# Information Systems Planning and the Database Design Process

Luis Aguilar
University of California, Berkeley
School of Information

INFO 257: Database Management

# Announcements

- Questions? About Ch. 1 in Hoffer?
- Roll
  - Everyone here at least on the Waitlist?
- Assignment 1 posted
- Anybody not on Piazza?
- Class Today
  - Lecture
  - Docker Workshop (After Break)

# Lecture Outline

- Review
  - Database Terms
  - Database Types
- Information Systems Planning
- Information Systems Architecture
- Database Design

# Lecture Outline

- Review
  - Database Terms
  - Database Types
- Information Systems Planning
- Information Systems Architecture
- Database Design

# Terms and Concepts

- Database activities:
  - ***Create***
    - Add new data to the database
  - ***Read***
    - Read current data from the database
  - ***Update***
    - Update or modify current database data
  - ***Delete***
    - Remove current data from the database

# Terms and Concepts

- ***Enterprise***
  - Organization
- ***Entity***
  - Person, Place, Thing, Event, Concept...
- ***Attributes***
  - Data elements (facts) about some entity
  - Also sometimes called fields or items or domains
- ***Data values***
  - instances of a particular attribute for a particular entity

# Terms and Concepts

- ***Records***
  - The set of values for all attributes of a particular entity
  - AKA "tuples" or "rows" in relational DBMS
- ***File***
  - Collection of records
  - AKA "Relation" or "Table" in relational DBMS

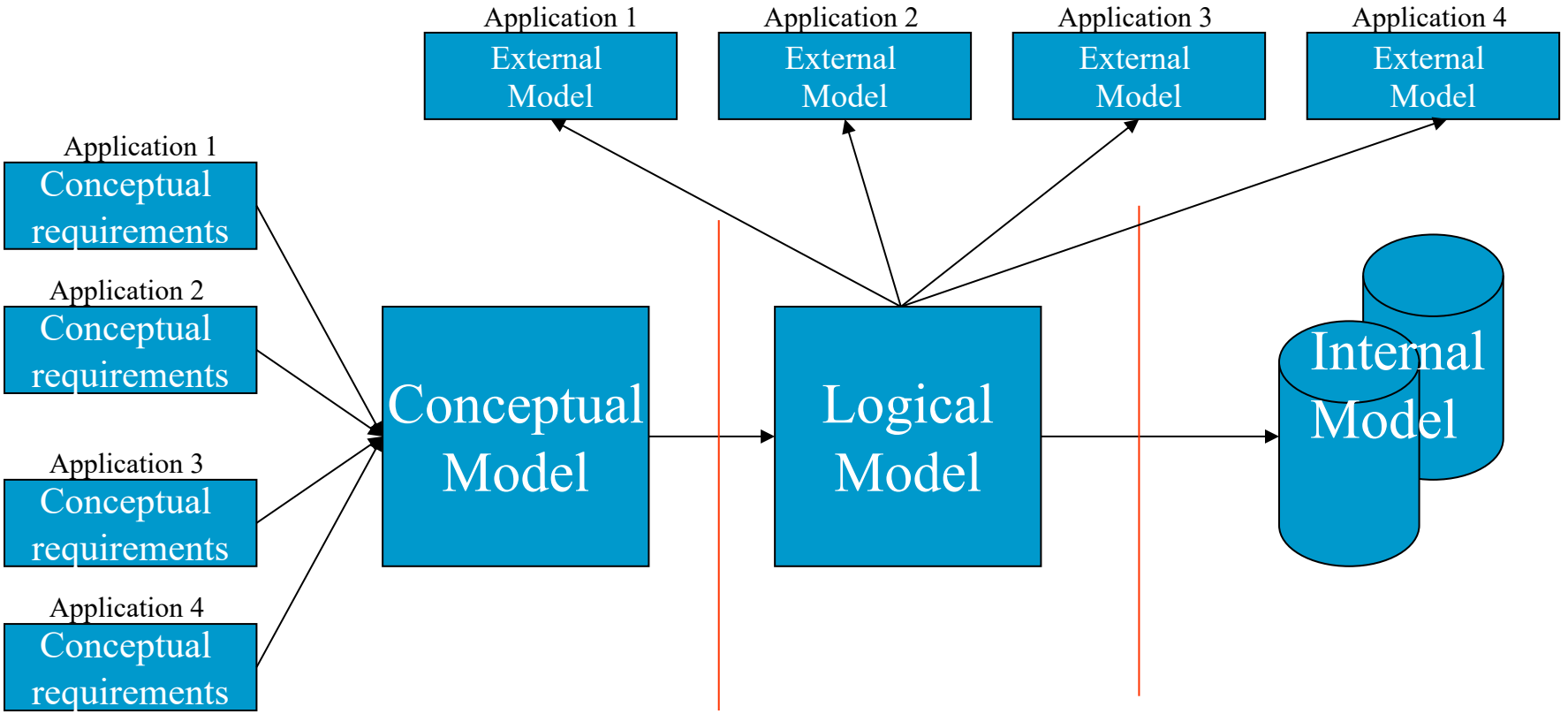
# Terms and Concepts

- ***Key***
  - an attribute or set of attributes used to identify or locate records in a file
- ***Primary Key***
  - an attribute or set of attributes that *uniquely* identifies each record in a file

# Terms and Concepts

- ***Models***
  - (1) Levels or views of the Database
    - Conceptual, logical, physical
  - (2) DBMS types
    - Relational, Hierarchic, Network, Object-Oriented, Object-Relational

# Models (1)


Application 1
External Model
Application 2
External Model
Application 3
External Model
Application 4
External Model
Application 1
Conceptual requirements
Application 2
Conceptual requirements
Application 3
Conceptual requirements
Application 4
Conceptual requirements
Conceptual Model
Logical Model
Internal Model

# Lecture Outline

- Review
  - Database Terms
  - Database Types
- **Information Systems Planning**
- Information Systems Architecture
- Database Design

# Information Systems Planning

- Scope of IS is now the entire organization
- Sometimes called “enterprise-wide” computing or “Information Architecture”
- Problem: isolated groups in an organization start their own databases and it becomes impossible to find out who has what information, where there are overlaps, and to assess the accuracy of the information

# Information Systems Planning

- To support enterprise-wide computing, there must be enterprise-wide information planning
- One framework for thinking about and planning for enterprise-wide computing is an *Information Systems Architecture* or ISA
- Most organizations do ***NOT*** have such an architecture

# Lecture Outline

- Review
  - Database Terms
  - Database Types
- Information Systems Planning
- Information Systems Architecture
- Database Design

# Information Systems Architecture

- An ISA is a "*conceptual blueprint or plan that expresses the desired future structure for information systems in an organization*"
- It provides a "*context within which managers throughout the organization can make consistent decisions concerning their information systems*"
  - Quotes from McFadden (Modern Database Management, 4th edition), Ch. 3

# Information Systems Architecture

- Benefits of ISA:
  - "Provides a basis for strategic planning of IS
  - Provides a basis for communicating with top management and a context for budget decisions concerning IS
  - Provides a unifying concept for the various stakeholders in information systems.
  - Communicates the overall direction for information technology and a context for decisions in this area
  - Helps achieve information integration when systems are distributed (increasing important in a global economy)
  - Provides a basis for evaluating technology options (for example, downsizing and distributed processing)"
    - Quotes from McFadden (Modern Database Management, 4th edition), Ch. 3

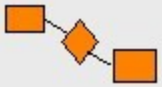
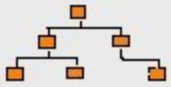
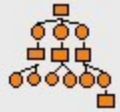
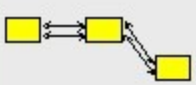
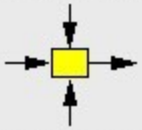
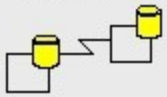
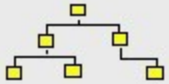
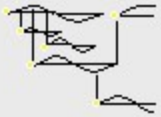
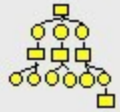
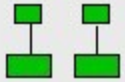
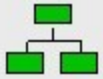
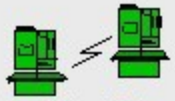
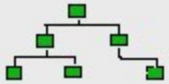
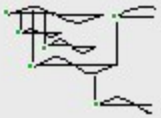
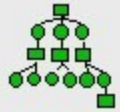
# Information Systems Architecture

- Zachman ISA Framework components
  - Data
    - The "**What**" of the information system
  - Process
    - The "**How**" of the information system
  - Network
    - The "**Where**" of the information system
  - People
    - **Who** performs processes and are the source and receiver of data and information.
  - Events and Points in time
    - **When** processes are performed
  - Reasons
    - **Why**: For events and rules that govern processing

# Information Systems Architecture

- Six roles or perspectives of the Data, Process and Network components
  - Business scope (Owner)
  - Business model (Architect)
  - Information systems model (Designer)
  - Technology model (Builder)
  - Technology definition (Contractor)
  - Information system (User)

# Zachman Framework

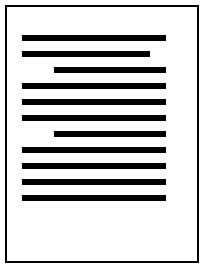
| abstractions / perspectives | DATA *What* | FUNCTION *How* | NETWORK *Where* | PEOPLE *Who* | TIME *When* | MOTIVATION *Why* |
|---|---|---|---|---|---|---|
| **SCOPE** *Planner* contextual | List of Things - *Important to the Business*  Entity = Class of Business Thing | List of Processes - *the Business Performs*  Function = Class of Business Process | List of Locations - *in which the Business Operates*  Node = Major Business Location | List of Organizations - *Important to the Business*  People = Class of People and Major Organizations | List of Events - *Significant to the Business*  Time = Major Business Event | List of Business Goals and Strategies  Ends/Means=Major Business Goal/Critical Success Factor |
| **ENTERPRISE MODEL** *Owner* conceptual | e.g., Semantic Model  Entity = Business Entity Rel. = Business Relationship | e.g., Business Process Model  Process = Business Process I/O = Business Resources | e.g., Logistics Network  Node = Business Location Link = Business Linkage | e.g., Work Flow Model  People = Organization Unit Work = Work Product | e.g., Master Schedule  Time = Business Event Cycle = Business Cycle | e.g., Business Plan  End = Business Objective Means = Business Strategy |
| **SYSTEM MODEL** *Designer* logical | e.g., Logical Data Model  Entity = Data Entity Rel. = Data Relationship | e.g., Application Architecture  Process.= Application Function I/O = User Views | e.g., Distributed System Architecture  Node = IS Function Link = Line Characteristics | e.g., Human Interface Architecture  People = Role Work = Deliverable | e.g., Processing Structure  Time = System Event Cycle = Processing Cycle | e.g., Business Rule Model  End = Structural Assertion Means = Action Assertion |
| **TECHNOLOGY CONSTRAINED MODEL** *Builder* physical | e.g., Physical Data Model  Entity = Tables/Segments/etc. Rel. = Key/Pointer/etc. | e.g., System Design  Process= Computer Function I/O =Data Elements/Sets | e.g., Technical Architecture  Node = Hardware/System Software Link = Line Specifications | e.g., Presentation Architecture  People = User Work = Screen/Device Format | e.g., Control Structure  Time = Execute Cycle = Component Cycle | e.g., Rule Design  End = Condition Means = Action |
| **DETAILED REPRESEN-TATIONS** *Subcontractor* out-of-context | e.g. Data Definition  Entity = Field Rel. = Address | e.g. Program  Process= Language Statement I/O = Control Block | e.g. Network Architecture  Node = Addresses Link = Protocols | e.g. Security Architecture  People = Identity Work = Job | e.g. Timing Definition  Time = Interrupt Cycle = Machine Cycle | e.g. Rule Specification  End = Sub-condition Means = Step |
| FUNCTIONING ENTERPRISE | DATA *Implementation* | FUNCTION *Implementation* | NETWORK *Implementation* | ORGANIZATION *Implementation* | SCHEDULE *Implementation* | STRATEGY *Implementation* |

John A. Zachman, Zachman International

# Information Systems Architecture

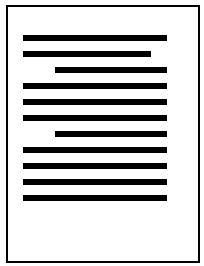
## Data

**List of entities important to the business**



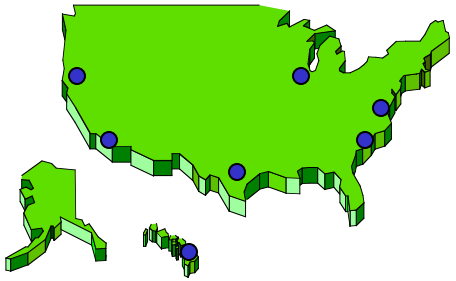
## Process

**List of processes or functions that the business performs**



## Network

**List of locations in which the business operates**



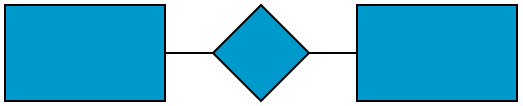
**1. Enterprise Scope**

**(Owner)**

# Information Systems Architecture

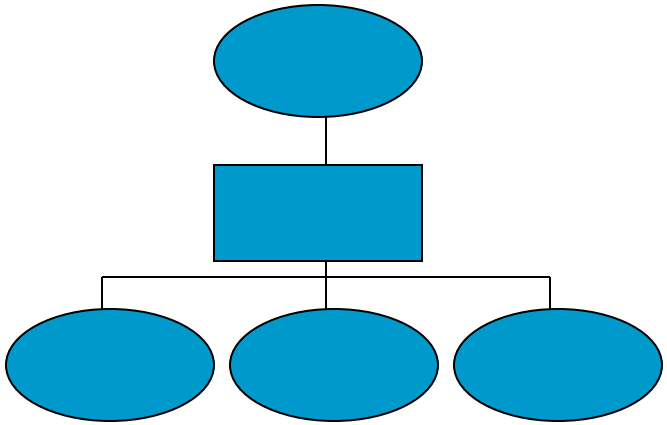
## Data

**Business entities and their relationships**



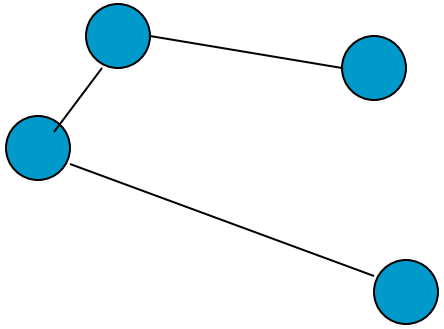
## Process

**Function and process decomposition**



## Network

**Communications links between business locations**



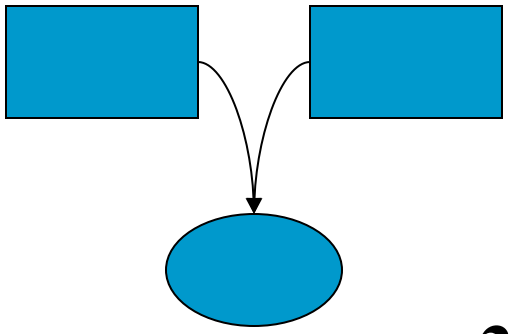
**2. Enterprise Model**

**(Architect)**

# Information Systems Architecture

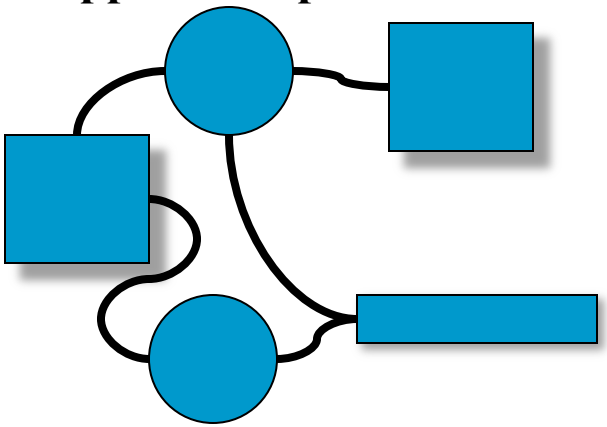
## Data

**Model of the business data and their relationships (ERD in Database design)**



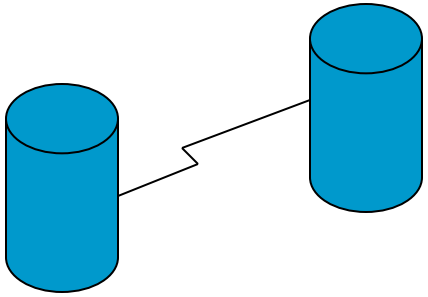
## Process

**Flows between application processes**



## Network

**Distribution Network**

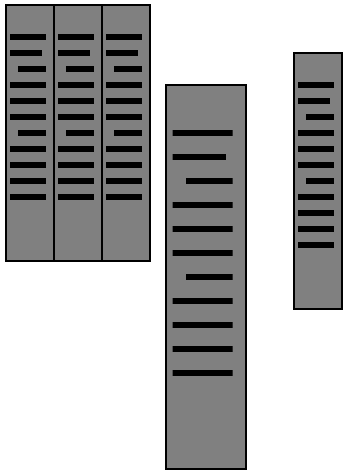


## 3. Information System Model (Designer)

# Information Systems Architecture

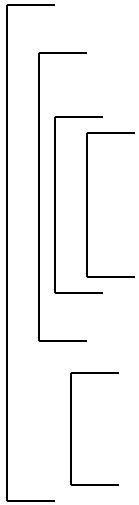
**Data**

**Database Design (logical)**



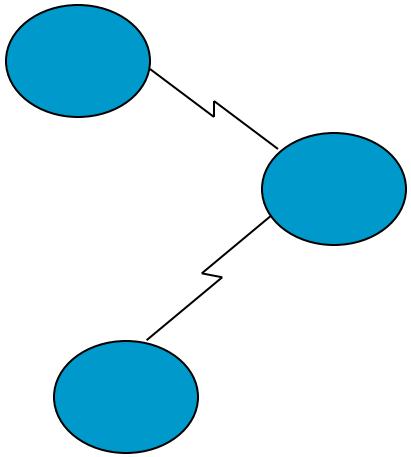
**Process**

**Process specifications**



**Network**

**Database Design**



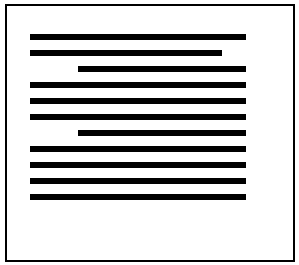
## 4. Technology Constrained Model

## (Builder)

# Information Systems Architecture

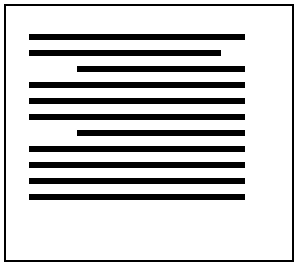
## Data

**Database Schema and subschema definition**



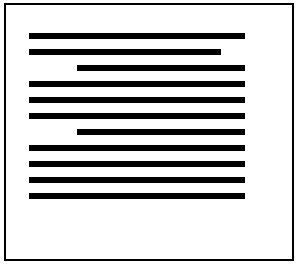
## Process

**Program Code and control blocks**



## Network

**Configuration definition/ Network Architecture**



**5. Technology Definition/ Detailed Representations (Contractor)**

# Information Systems Architecture

| Data | Process | Network |
| --- | --- | --- |
| Implemented Database and information | Implemented Application Programs | Current System Configuration |

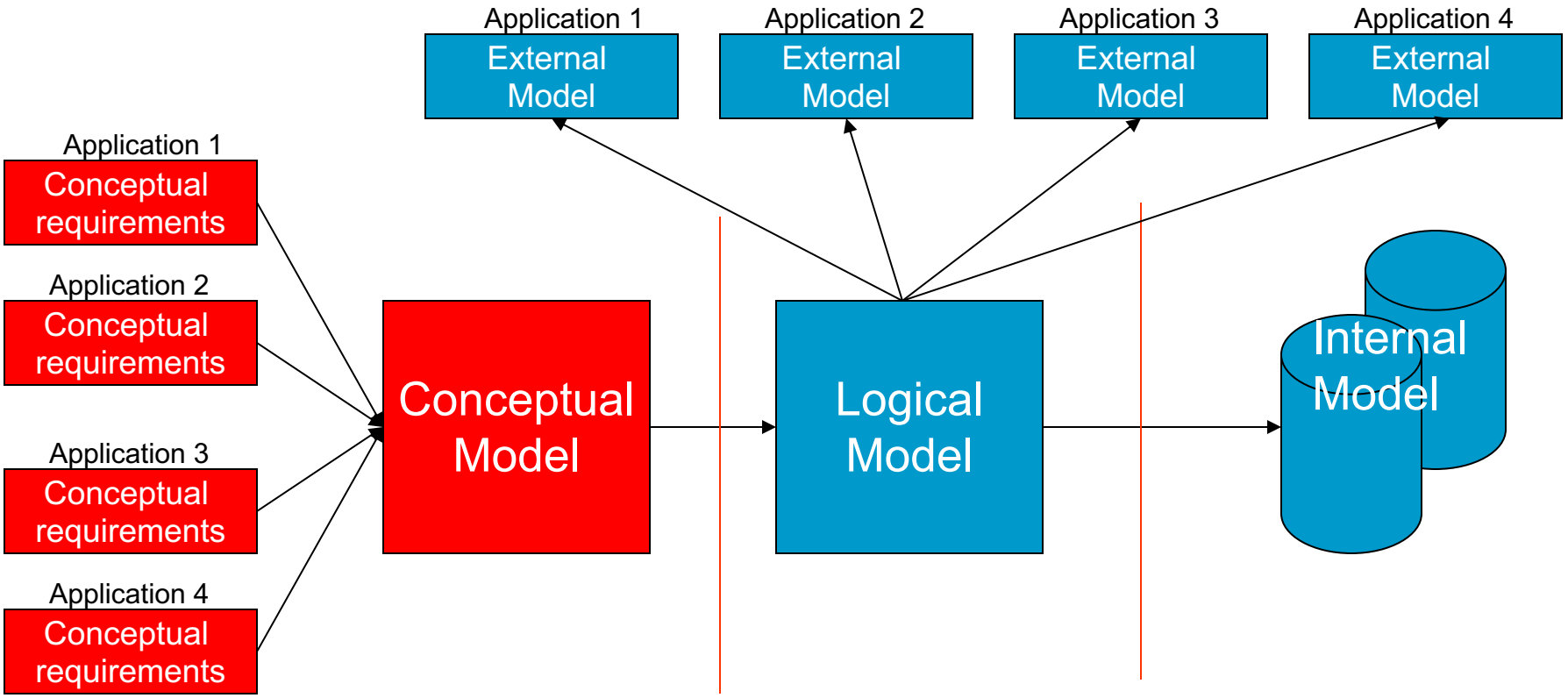
**6. Functioning Enterprise**

**(User)**

# Lecture Outline

- Review
  - Database Terms
  - Database Types
- Information Systems Planning
- Information Systems Architecture
- Database Design

# Database Design Process



# Stages in Database Design

1. Requirements formulation and analysis
2. Conceptual Design -- Conceptual Model
3. Implementation Design -- Logical Model
4. Physical Design -- Physical Model

# Database Design Process

- Requirements formulation and analysis
  - Purpose: Identify and describe the data that are used by the organization
  - Results: Metadata identified, Data Dictionary, Conceptual Model-- ER diagram

# Database Design Process

- Requirements Formulation and analysis
  - Systems Analysis Process
    - Examine all of the information sources used in existing applications
    - Identify the characteristics of each data element
      - numeric
      - text
      - date/time
      - etc.
    - Examine the tasks carried out using the information
    - Examine results or reports created using the information

# Database Design Process

- Conceptual Model
  - Merge the collective needs of all applications
  - Determine what ***Entities*** are being used
    - Some object about which information is to maintained
  - What are the ***Attributes*** of those entities?
    - Properties or characteristics of the entity
    - What attributes uniquely identify the entity
  - What are the ***Relationships*** between entities
    - How the entities interact with each other?

# Database Design Process

- Logical Model
  - How is each entity and relationship represented in the Data Model of the DBMS
    - Hierarchic?
    - Network?
    - Relational?
    - Object-Oriented?

# Database Design Process

- Physical (AKA Internal) Model
  - Choices of index file structure
  - Choices of data storage formats
  - Choices of disk layout

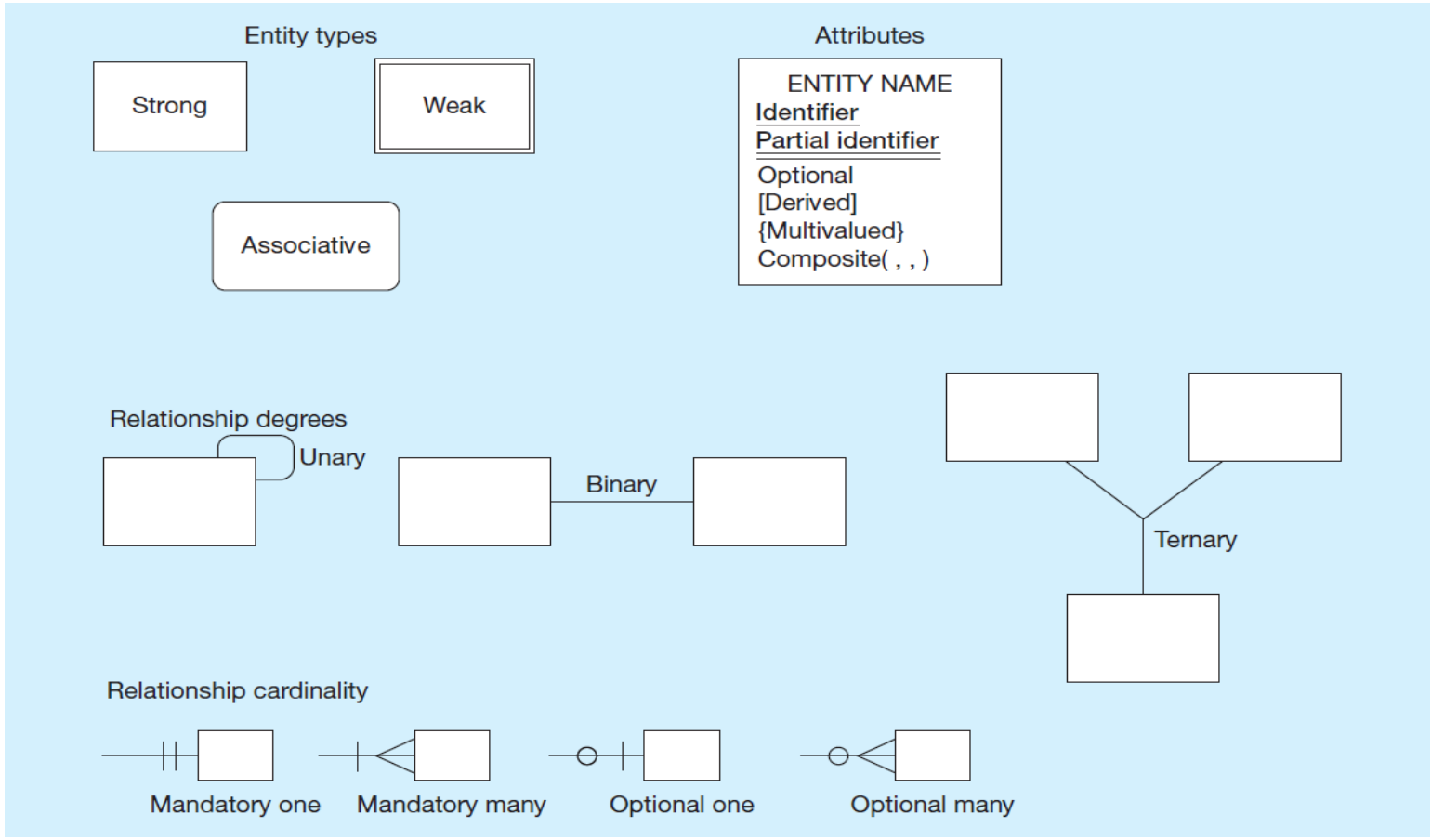
# Database Design Process

- External Model
  - User views of the integrated database
  - Making the old (or updated) applications work with the new database design

# Developing a Conceptual Model

- Overall view of the database that integrates all the needed information discovered during the requirements analysis.
- Elements of the Conceptual Model are represented by diagrams, *Entity-Relationship or ER Diagrams*, that show the meanings and relationships of those elements independent of any particular database systems or implementation details.
- We will be looking at Chen and Crow's Foot notation

# Basic E-R Notation



# Entity

- An Entity is an object in the real world (or even imaginary worlds) about which we want or need to maintain information
  - Persons (e.g.: customers in a business, employees, authors)
  - Things (e.g.: purchase orders, meetings, parts, companies)

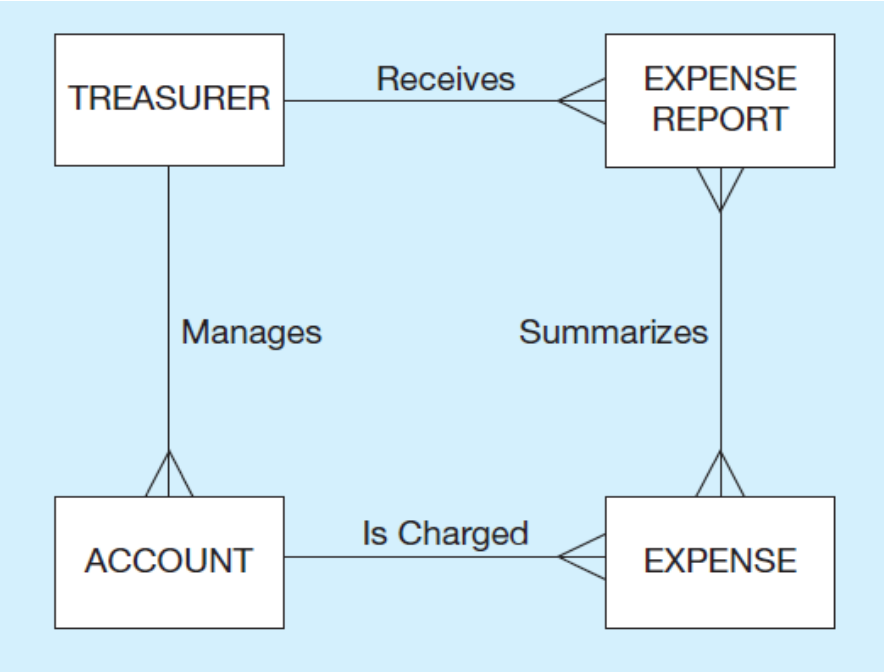
Employee

# An Entity…

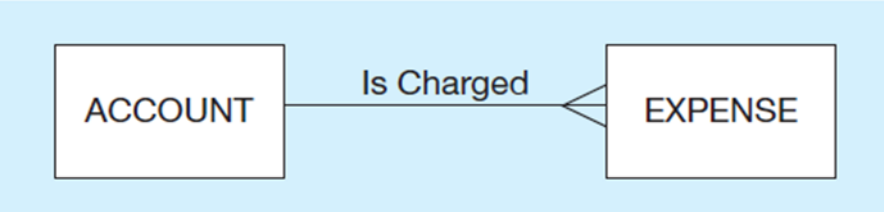
- **Should Be**:
  - An object that will have many instances in the database
  - An object that will be composed of multiple attributes
  - An object that we are trying to model
- **Should Not Be**:
  - A user of the database system
  - An output of the database system (e.g., a report)

# Example of Inappropriate Entities

(a) System user (Treasurer) and output (Expense Report) shown as entities




b) E-R diagram with only the necessary entities



# Guidelines for Naming Entities

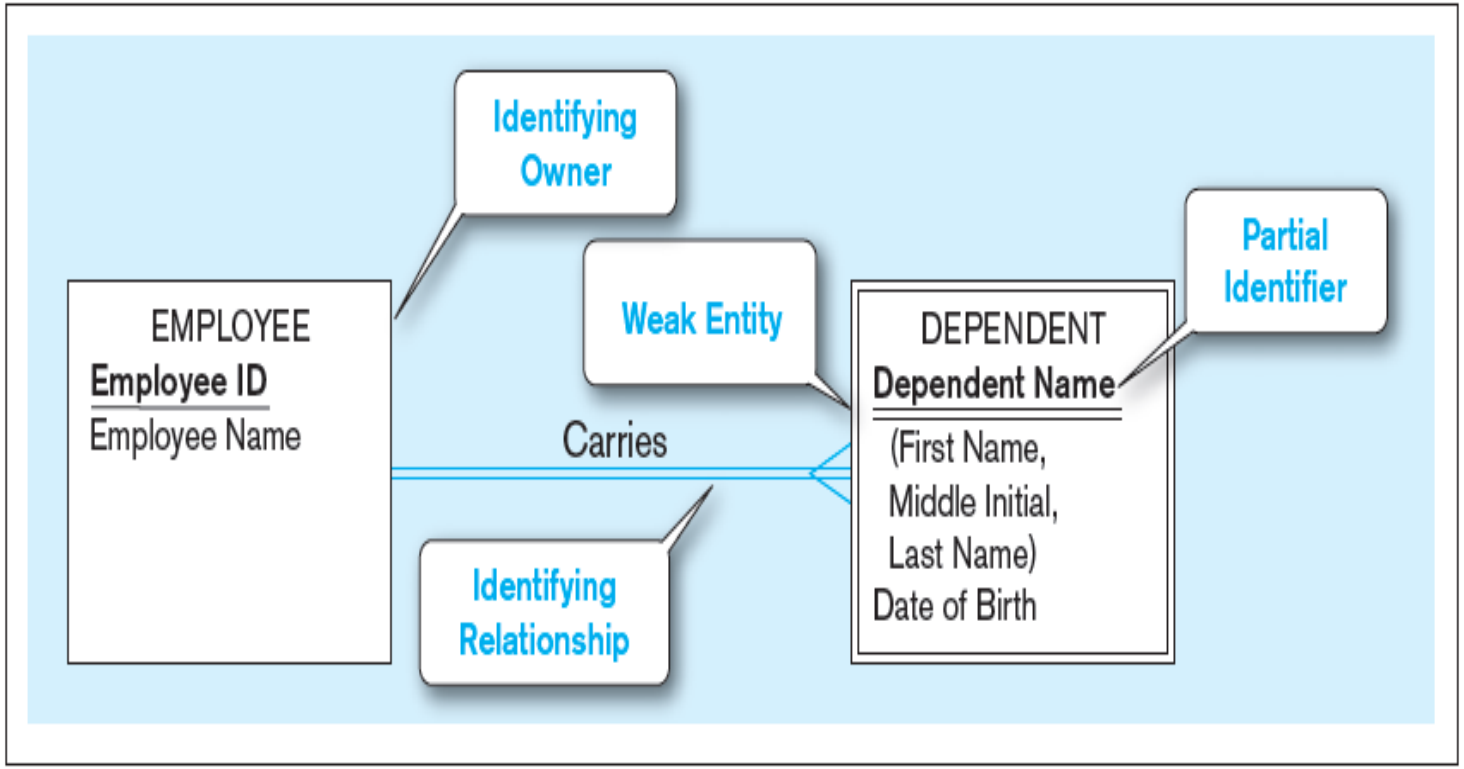
- Singular noun
- Specific to organization
- Concise, or abbreviation
- For event entities, the result not the process
- Name consistent for all diagrams

# Strong vs. Weak Entities, & Identifying Relationships

- Strong entity
  - exists independently of other types of entities
  - has its own unique identifier
    - identifier underlined with single line
- Weak entity
  - dependent on a strong entity (identifying owner); cannot exist on its own
  - does not have a unique identifier (only a partial identifier)
  - entity box and partial identifier have double lines
- Identifying relationship
  - links strong entities to weak entities

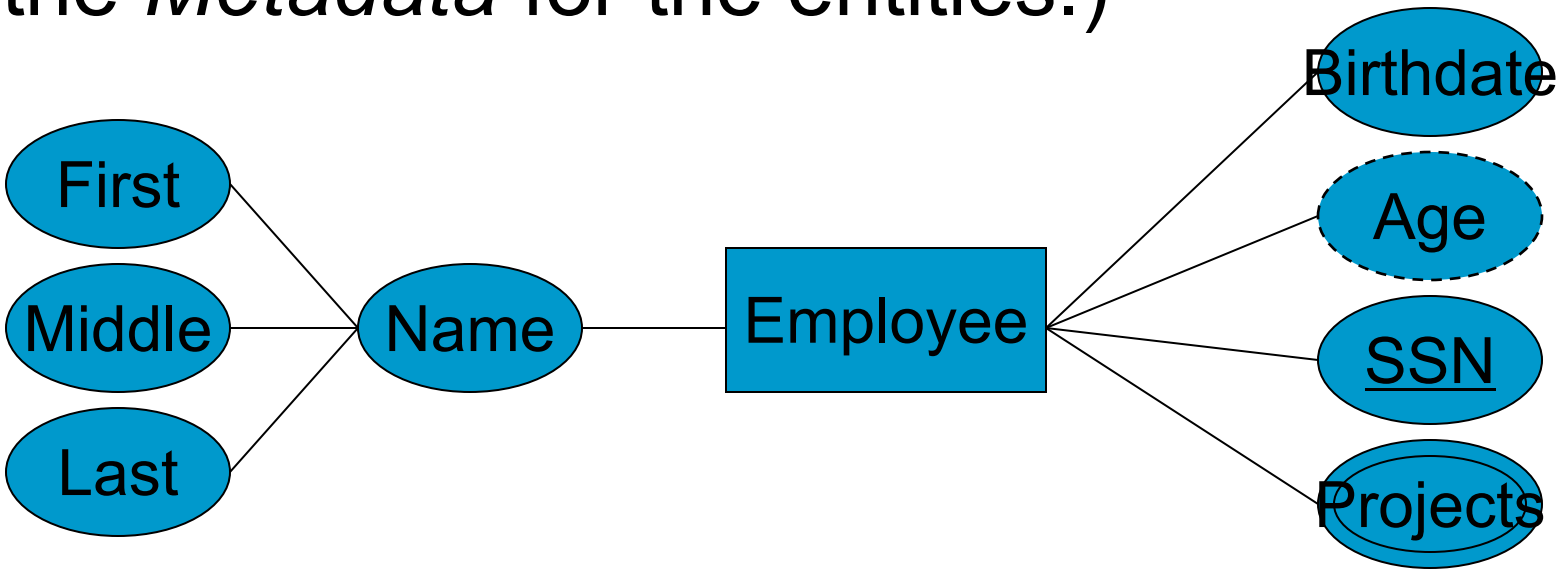
# Example of a Weak Identity and Its Identifying Relationship

Weak entity with identifying relationship.



# Attributes

- Attributes are the significant properties or characteristics of an entity that help identify it and provide the information needed to interact with it or use it. (This is the *Metadata* for the entities.)



# Required versus. Optional Attributes

Entity type: STUDENT

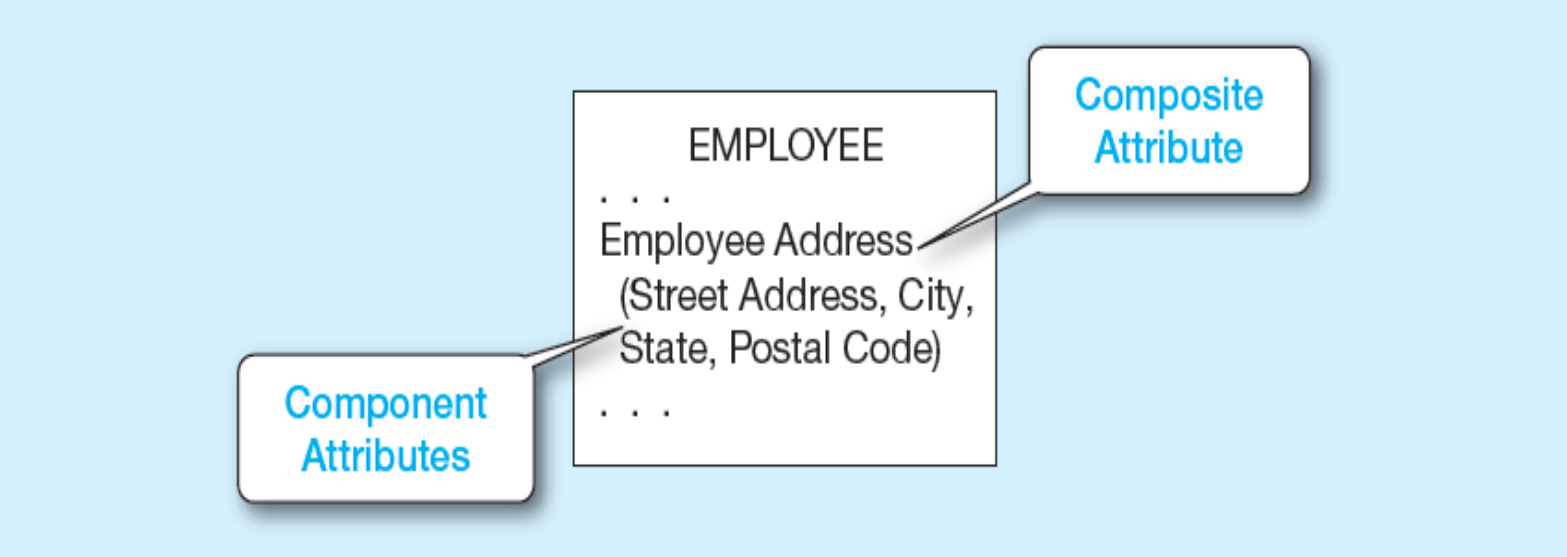
| Attributes | Attribute Data Type | Required or Optional | Example Instance | Example Instance |
|---|---|---|---|---|
| Student ID | CHAR (10) | Required | 28-618411 | 26-844576 |
| Student Name | CHAR (40) | Required | Michael Grant | Melissa Kraft |
| Home Address | CHAR (30) | Required | 314 Baker St. | 1422 Heft Ave |
| Home City | CHAR (20) | Required | Centerville | Miami |
| Home State | CHAR (2) | Required | OH | FL |
| Home Zip Code | CHAR (9) | Required | 45459 | 33321 |
| Major | CHAR (3) | Optional | MIS | |

**Required** – must have a value for every entity (or relationship) instance with which it is associated

**Optional** – may not have a value for every entity (or relationship) instance with which it is associated

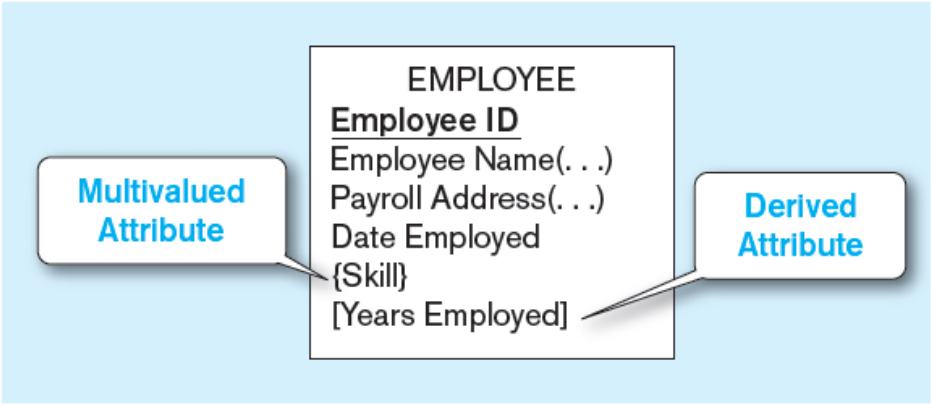
# Composite Attribute

- **Composite attribute** – An attribute that has meaningful component parts (sub-attributes)



# Multivalued and Derived Attributes

- **Multivalued**
  - May take on more than one value for a given entity (or relationship) instance
  - An employee can have more than one skill
- **Derived**
  - Values can be calculated from related attribute values (not physically stored in the database)
  - Years employed calculated from date employed and current date



# Identifiers (Keys)

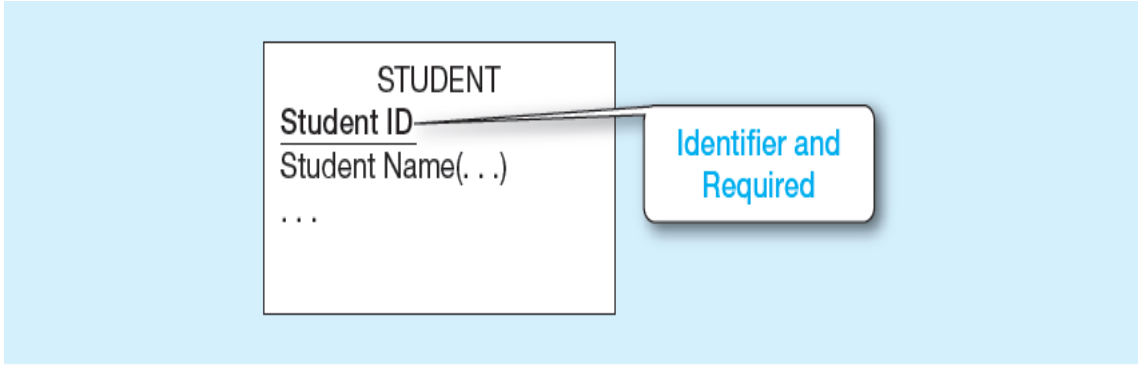
- Identifier (Key) – an attribute (or combination of attributes) that uniquely identifies individual instances of an entity type
- Simple versus Composite Identifier
- Candidate Identifier – an attribute that could be an identifier; it satisfies the requirements for being an identifier

# Criteria for Identifiers

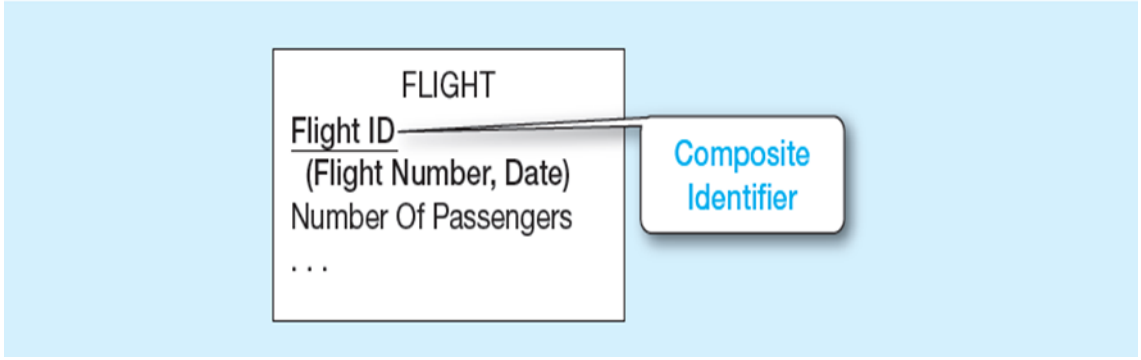
- Choose Identifiers that
  - Will not change in value
  - Will not be null
- Avoid intelligent identifiers (e.g., containing locations or people that might change)
- Substitute new, simple keys for long, composite keys

# Simple and Composite Identifier Attributes

a) Simple identifier attribute




b) Composite identifier attribute



# Naming Attributes

- Name should be a singular noun or noun phrase
- Name should be unique
- Name should follow a standard format
- Similar attributes of different entity types should use the same qualifiers and classes

# Defining Attributes (1 of 2)

- State what the attribute is and possibly why it is important
- Make it clear what is and is not included in the attribute's value
- Include aliases in documentation

# Defining Attributes (2 of 2)

- State source of values
- State whether attribute value can change once set
- Specify whether required or optional
- State min and max number of occurrences allowed
- Indicate relationships with other attributes

# Relationships

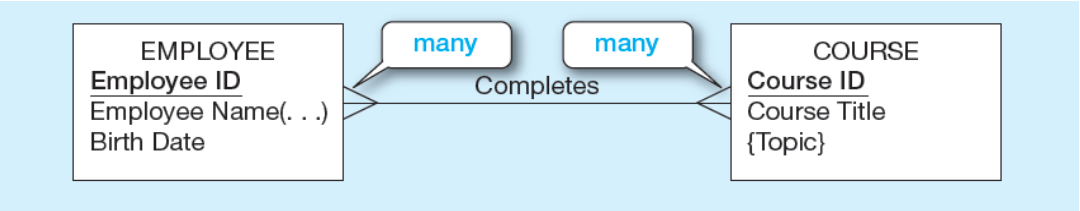
- Relationships are the associations between entities. They can involve one or more entities and belong to particular relationship types

# Modeling Relationships

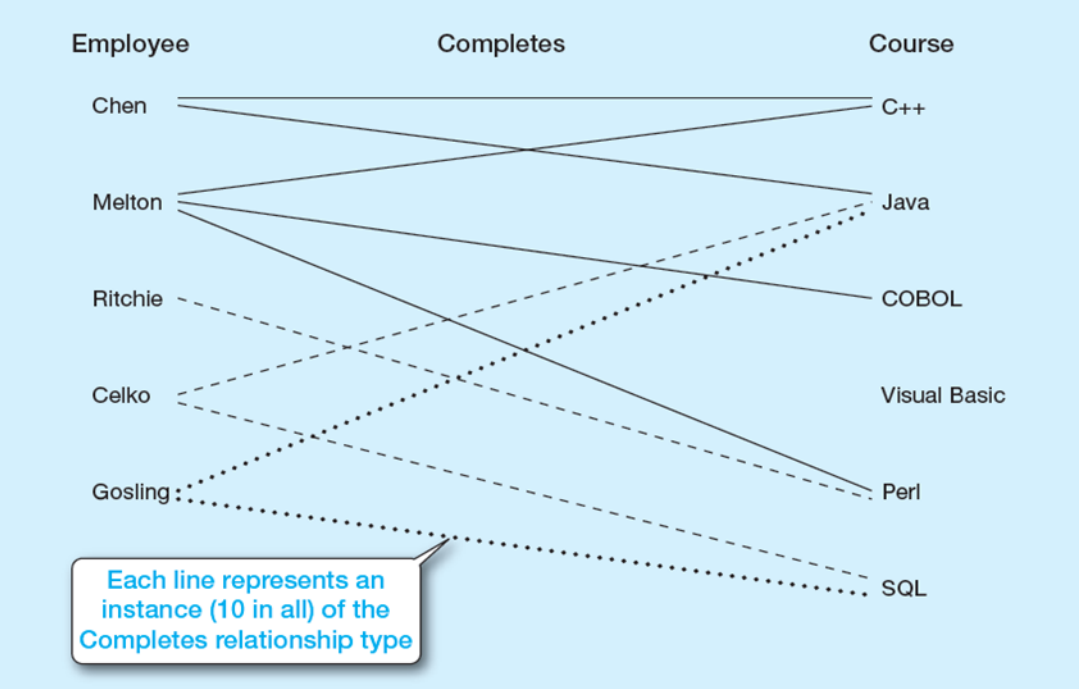
- Relationship Types vs. Relationship Instances
  - The relationship type is modeled as lines between entity types. The relationship instance is between specific entity instances
- Relationships can have attributes
  - These describe features pertaining to the association between the entities in the relationship
- Two entities can have more than one type of relationship between them (multiple relationships)
- Associative Entity – combination of relationship and entity

# Relationship Type and Instances

a) Relational type (Completes)




b) Relationship instances

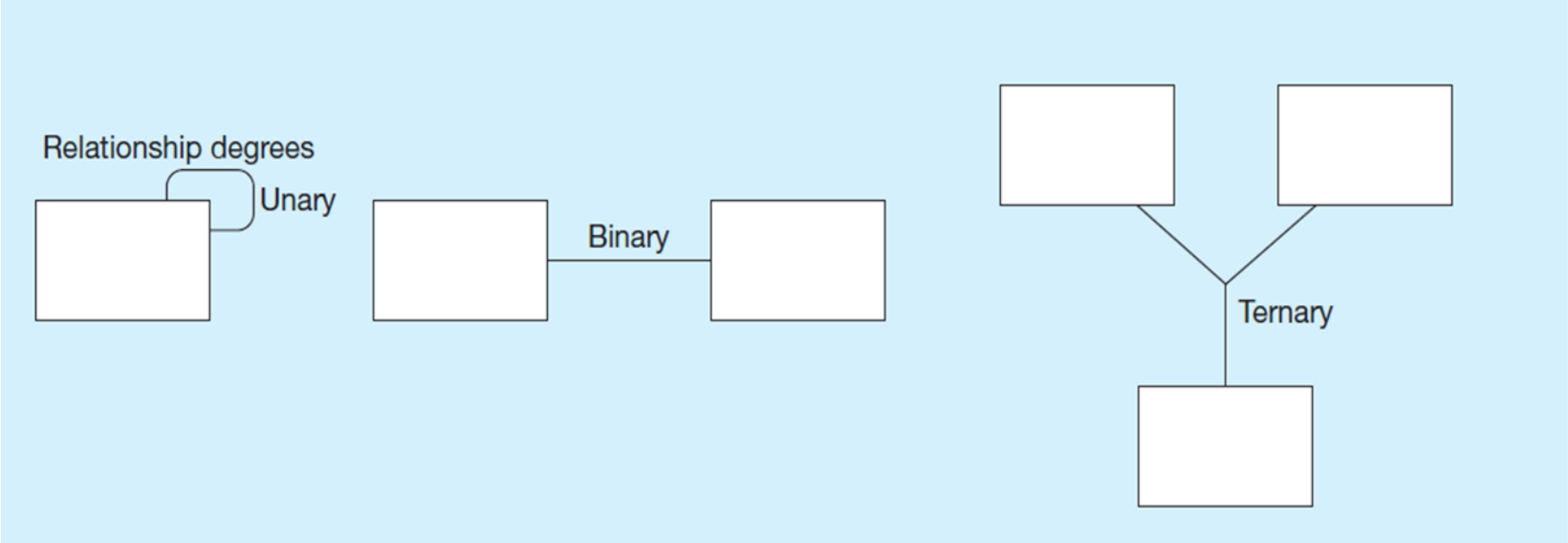


# Degrees of Relationships

Unary – entities of the same entity type related to each other

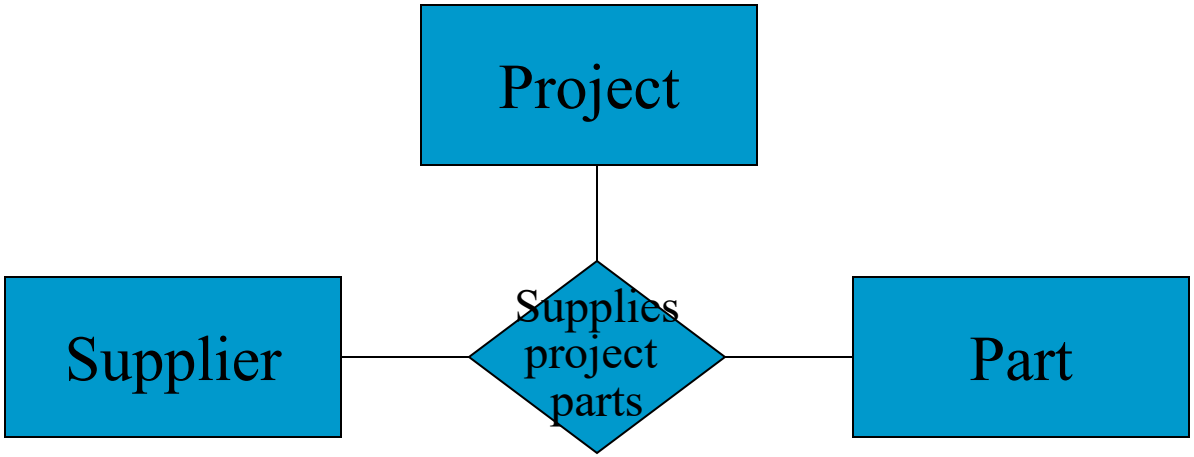
Binary – entities of one type related to entities of another

Ternary – entities of three different types involved in the same relationship



# Relationships

Binary

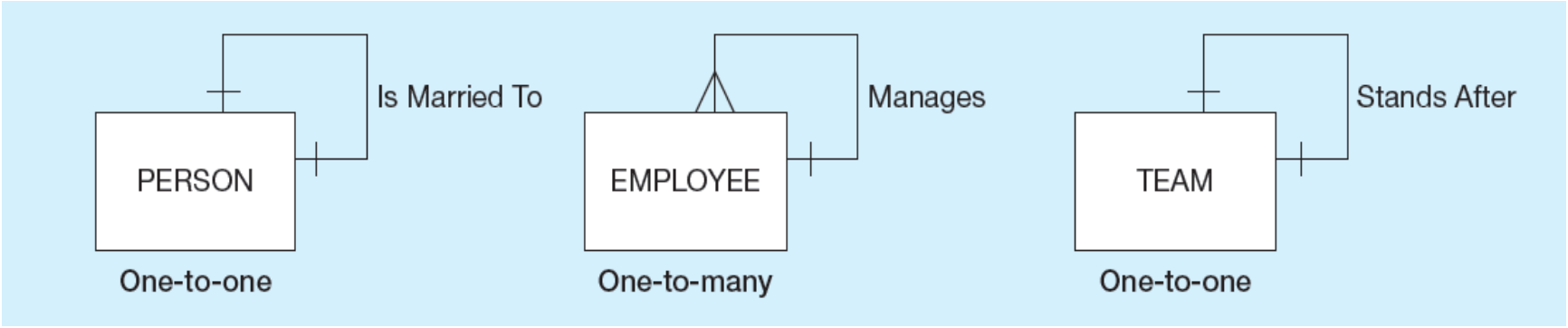

Ternary

# Cardinality of Relationships

- One-to-One
  - Each entity in the relationship will have exactly one related entity
- One-to-Many
  - An entity on one side of the relationship can have many related entities, but an entity on the other side will have a maximum of one related entity
- Many-to-Many
  - Entities on both sides of the relationship can have many related entities on the other side

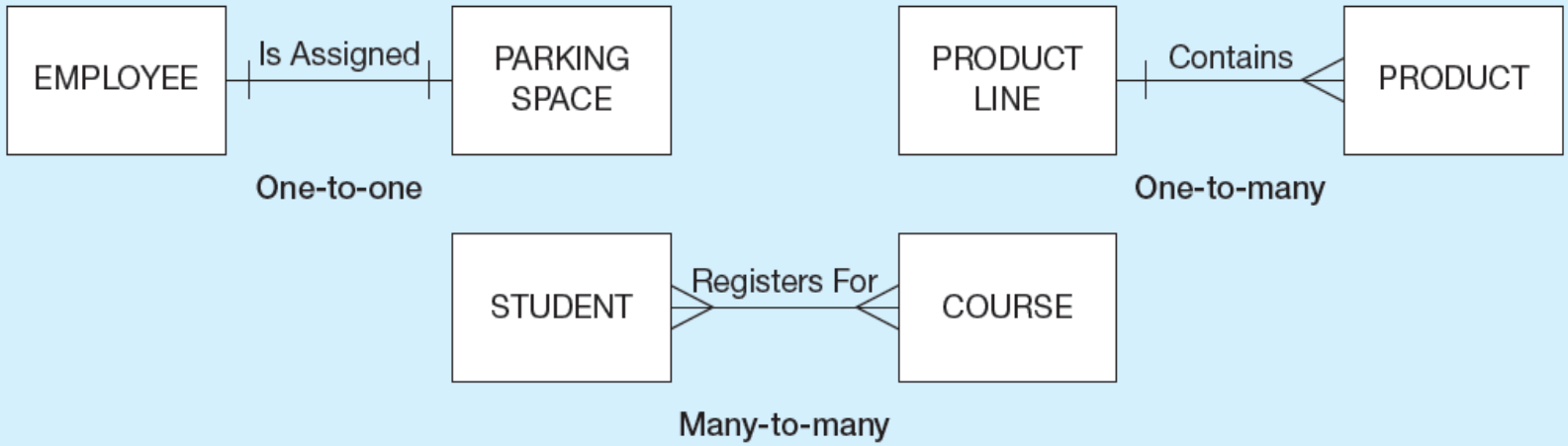
# Examples of Relationships of Different Degrees

a) Unary relationships



# Examples of Relationships of Different Degrees

b) Binary relationships

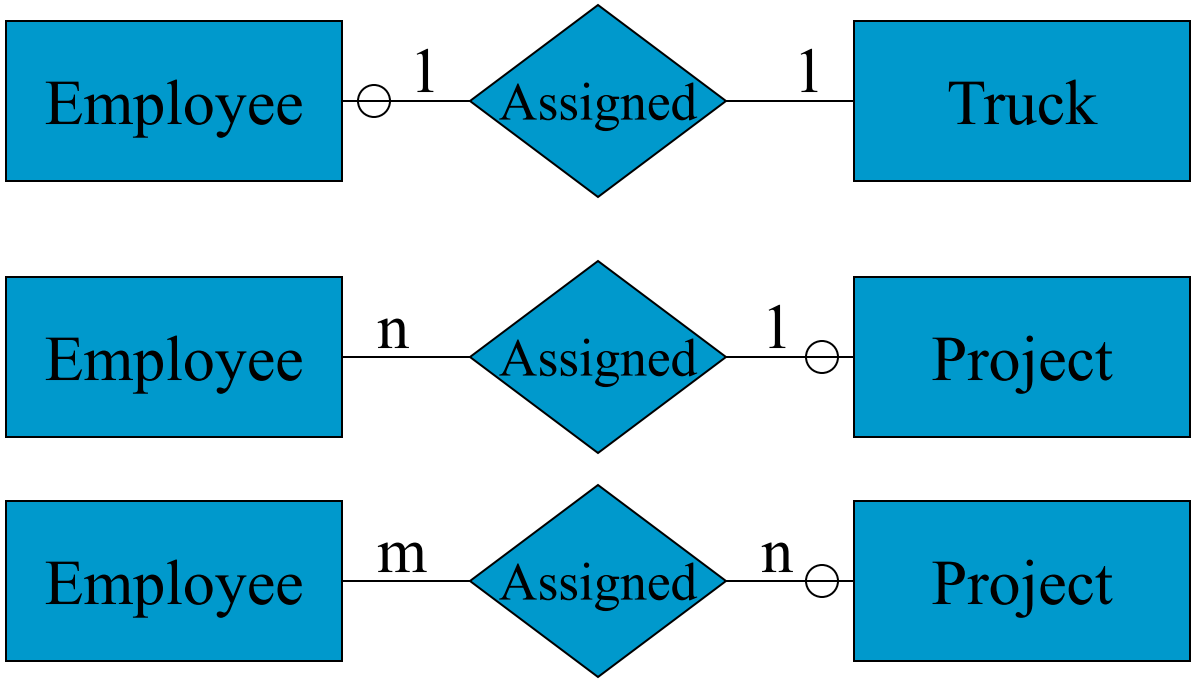


# Cardinality Constraints

- Cardinality Constraints — the number of instances of one entity that can or must be associated with each instance of another entity
- Minimum Cardinality
  - If zero, then optional
  - If one or more, then mandatory
- Maximum Cardinality
  - The maximum number

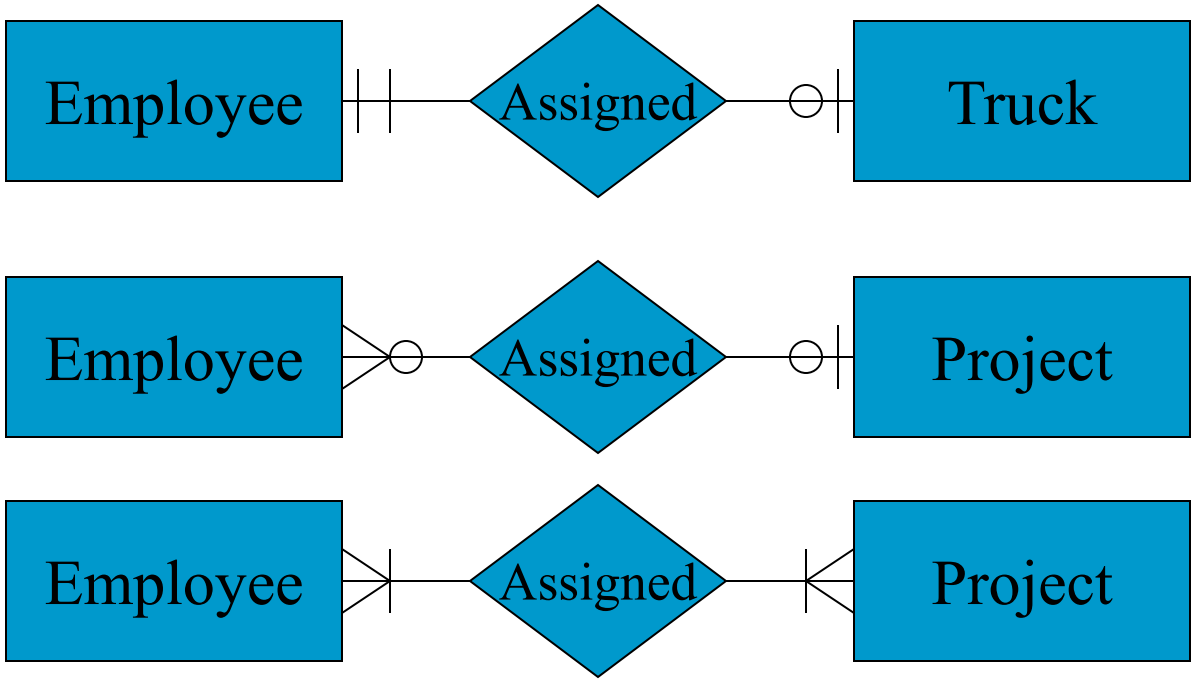
# Types of Relationships

- Concerned only with *cardinality* of relationship




Chen ER notation

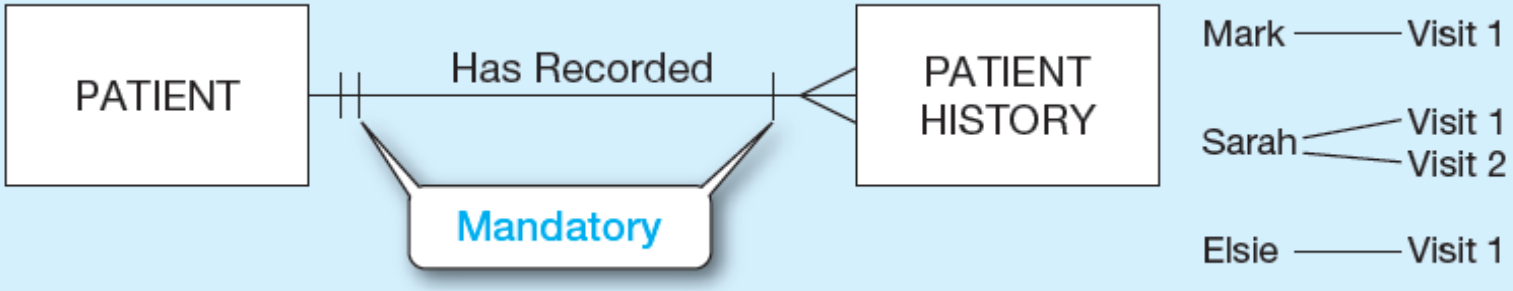
# Other Notations




"Crow's Foot"

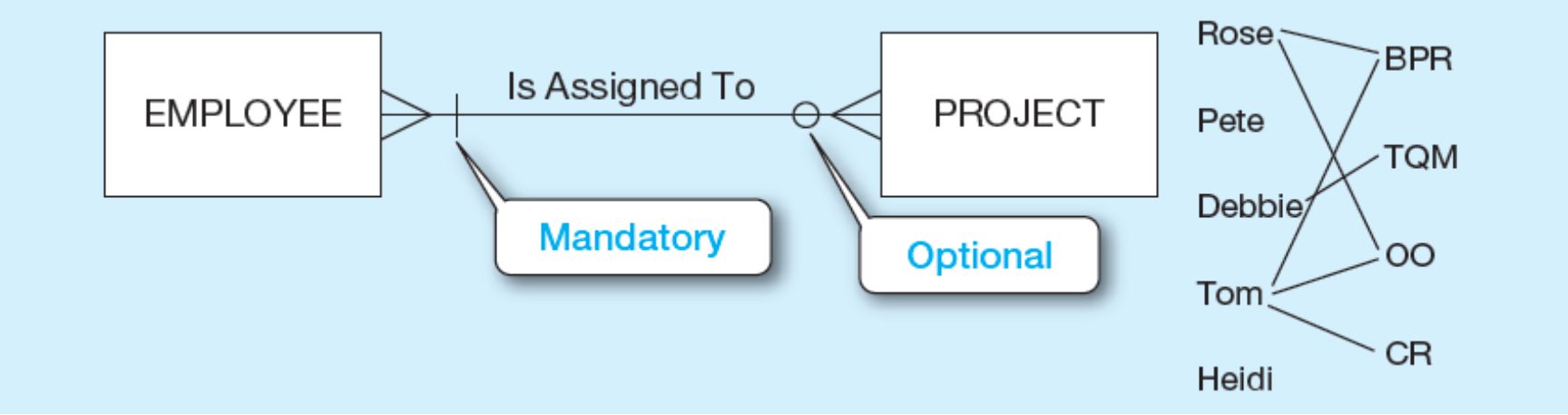
# Examples of Cardinality Constraints (1 of 3)

a) Mandatory cardinalities



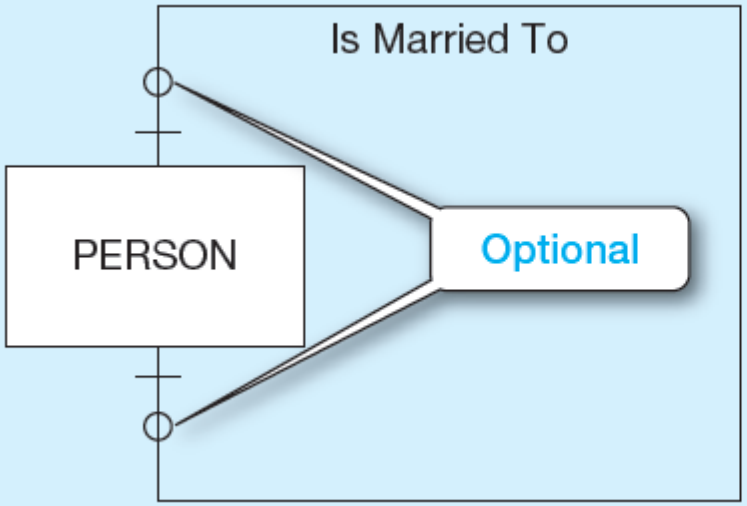
# Examples of Cardinality Constraints
(2 of 3)

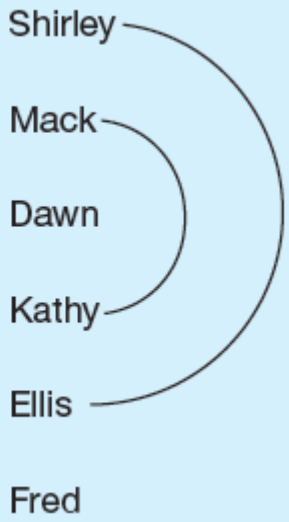
b) One mandatory, one optional cardinality



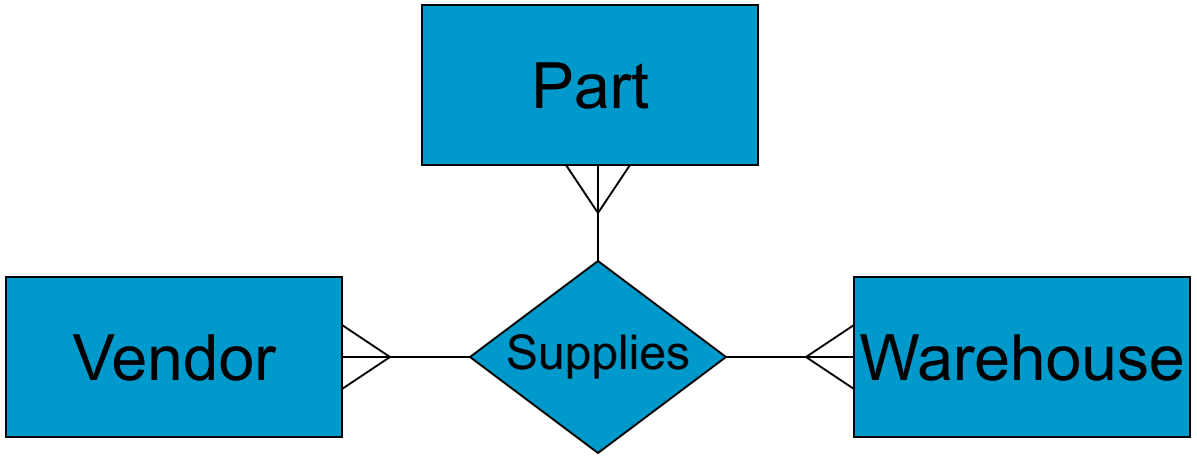
# Examples of Cardinality Constraints
(3 of 3)

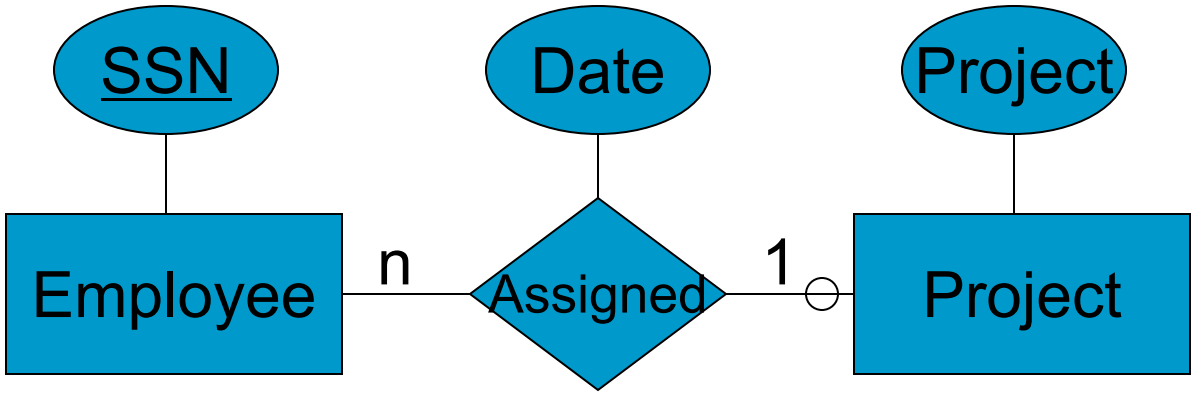
c) Optional cardinalities

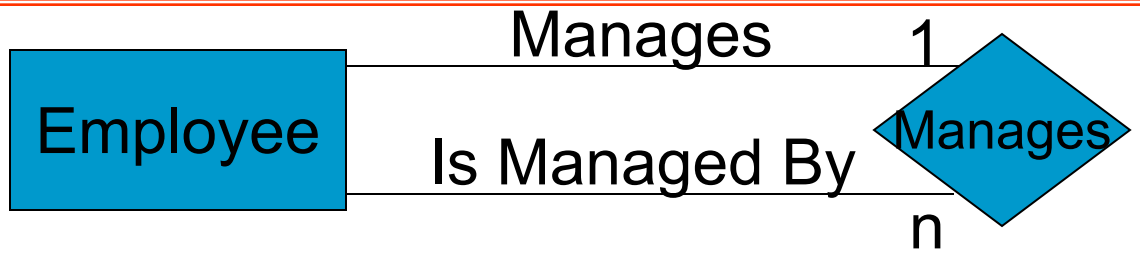


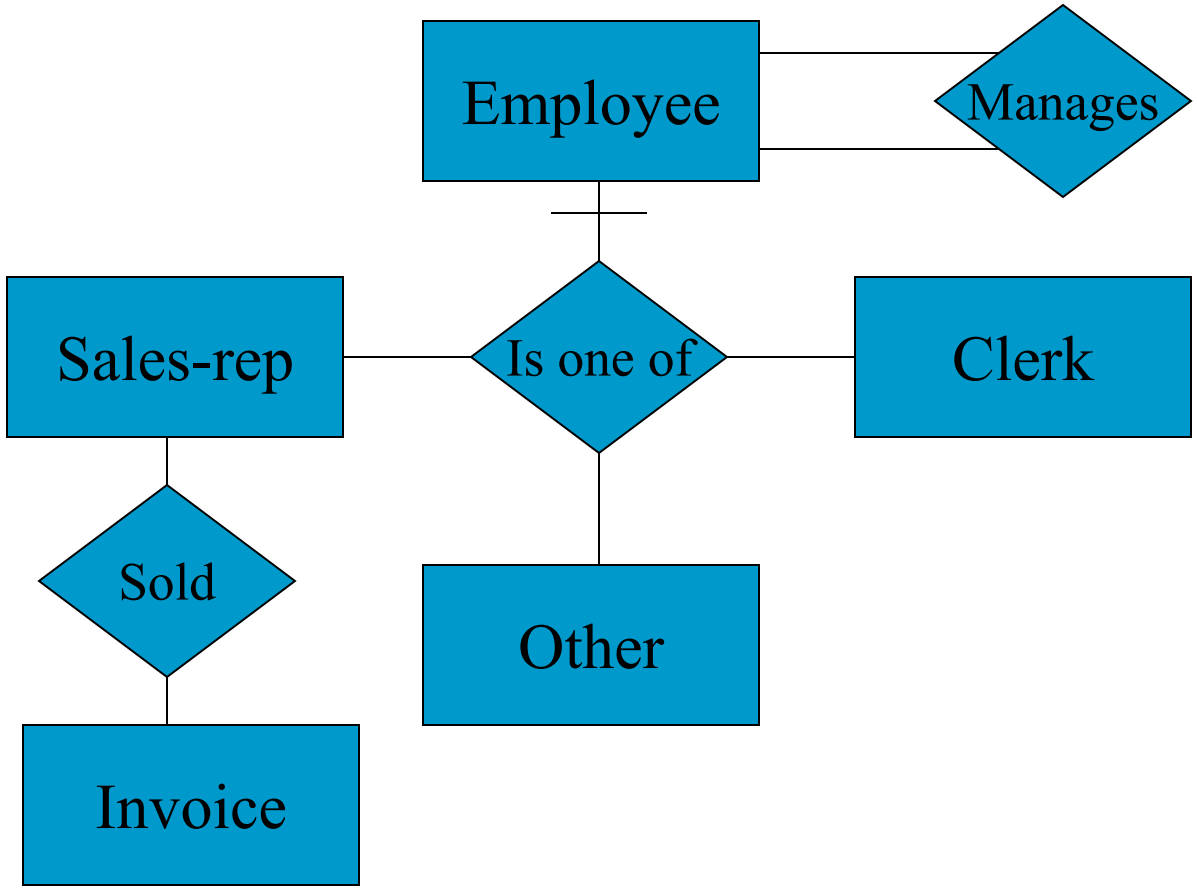
# More Complex Relationships



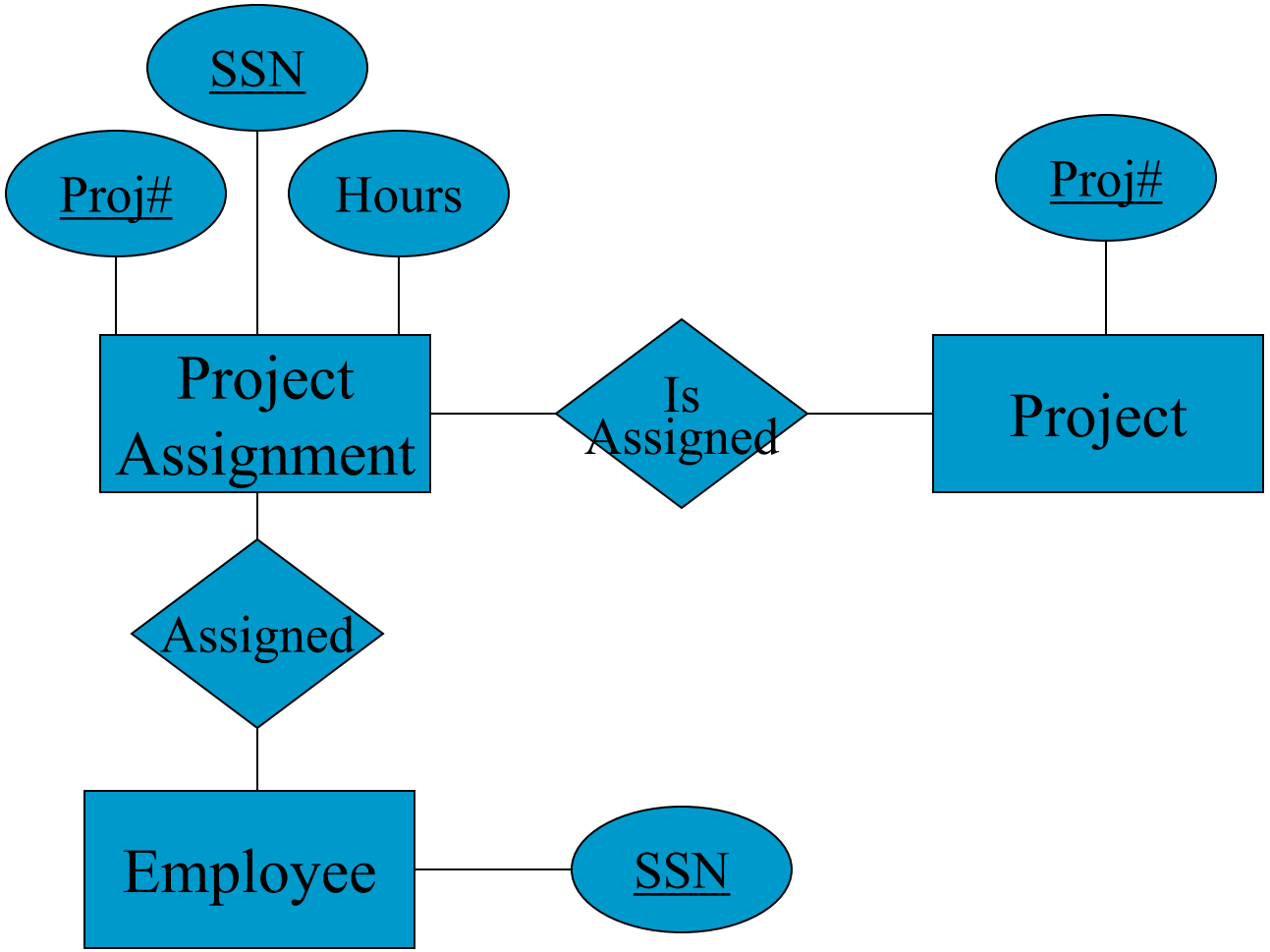



# Supertype and Subtype Entities



# Many to Many Relationships



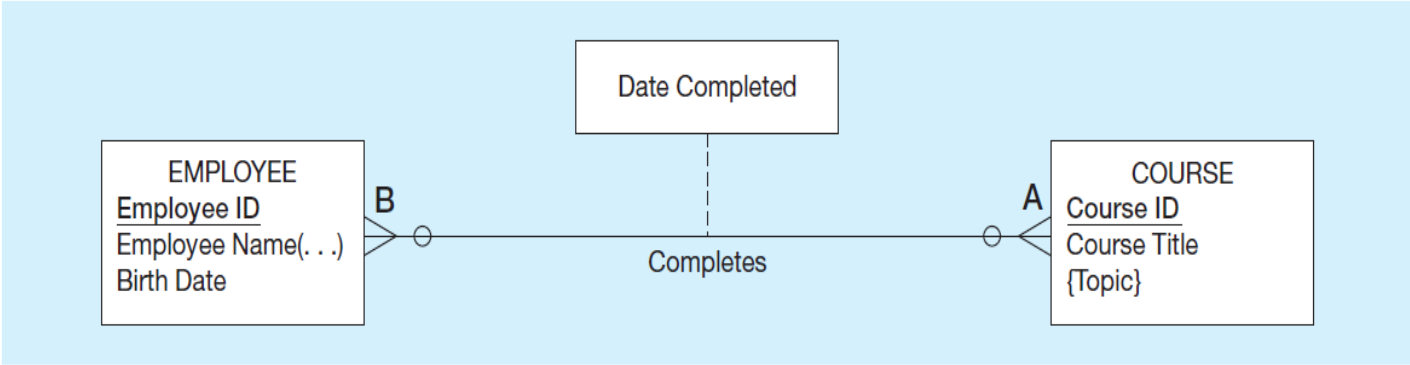
# Associative Entities

- When should a **relationship with attributes** instead be an **associative entity**?
  - All relationships for the associative entity should be many
  - The associative entity could have meaning independent of the other entities
  - The associative entity preferably has a unique identifier, and should also have other attributes
  - The associative entity may participate in other relationships other than the entities of the associated relationship
  - Convert ternary relationships to associative entities

# Associative Entities

An associative entity is an entity. It has attributes. It is also a relationship. It serves to link other entities together in a many-to-many relationship.

(a) Attribute on a relationship




(b) An associative entity (CERTIFICATE)

# Next Lecture

- Designing the Conceptual Model for the Diveshop Database
  - More on ER modeling
- SQL!