Ceng 302
Database Management Systems

The Enhanced Entity-Relationship (EER) Model

Prof. Dr. Adnan YAZICI
Department of Computer Engineering,
Middle East Technical University

(**Fall 2021**)

Enhanced ER (EER) Model

- EER stands for Enhanced ER or Extended ER
- EER Model Concepts
  - Includes all modeling concepts of basic ER
  - Additional concepts:
    - subclasses/superclasses
    - specialization/generalization
    - attribute and relationship inheritance
    - categories (UNION types)
  - These are fundamental to conceptual modeling
- The additional EER concepts are used to model applications more completely and more accurately

Enhanced Entity-Relationship Model

- **Additional entity types**
  - **Superclass**: including one or more distinct subgroups in the data model
  - **Subclass**: a distinct subgroup of an entity type in the data model
- **Attribute Inheritance**
  - **Specialization hierarchy** (specialization: maximizing the differences between members of an entity by identifying their distinguishing characteristics)
  - **Generalization hierarchy** (generalization: minimizing the differences between entities by identifying their common characteristics)
  - **Is-A hierarchy**
- **Constraints on specialization/generalization**
  - **Participation** (mandatory, optional)
  - **Disjoint**: disjoint (or), non-disjoint (and)
- **Other**
  - **Aggregation** (has a or is part of)
  - **Composition** (strong ownership of aggregation)

Subclasses and Superclasses

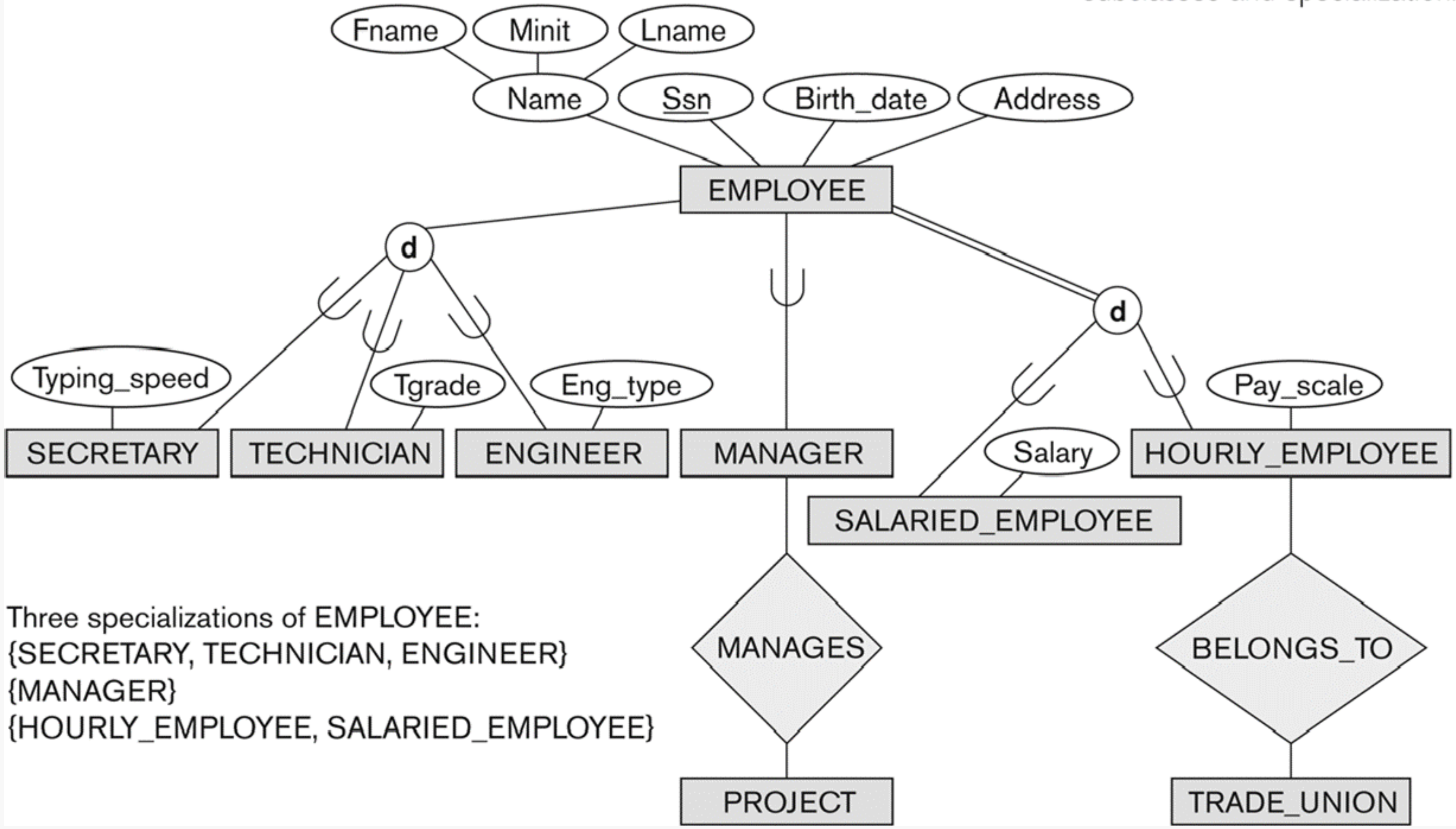
- An entity type may have additional meaningful subgroupings of its entities
    - Example: EMPLOYEE may be further grouped into:
        - SECRETARY, ENGINEER, TECHNICIAN, …
            - Based on the EMPLOYEE's Job
        - MANAGER
            - EMPLOYEEs who are managers
        - SALARIED_EMPLOYEE, HOURLY_EMPLOYEE
            - Based on the EMPLOYEE's method of pay
- EER diagrams extend ER diagrams to represent these additional subgroupings, called ***subclasses*** or ***subtypes***

Specialization

- 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

Subclasses and Superclasses

Figure 4.1
EER diagram notation to represent subclasses and specialization.



Three specializations of EMPLOYEE:
{SECRETARY, TECHNICIAN, ENGINEER}
{MANAGER}
{HOURLY_EMPLOYEE, SALARIED_EMPLOYEE}

Generalization

- Generalization is 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

Generalization

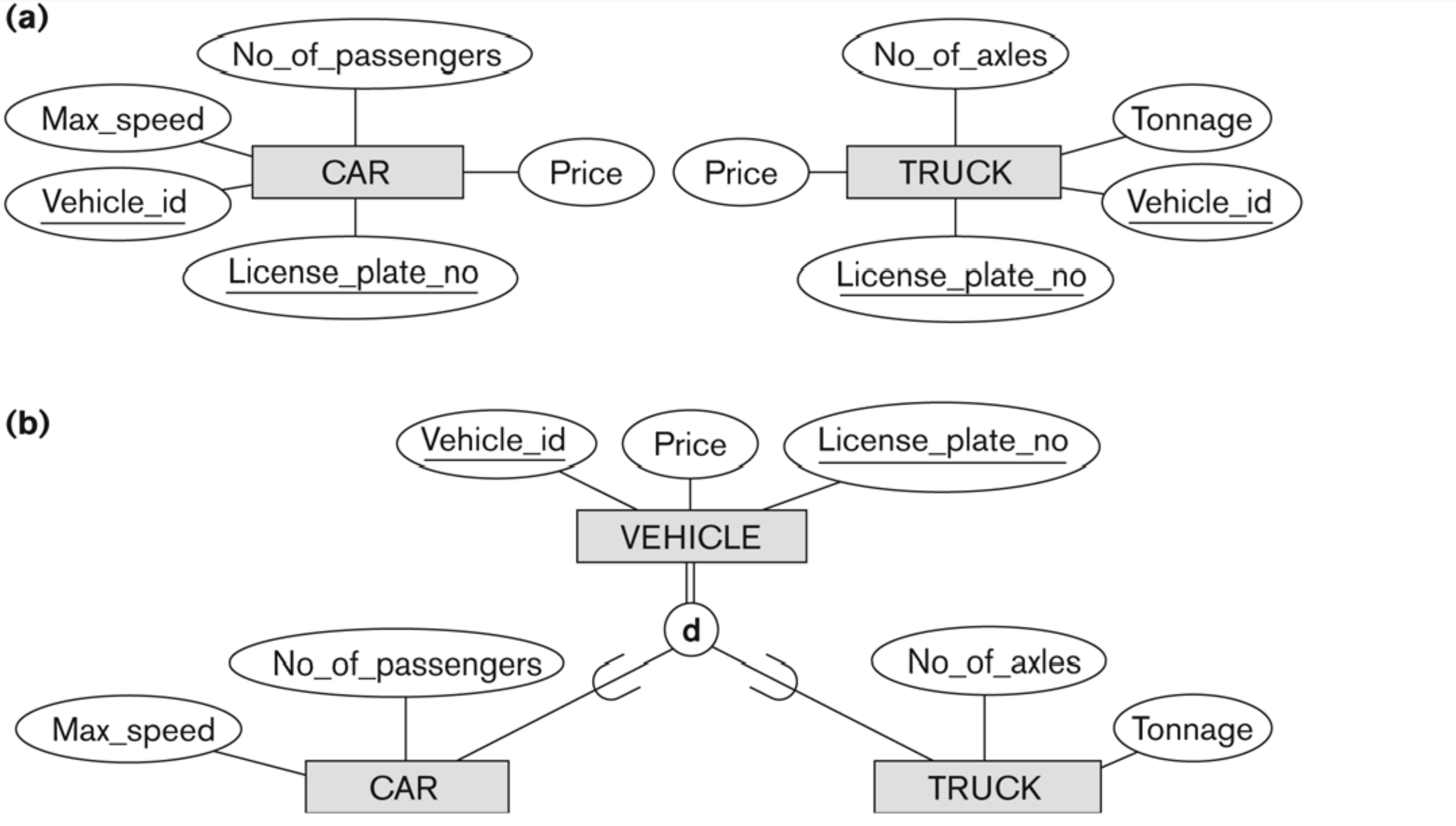


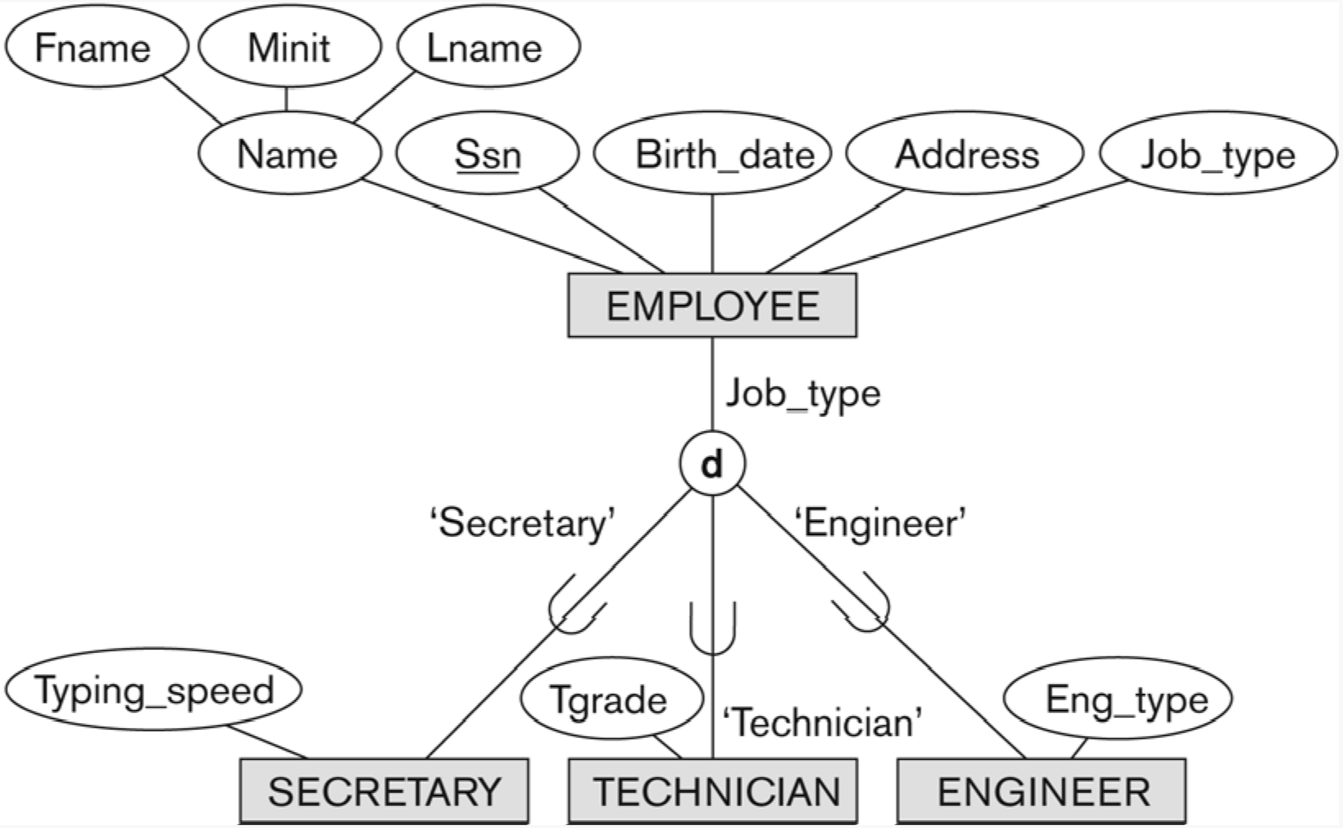
Figure 4.3
Generalization. (a) Two entity types, CAR and TRUCK.
(b) Generalizing CAR and TRUCK into the superclass VEHICLE.

Displaying an attribute-defined specialization in EER diagrams

- 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

Displaying an attribute-defined specialization in EER diagrams

Figure 4.4
EER diagram notation for an attribute-defined specialization on Job_type.



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.

Constraints on Specialization and Generalization

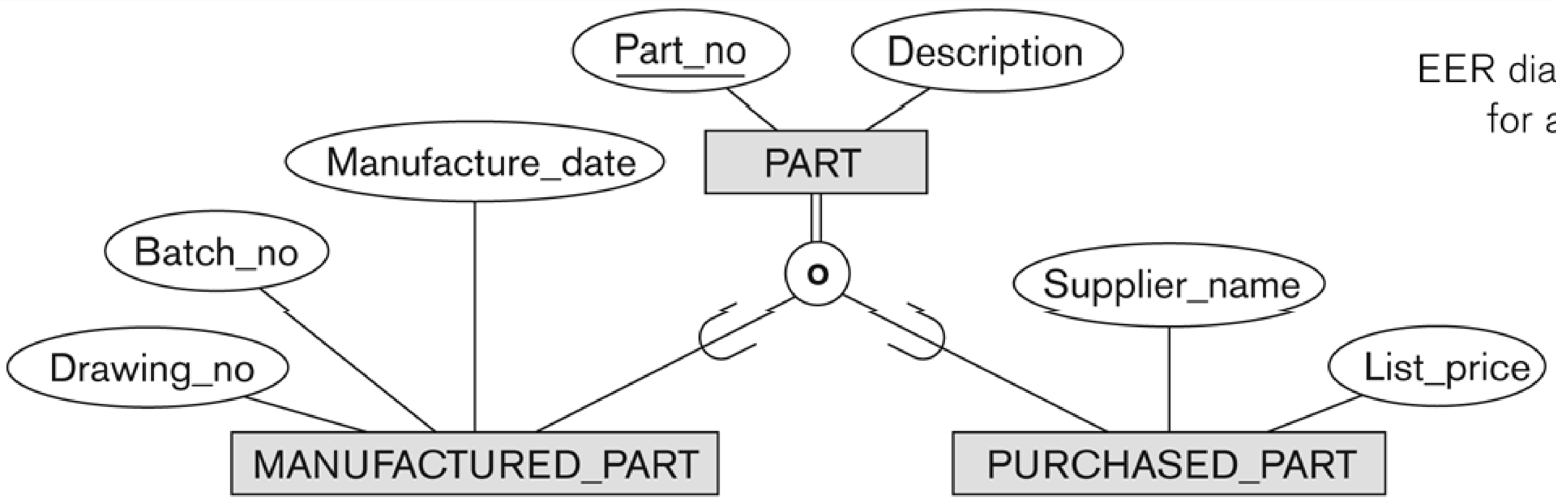


Figure 4.5
EER diagram notation for an overlapping (nondisjoint) specialization.

Specialization/Generalization Hierarchies, Lattices & 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***); this is basically a ***tree structure***
- In a ***lattice***, a subclass can be subclass of more than one superclass (called ***multiple inheritance***)

Specialization/Generalization Hierarchies, Lattices & Shared Subclasses

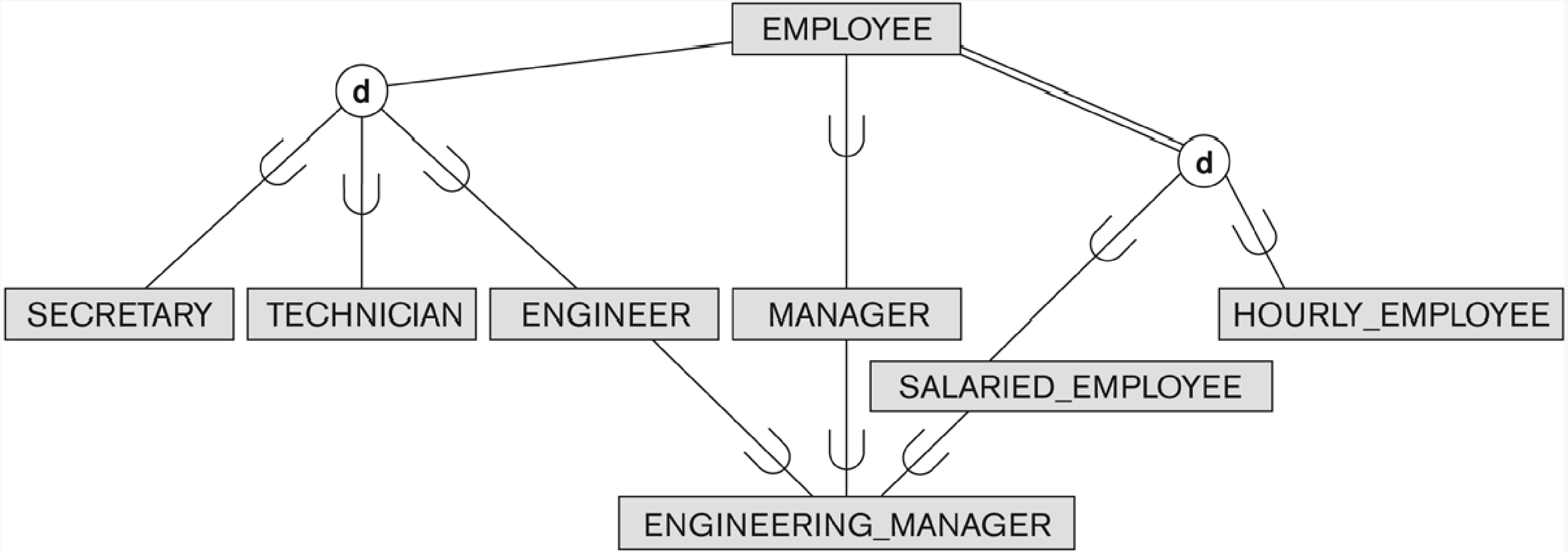


Figure 4.6
A specialization lattice with shared subclass ENGINEERING_MANAGER.

Specialization/Generalization Hierarchies, Lattices & Shared Subclasses

- 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 (multiple inheritance)

Specialization/Generalization Hierarchies, Lattices & Shared Subclasses

- In ***specialization***, start with an entity type and then define subclasses of the entity type by successive specialization
  - called a ***top down*** conceptual refinement process
- In ***generalization***, start with many entity types and generalize those that have common properties
  - Called a ***bottom up*** conceptual synthesis process
- In practice, a *combination of both processes* is usually employed

Specialization/Generalization Hierarchies, Lattices & Shared Subclasses

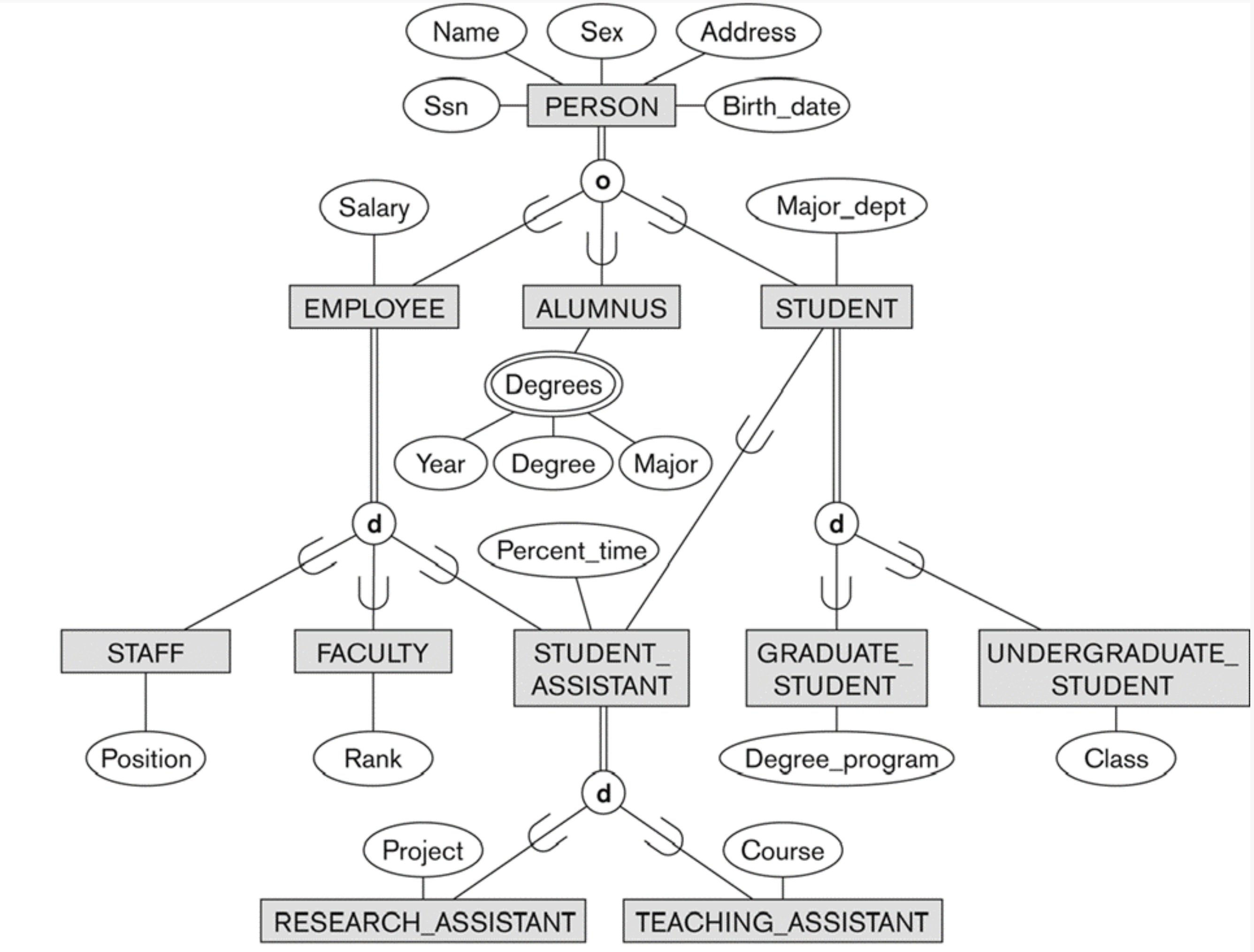


Figure 4.7
A specialization lattice with multiple inheritance for a UNIVERSITY database.

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 relationships
- In some cases, we need to model a *single superclass/subclass relationship* with *more than one* superclass
  - Superclasses can represent different entity types
  - Such a subclass is called a category or UNION TYPE

Categories (UNION TYPES)

- Example: In a database for vehicle registration, a vehicle owner can be a PERSON, a BANK (holding a lien on a vehicle) or a COMPANY.
  - A ***category*** (UNION type) called OWNER is created to represent 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
- Difference from shared subclass, which is a:
  - subset of the intersection of its superclasses
  - shared subclass member must exist in all of its superclasses

Categories (UNION TYPES)

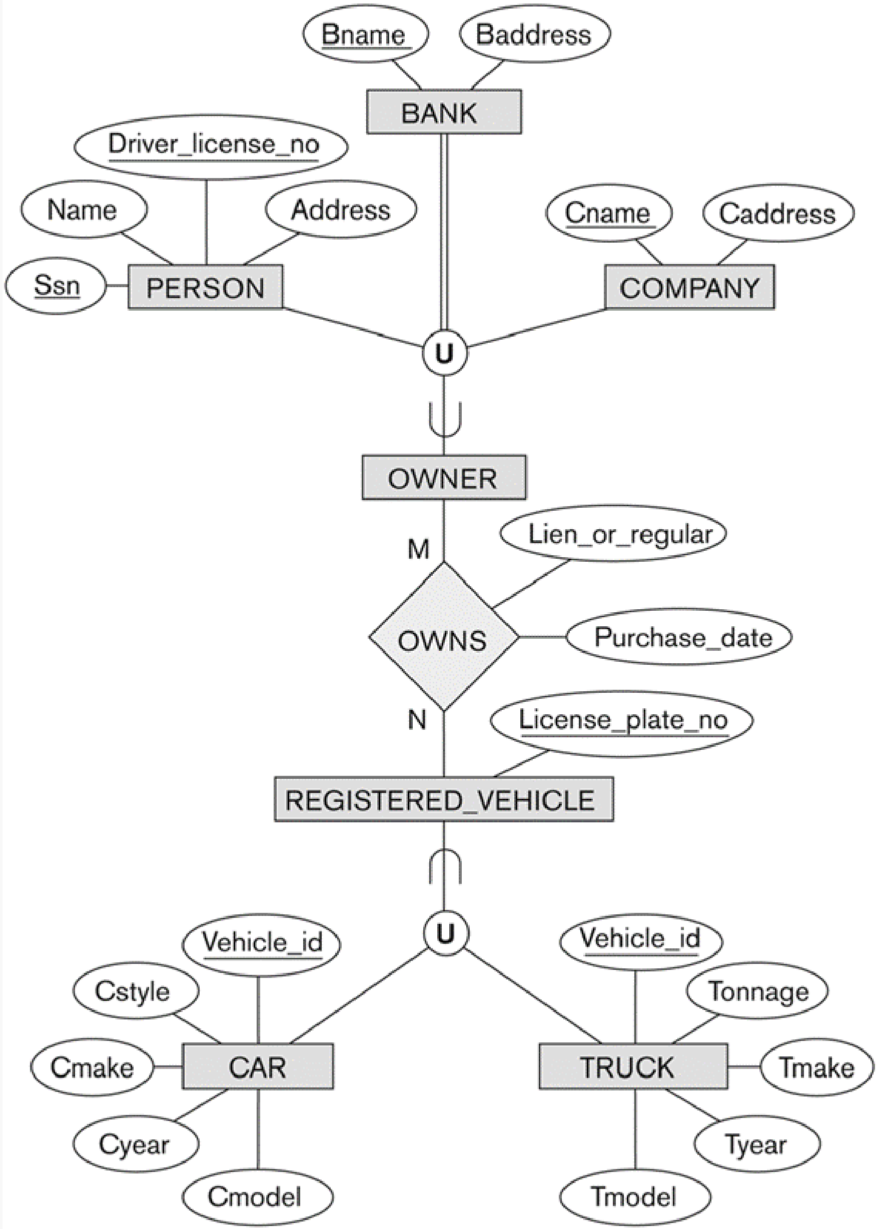


Figure 4.8
Two categories (union types): OWNER and REGISTERED_VEHICLE.

Alternative Diagrammatic Notations

- ER/EER diagrams are a specific notation for displaying the concepts of the model diagrammatically.
- DB design tools use many alternative notations for the same or similar concepts
- One popular alternative notation uses *UML class diagrams.*
- see next slides for UML class diagrams and other alternative notations

Alternative Diagrammatic Notations

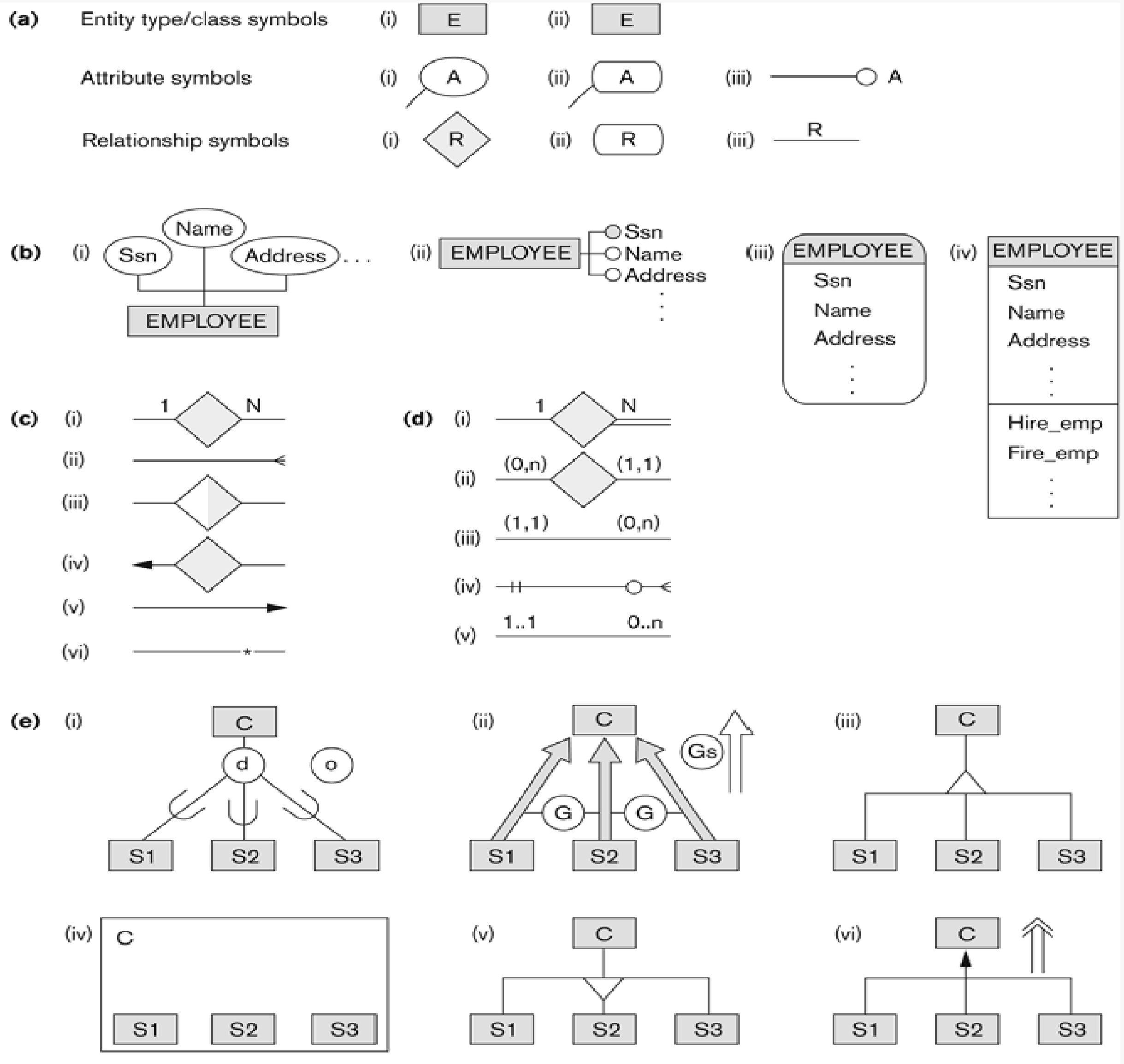
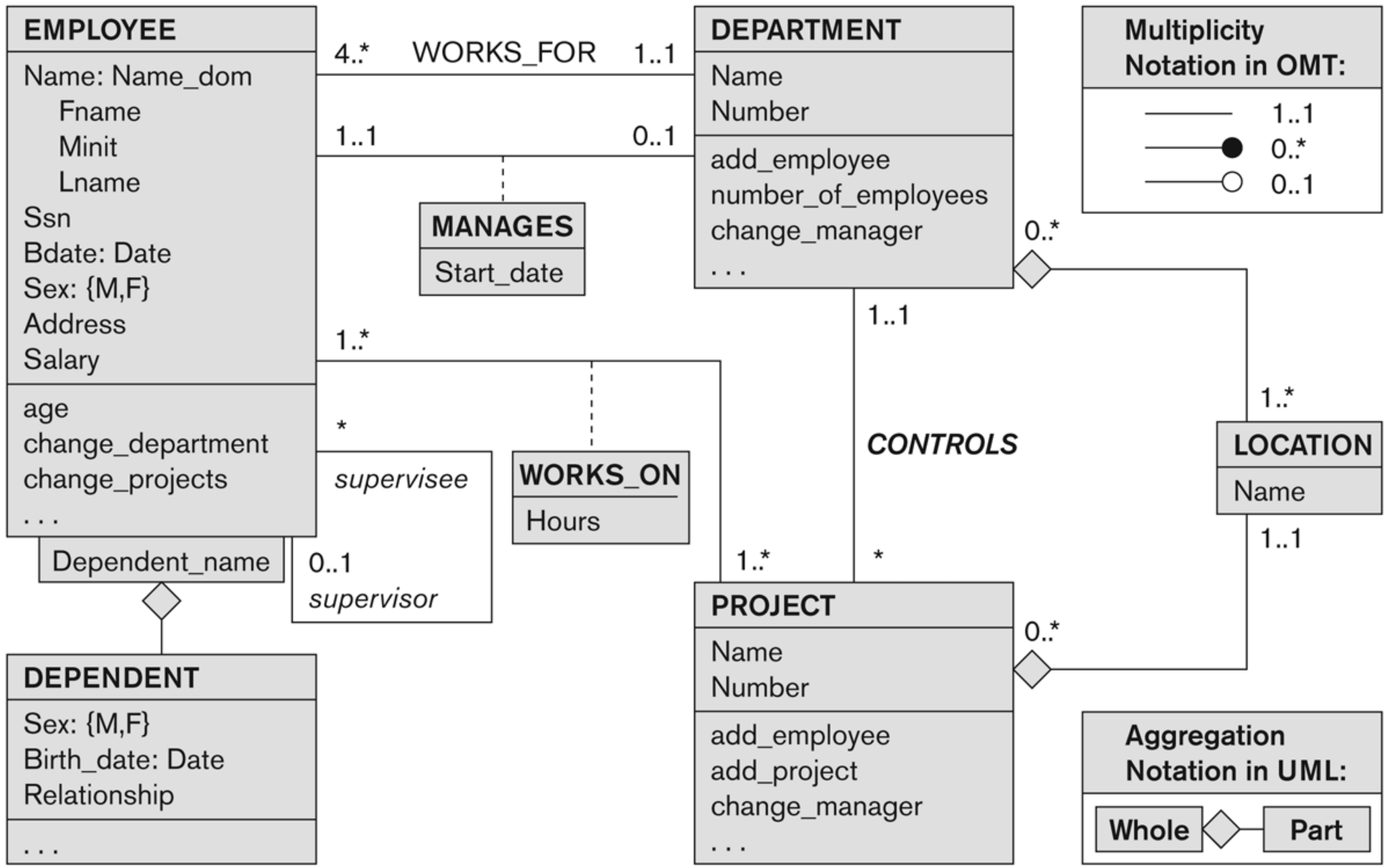


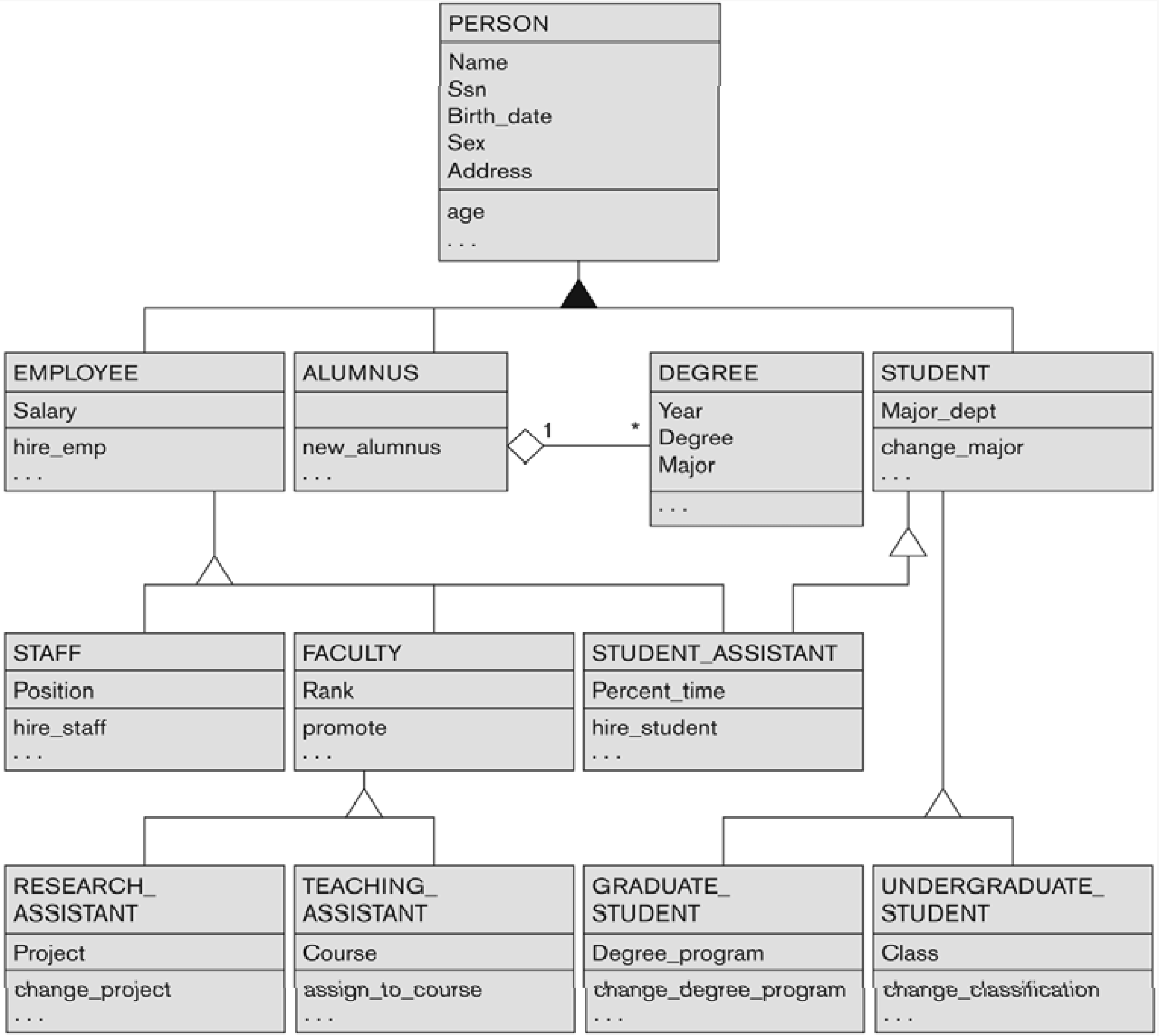
Figure A.1
Alternative notations. (a) Symbols for entity type/class, attribute, and relationship. (b) Displaying attributes. (c) Displaying cardinality ratios. (d) Various (min, max) notations. (e) Notations for displaying specialization/generalization.

UML class diagram for COMPANY database schema

Figure 3.16
The COMPANY conceptual schema in UML class diagram notation.



UML Example for Displaying Specialization / Generalization



PERSON
Name
Ssn
Birth_date
Sex
Address
age
. . .

EMPLOYEE
Salary
hire_emp
. . .

ALUMNUS
new_alumnus
. . .

1

*

DEGREE
Year
Degree
Major
. . .

STUDENT
Major_dept
change_major
. . .

STAFF
Position
hire_staff
. . .

FACULTY
Rank
promote
. . .

STUDENT_ASSISTANT
Percent_time
hire_student
. . .

RESEARCH_ASSISTANT
Project
change_project
. . .

TEACHING_ASSISTANT
Course
assign_to_course
. . .

GRADUATE_STUDENT
Degree_program
change_degree_program
. . .

UNDERGRADUATE_STUDENT
Class
change_classification
. . .

Figure 4.10
A UML class diagram corresponding to the EER diagram in Figure 4.7, illustrating UML notation for specialization/generalization.

Design Steps

- Identify
  - Entity types, relationship types
  - Cardinality and participation constraints
  - Attributes
  - Keys
  - Specialize/generalize
  - EER diagram
  - Class diagram
- EER Model examples
- UML Model examples

Data Modeling Tools

- A number of popular tools that cover conceptual modeling and mapping into relational schema design.
  - Examples: ERWin, S-Designer (Enterprise Application Suite), ER- Studio, etc.
- POSITIVES:
  - Serves as documentation of application requirements, easy user interface - mostly graphics editor support
- NEGATIVES:
  - Most tools lack a proper distinct notation for relationships with relationship attributes
  - Mostly represent a relational design in a diagrammatic form rather than a conceptual EER-based design.

Some of the Currently Available Automated Database Design Tools

COMPANY	TOOL	FUNCTIONALITY
Embarcadero Technologies	**ER Studio**	Database Modeling in ER and IDEF1X
	DB Artisan	Database administration, space and security management
Oracle	**Developer 2000/Designer 2000**	Database modeling, application development
Popkin Software	**System Architect 2001**	Data modeling, object modeling, process modeling, structured analysis/design
Platinum (Computer Associates)	**Enterprise Modeling Suite: Erwin, BPWin, Paradigm Plus**	Data, process, and business component modeling
Persistence Inc.	**Pwertier**	Mapping from O-O to relational model
Rational (IBM)	**Rational Rose**	UML Modeling & application generation in C++/JAVA
Resolution Ltd.	**Xcase**	Conceptual modeling up to code maintenance
Sybase	**Enterprise Application Suite**	Data modeling, business logic modeling
Visio	**Visio Enterprise**	Data modeling, design/reengineering Visual Basic/C++

Summary of Conceptual Design

- *Conceptual design* follows *requirements analysis*,
  - Yields a high-level description of data to be stored
- EER model popular for conceptual design
  - Constructs are expressive, close to the way people think about their applications.
- Basic constructs: *entities*, *relationships*, and *attributes* (of entities and relationships).
- Some additional constructs: *weak entities*, *ISA hierarchies*, and *aggregation*.
- Note: There are many variations on EER model.

Summary of EER (Contd.)

- Several kinds of integrity constraints can be expressed in the EER model: **key constraints, participation constraints, and overlap/covering constraints** for ISA hierarchies. Some **foreign key constraints** are also implicit in the definition of a relationship set.
  - Some constraints (notably, **functional dependencies**) cannot be expressed in the EER model.
  - Constraints play an important role in determining the best database design for an enterprise.

Summary of EER (Contd.)

- EER design is *subjective*. There are often many ways to model a given scenario! Analyzing alternatives can be tricky, especially for a large enterprise. Common choices include:
  - Entity vs. attribute, entity vs. relationship, binary or n-ary relationship, whether or not to use ISA hierarchies, and whether or not to use aggregation.
- Ensuring good database design: resulting relational schema should be analyzed and refined further. FD information and normalization techniques are especially very useful.