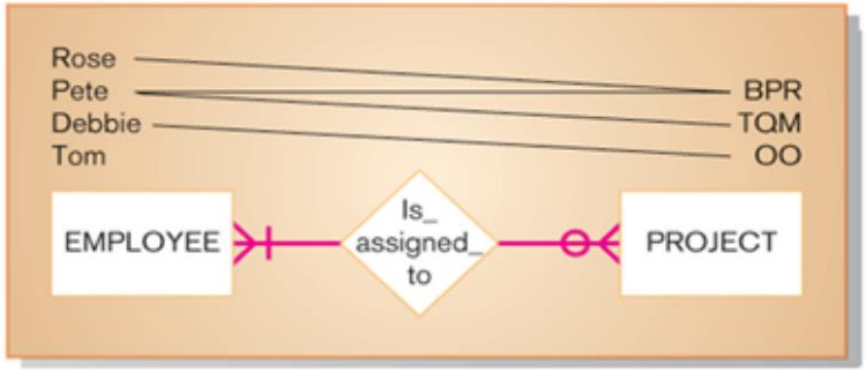


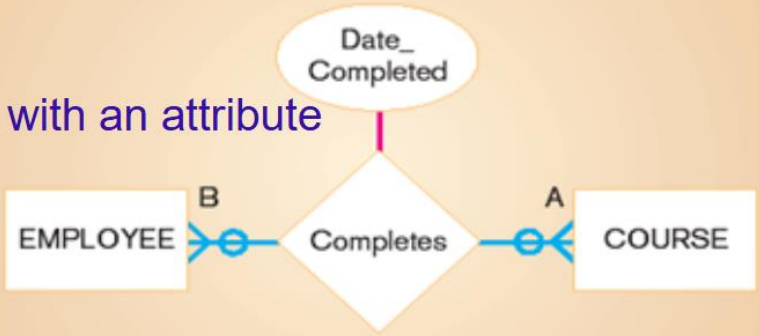

**Figure 9-8b** Examples of cardinalities in relationships - One optional, one mandatory cardinality



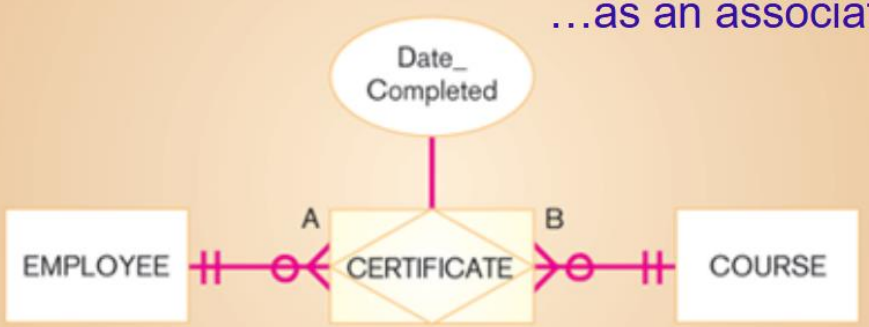
# Associative Entities

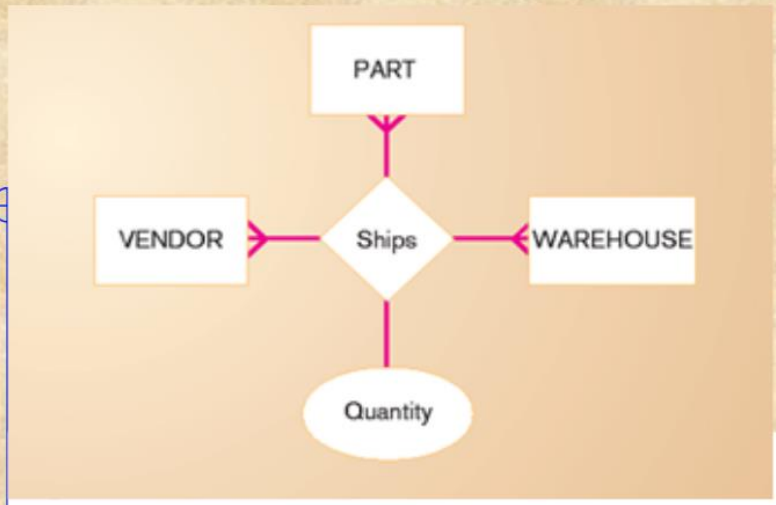

- An entity type that associates the instances of one or more entity types and contains attributes that are peculiar to the relationship between those entity instances
- An associative entity is:
  - An entity
  - A relationship
- This is the preferred way of illustrating a relationship with attributes

A relationship with an attribute

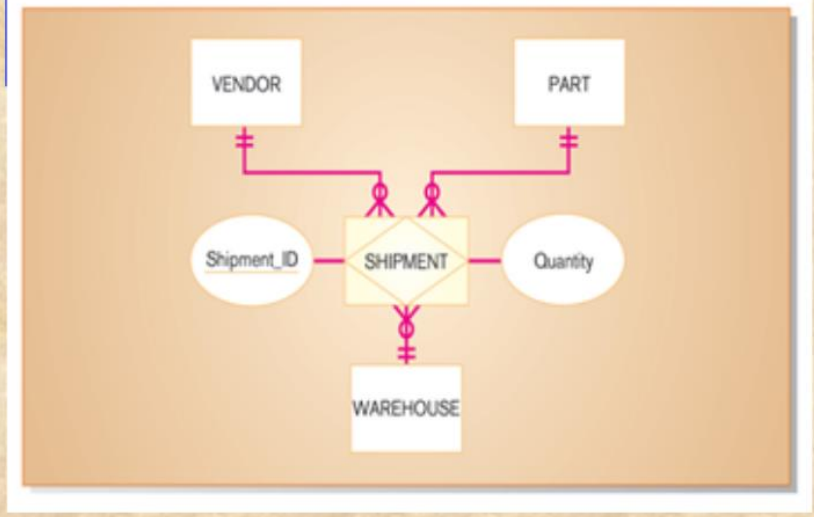



…as an associative entity

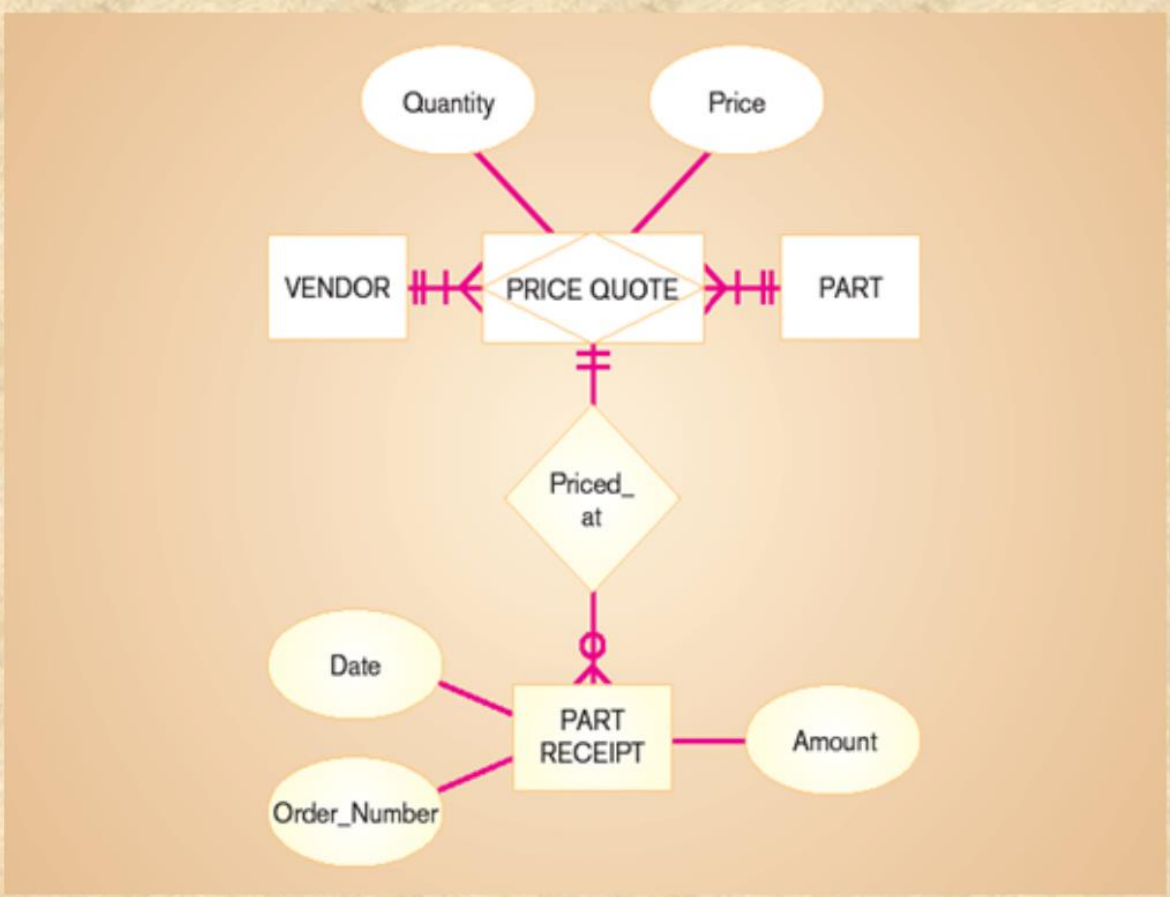





Ternary relationship




…as an associative entity



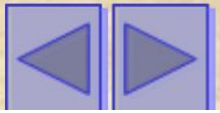

A relationship that itself is related to other entities via another relationship *must* be represented as an associative entity.

# Supertypes and Subtypes

- Subtype: a subrouping of the entities in an entity type that shares common attributes or relationships distinct from other subtypes
- Supertype: a generic entity type that has a relationship with one or more subtype







# Rules for Supertype/Subtypes Relationships

- Total specialization: an entity instance of the supertype must be an instance of one of the subtypes
- Partial specialization: an entity instance of the supertype may or may not be an instance of one of the subtypes
- Disjoint: an entity instance of the supertype can be an instance of only one subtype
- Overlap: an entity instance of the supertype may be an instance of multiple subtypes





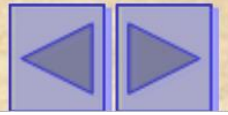
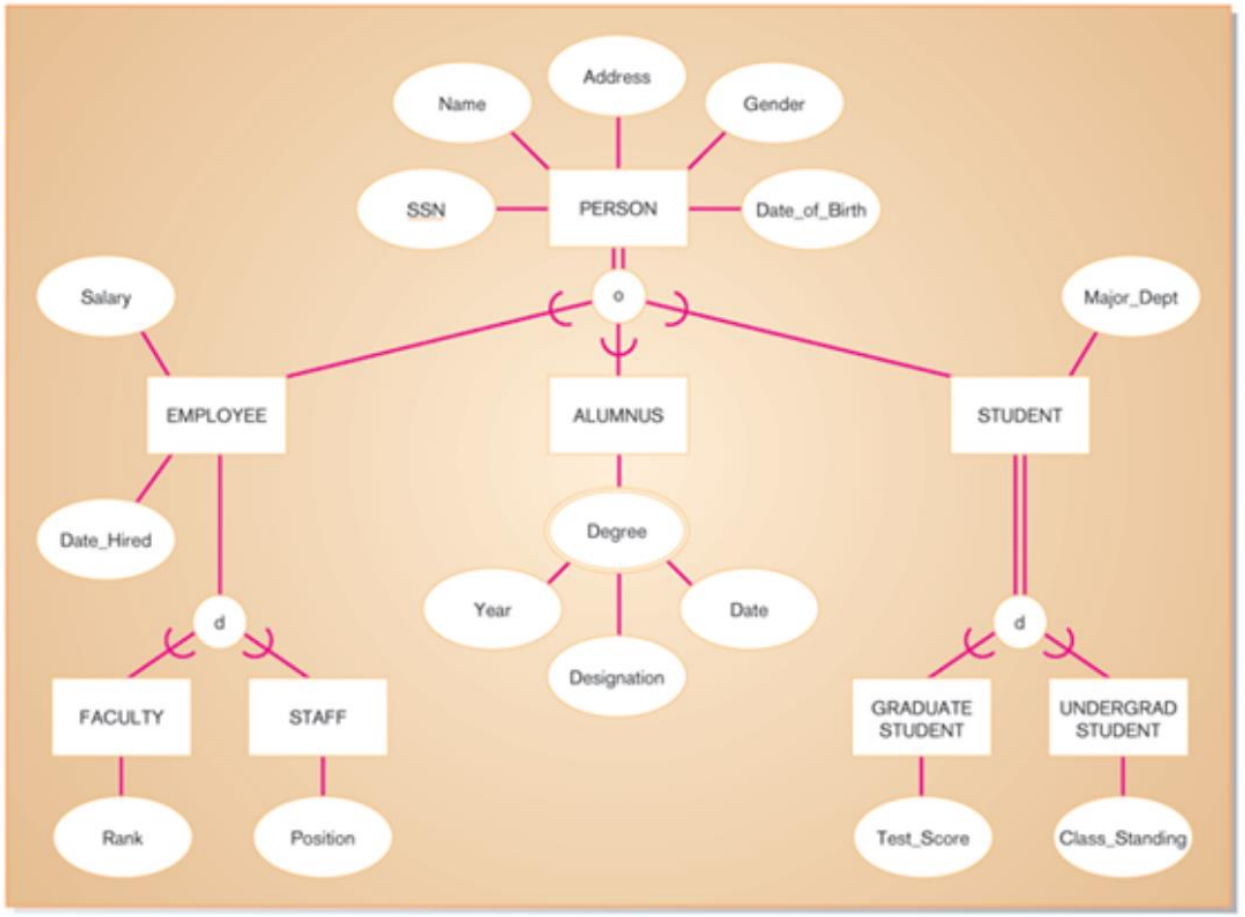


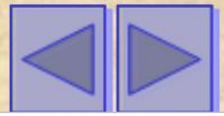
**Figure 9-12** Example of supertype/subtype heirarchy



# Business Rules

- Specifications that preserve the integrity of the logical data model
- Four types
  - Entity integrity: unique, non-null identifiers
  - Referential integrity constraints: rules governing relationships
  - Domains: valid values for attributes
  - Triggering operations: other business rules regarding attribute values



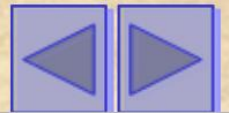



# Domains

- The set of all data types and ranges of values that an attribute can assume
- Several advantages
  1. Verify that the values for an attribute are valid
  2. Ensure that various data manipulation operations are logical
  3. Help conserve effort in describing attribute characteristics







# Triggering Operations

- An assertion or rule that governs the validity of data manipulation operations such as insert, update and delete
- Components:
  - User rule: statement of the business rule to be enforced by the trigger
  - Event: data manipulation operation that initiates the operation
  - Entity Name: name of entity being accessed or modified
  - Condition: condition that causes the operation to be triggered
  - Action: action taken when the operation is triggered





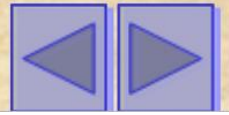


**Figure 9-13b** Examples of business rules - Typical domain definitions

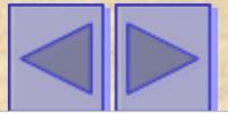
| | |
|---|---|
| Name: Account_Number | Name: Amount |
| Meaning: Customer account number in bank | Meaning: Dollar amount of transaction |
| Data type: Character | Data type: Numeric |
| Format: nnn-nnnn | Format: 2 decimal places |
| Uniqueness: Must be unique | Range: 0-10,000 |
| Null support: Non-null | Uniqueness: Nonunique |
| | Null support: Non-null |

**Figure 9-13c** Examples of business rules - Typical triggering operation

User rule: WITHDRAWAL Amount may not exceed ACCOUNT Balance
Event: Insert
Entity Name: WITHDRAWAL
Condition: WITHDRAWAL Amount > ACCOUNT Balance
Action: Reject the insert transaction







# Packaged Data Models

- Generic data models that can be applied and modified for an organization
- Two categories
  - Universal
  - Industry-specific
- Benefits
  - Reduced implementation time and cost
  - High-quality modeling





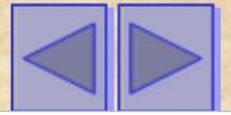
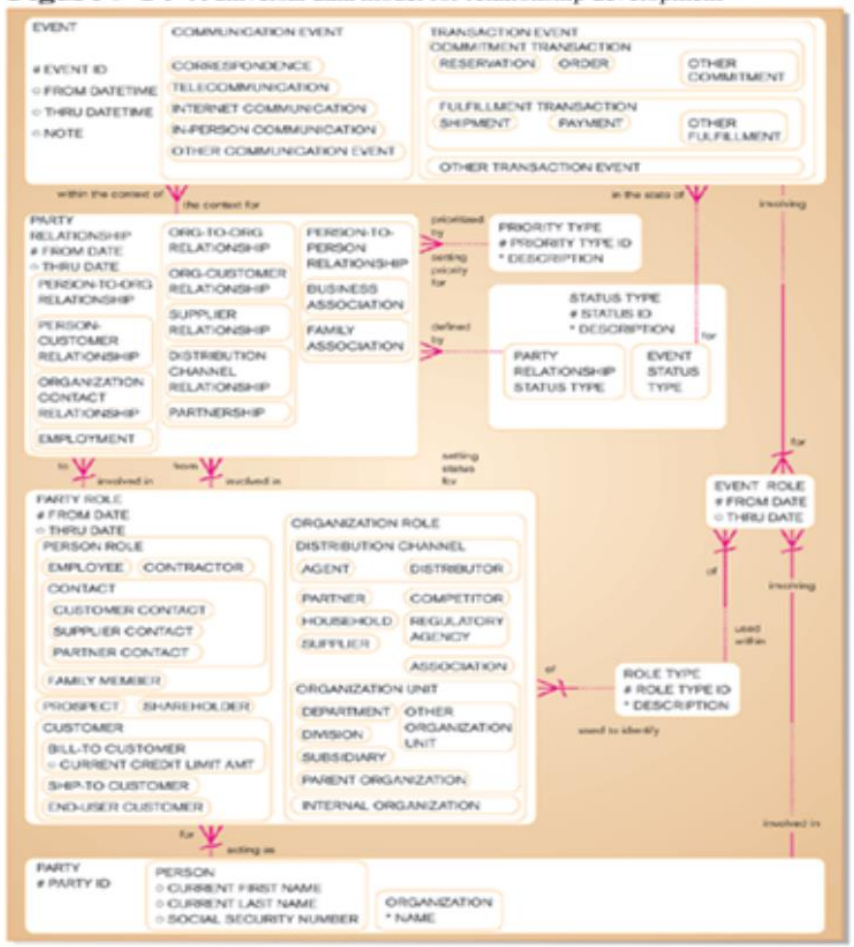


**Figure 9-14** A universal data model for relationship development




Source: From L. Silverston, "A Universal Data model for Relationship Development" *DM Review*, 2002, used with permission.)

Packaged data models provide generic models that can be customized for a particular organization's business rules