Chapter 4
Enhanced Entity-Relationship Modeling

Enhanced-ER (EER) Model Concepts
- Includes all modeling concepts of basic ER
- Additional concepts: subclasses/superclasses, specialization/generalization, categories, attribute inheritance
- The resulting model is called the enhanced-ER or Extended ER (E2R or EER) model
- It is used to model applications more completely and accurately if needed
- It includes some object-oriented concepts, such as inheritance

Subclasses and Superclasses
- An entity type may have additional meaningful subgroupings of its entities
- Example: EMPLOYEE may be further grouped into SECRETARY, ENGINEER, MANAGER, TECHNICIAN, SALARIED_EMPLOYEE, HOURLY_EMPLOYEE,
  - Each of these groupings is a subset of EMPLOYEE entities
  - Each is called a subclass of EMPLOYEE
  - EMPLOYEE is the superclass for each of these subclasses
- These are called superclass/subclass relationships.
- Example: EMPLOYEE/SECRETARY, EMPLOYEE/TECHNICIAN
Subclasses and Superclasses

- These are also called IS-A relationships (SECRETARY IS-A EMPLOYEE, TECHNICIAN IS-A EMPLOYEE, ...).
- Note: An entity that is member of a subclass represents the same real-world entity as some member of the superclass.
  - The Subclass member is the same entity in a distinct specific role.
  - An entity cannot exist in the database merely by being a member of a subclass; it must also be a member of the superclass.
  - A member of the superclass can be optionally included as a member of any number of its subclasses.
- Example: A salaried employee who is also an engineer belongs to the two subclasses ENGINEER and SALARIED_EMPLOYEE.
  - It is not necessary that every entity in a superclass be a member of some subclass.

Subclasses and Specialization

Attribute Inheritance in Superclass / Subclass Relationships

- An entity that is a member of a subclass inherits all attributes of the entity as a member of the superclass.
- It also inherits all relationships.
Specialization

- Is the process of defining a set of subclasses of a superclass.
- The set of subclasses is based upon some distinguishing characteristics of the entities in the superclass.
- Example: (SECRETARY, ENGINEER, TECHNICIAN) is a specialization of EMPLOYEE based upon job type.
- May have several specializations of the same superclass.
- Example: Another specialization of EMPLOYEE based on method of pay is (SALARIED_EMPLOYEE, HOURLY_EMPLOYEE).
- Superclass/subclass relationships and specialization can be diagrammatically represented in EER diagrams.
- Attributes of a subclass are called specific attributes.
- The subclass can participate in specific relationship types. For example, BELONGS_TO of HOURLY_EMPLOYEE.

Example of a Specialization

Instances of Specialization
Generalization

- The reverse of the specialization process
- Several classes with common features are generalized into a superclass; original classes become its subclasses
- Example: CAR, TRUCK generalized into VEHICLE; both CAR, TRUCK become subclasses of the superclass VEHICLE.
  - We can view (CAR, TRUCK) as a specialization of VEHICLE
  - Alternatively, we can view VEHICLE as a generalization of CAR and TRUCK

Diagrammatic notation sometimes used to distinguish between generalization and specialization
- Arrow pointing to the generalized superclass represents a generalization
- Arrows pointing to the specialized subclasses represent a specialization
- Because it is often subjective as to which process is more appropriate for a particular situation, it is common not to draw any arrows in these situations

Data Modeling with Specialization and Generalization
- A superclass or subclass represents a set of entities
- Shown in rectangles in EER diagrams (as are entity types)
- Sometimes, all entity sets are simply called classes, whether they are entity types, superclasses, or subclasses
Constraints on Specialization and Generalization

- If we can determine exactly those entities that will become members of each subclass by a condition, the subclasses are called predicate-defined (or condition-defined) subclasses.
  - Condition is a constraint that determines subclass members.
  - Display a predicate-defined subclass by writing the predicate condition next to the line attaching the subclass to its superclass.

- If all subclasses in a specialization have membership condition on same attribute of the superclass, specialization is called an attribute-defined specialization.
  - Attribute is called the defining attribute of the specialization.
  - Example: JobType is the defining attribute of the specialization {SECRETARY, TECHNICIAN, ENGINEER} of EMPLOYEE.

- If no condition determines membership, the subclass is called user-defined.
  - Membership in a subclass is determined by the database users by applying an operation to add an entity to the subclass.
  - Membership in the subclass is specified individually for each entity in the superclass by the user.

Two other conditions apply to a specialization/generalization:

- **Disjointness Constraint:**
  - Specifies that the subclasses of the specialization must be disjointed (an entity can be a member of at most one of the subclasses of the specialization).
  - Specified by a in EER diagram.
  - If not disjointed, overlap; that is the same entity may be a member of more than one subclass of the specialization.

- **Completeness Constraint:**
  - Total specifies that every entity in the superclass must be a member of some subclass in the specialization/generalization.
  - Shown in EER diagrams by a double line.
  - Partial allows an entity not to belong to any of the subclasses.
  - Shown in EER diagrams by a single line.

Attribute-Defined Specialization

![Diagram of Attribute-Defined Specialization]

- EMPLOYEE
- JobType: ‘Secretary’, ‘Technician’, ‘Engineer’
- JobType: ‘Engineer’
- JobType: ‘Secretary’
- JobType: ‘Technician’
- EMPLOYEE
- Name, Address, Phone, Title, BirthDate, JobType
Overlapping (Non-disjoint) Specialization

Constraints on Specialization and Generalization

- Hence, we have four types of specialization/generalization:
  - Disjoint, total
  - Disjoint, partial
  - Overlapping, total
  - Overlapping, partial
- Note: Generalization usually is total because the superclass is derived from the subclasses.

Example of disjoint partial Specialization
Specialization / Generalization
Hierarchies, Lattices and Shared Subclasses

- A subclass may itself have further subclasses specified on it
- Forms a hierarchy or a lattice
- Hierarchy has a constraint that every subclass has only one superclass (called single inheritance)
- In a lattice, a subclass can be a subclass of more than one superclass (called multiple inheritance)
- In a lattice or hierarchy, a subclass inherits attributes not only of its direct superclass, but also of all its predecessor superclasses
- A subclass with more than one superclass is called a shared subclass
- Can have specialization hierarchies or lattices, or generalization hierarchies or lattices
- In specialization, start with an entity type and then define subclasses of the entity type by successive specialization (top down conceptual refinement process)
- In generalization, start with many entity types and generalize those that have common properties (bottom up conceptual synthesis process)
- In practice, a combination of the two processes is employed

Specialization Lattice with Shared Subclass

Specialization / Generalization Lattice Example (COLLEGE)
Categories (UNION TYPES)

- All of the superclass/subclass relationships we have seen thus far have a single superclass.
- A shared subclass is a subclass in more than one distinct superclass/subclass relationship, where each relationship has a single superclass (multiple inheritance).
- In some cases, need to model a single superclass/subclass relationship with more than one superclass.
- Superclasses represent different entity types.
- Such a subclass is called a category or UNION TYPE.
- Example: Database for vehicle registration, vehicle owner can be a person, a bank (holding a lien on a vehicle), or a company.
  - Category (subclass) OWNER is a subset of the union of the three superclasses COMPANY, BANK, and PERSON.
  - A category member must exist in at least one of its superclasses.
- Note: The difference from shared subclass, which is a subset of the intersection of its superclasses (shared subclass member must exist in all of its superclasses).

Example of categories (UNION TYPES)

- EER Diagram:
  - College
The primary goal for a database is to store data about some domain for retrieval. The primary goal for knowledge representation is to store “knowledge” about some domain for use in inferencing of further knowledge about the domain.

Both use abstraction to identify common properties of the domain. Both provide similar concepts, constraints, operations and languages for defining data/knowledge.
Differences between DBs and KR

- KR is broader in scope incorporating forms of knowledge such as rules, and defaults
- KR uses the stored facts to inference further facts
- KR uses less formal representations of data/facts often mixing schemas and instances in order to facilitate reasoning
- The last point results in less efficient representations in KR systems

Ontologies

- Similar to a conceptual schema but with more knowledge, rules, and exceptions
- Specification of a conceptualization
- Used by the semantic web to facilitate retrieval of information
- Forms:
  - Thesaurus
  - Taxonomy
  - Database schema

Use conceptual modeling and other tools to develop “a specification of a conceptualization”

- Specification refers to the language and vocabulary (data model concepts) used
- Conceptualization refers to the description (schema) of the concepts of a particular field of knowledge and the relationships among these concepts
- Many medical, scientific, and engineering Ontologies are being developed as a means of standardizing concepts and terminology
KR Desirable Features

- Must be based on human expertise
- Should be enable presentation of intermediate results
- Should enable answering questions about reasoning processes
- Should be easy to modify

Example Knowledge-Based System Categories

- Interpretation
- Prediction
- Diagnosis
- Recommendations for remediation
- Design
- Planning
- Monitoring
- Instruction
- Control

Components of Knowledge-Based Systems

- Knowledge Base
- Inference Engine
- User Interface
- Knowledge Base Editor
- Case Specific Data
- Explanation sub-system
Good Problems for Knowledge-Based Systems

- There is an existing need that should be met
- Scarce human expertise
- Symbolic Reasoning is appropriate
- Well structured domain
- Cannot be solved using more traditional methods
- Availability of cooperative experts
- Proper size and scope

Knowledge Engineering

- Conducted by a knowledge engineer
- Conduct interviews with expert(s)
- Code the knowledge
- Test the system
- Modify the system
- Often has to become a quasi expert

Must take into account ……

- User Expectations
- Level of Expertise of user(s)
- Appropriate explanation for different users
- User preferences for the user interface
Difficulties

- Human skills are not always accessible to the conscious mind
- Human problem solving is not always based on a systematic method
- Expert decisions are often influenced by cultural and/or environmental factors
- Expertise changes

Model Based Reasoning

- Relies on the existence of a quantitative or qualitative model of the domain
- Very useful for diagnostic systems to simulate the actual behavior
- Appropriate for single fault situations
- Can easily be used to generate explanations

MBR in Diagnosis

- Hypothesis Formation (sometimes using some other method)
- Testing of Hypothesis (by simulating the behavior assuming the hypothesis)
- Discrimination among hypotheses (by observing the behavior under each hypothesis)