



M.Sc. in Computer Science Department of Computer Science University of Sri Jayewardenepura

CCS1522 | CSE1522| CIS1522 Database Management
Systems

Presented By:

Surani Tissera(PhD)

Department of Computer Science

Data Modeling Using the Entity-Relationship (ER) Model



Content outline

- Overview of Database Design Process
- Example Database Application (COMPANY)
- ER Model Concepts
 - Entities and Attributes
 - Entity Types, Value Sets, and Key Attributes
 - Relationships and Relationship Types
 - Weak Entity Types
 - Roles and Attributes in Relationship Types
- ER Diagrams - Notation
- ER Diagram for COMPANY Schema
- Alternative Notations – UML class diagrams, others

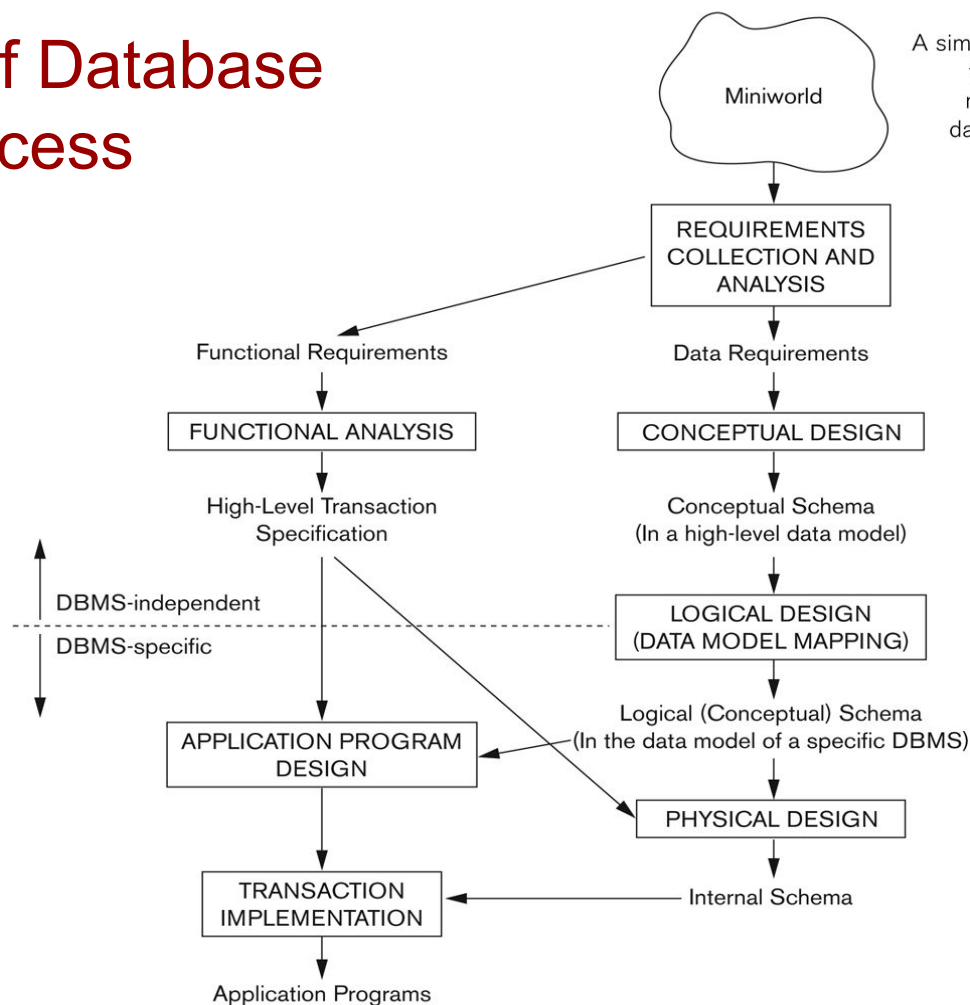


Overview of Database Design Process

- Two main activities:
 - Database design
 - Applications design
- Focus in this lesson on database design
 - To design the conceptual schema for a database application
- Applications design focuses on the programs and interfaces that access the database
 - Generally considered part of software engineering



Overview of Database Design Process



The Entity Relationship Model (ERM)

- ER model forms the basis of an ER diagram
- ERD represents conceptual database as viewed by end user
- ERDs depict database's main components:
 - Entities
 - Attributes
 - Relationships



Entity



Entities

- Entity is an object of interest to the end user
- Refers to entity set and not to single entity occurrence (entity instance)
- Corresponds to table and not to row in relational environment
- In Chen and Crow's Foot models, entity is represented by rectangle with entity's name
- Entity name, a noun, written in capital letters



Entities-Example

- Entities are specific objects or things in the mini-world that are represented in the database.
 - For example the EMPLOYEE John Smith, the Research DEPARTMENT, the ProductX PROJECT



Entities- Exercise

Imagine you are designing a database for a library system. The library wants to keep track of information related to books, authors, and library members. Your task is to identify the entities in this system.



Weak Entity



WEAK ENTITY

An entity that cannot be uniquely identified by its attributes alone. The existence of a weak entity is dependent upon another entity called the owner entity. The weak entity's identifier is a combination of the identifier of the owner entity and the partial key of the weak entity.

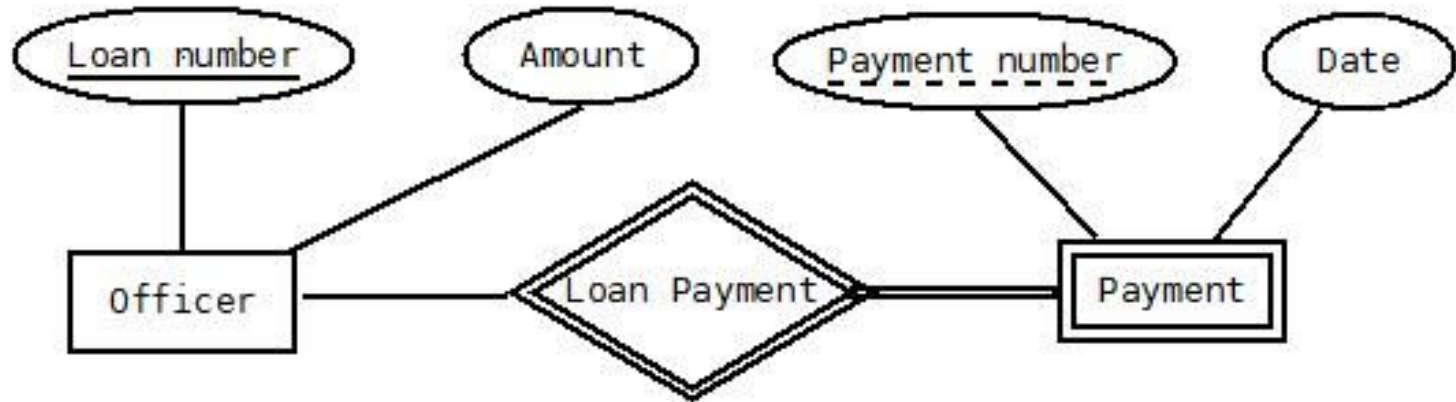


Weak Entities

- **Weak entity** meets two conditions
 - Existence-dependent
 - Primary key partially or totally derived from parent entity in relationship
- Database designer determines whether an entity is weak based on business rules



Weak Entity



Entities- Exercise

The HEG has 12 instructors and can handle up to 30 trainees per class. HEG offers five Advanced Technology courses, each of which may generate several classes. If a class has fewer than 10 trainees, it will be canceled. Therefore, it is possible for a course not to generate any classes. Each class is taught by one instructor. Each instructor may teach up to two classes or may be assigned to do research only. Each trainee may take up to two classes per year.

Your task is to identify the entities in this system



Entities- Exercise

Automata, Inc. produces specialty vehicles by contract. The company operates several departments, each of which builds a particular vehicle, such as a limousine, a truck, a van, or an RV.

Before a new vehicle is built, the department places an order with the purchasing department to request specific components. Automata's purchasing department is interested in creating a database to keep track of orders and to accelerate the process of delivering materials.

The order received by the purchasing department may contain several different items. An inventory is maintained so that the most frequently requested items are delivered almost immediately. When an order comes in, it is checked to determine whether the requested item is in inventory. If an item is not in inventory, it must be ordered from a supplier. Each item may have several suppliers.

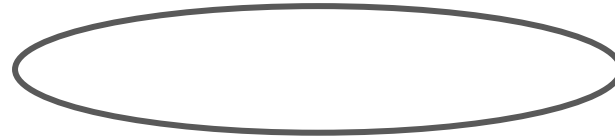


Attributes



Attributes

- Characteristics of entities
- Chen notation: attributes represented by ovals connected to entity rectangle with a line
 - Each oval contains the name of attribute it represents

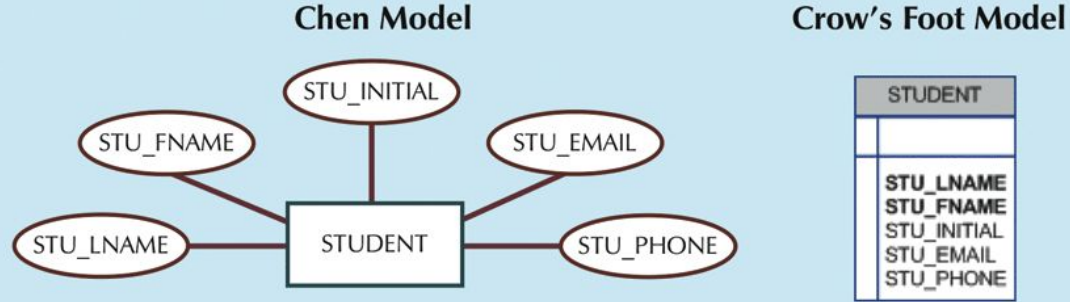


- Crow's Foot notation: attributes written in attribute box below entity rectangle



Attributes-Example

FIGURE 4.1 The attributes of the STUDENT entity: Chen and Crow's Foot



Attributes

- **Required Attributes**
 - an attribute that must have a value. Cannot be null.
- **Optional Attribute**
 - an attribute that does not require a value. Can be left empty.
- **Has a domain**
 - a domain is the set of possible values for a given attribute.



Domain

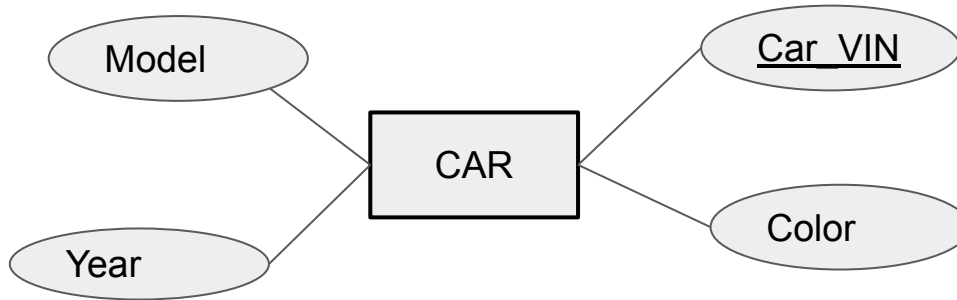
- **Attribute has a domain**
 - a domain is the set of possible values for a given attribute.
 - the domain for the grade point average (GPA) is (0,4)
 - gender (M,F)



Identifiers (Key attribute)

KEY ATTRIBUTE

- one or more attributes that uniquely identify each entity instance
- are **underlined** in the ERD.

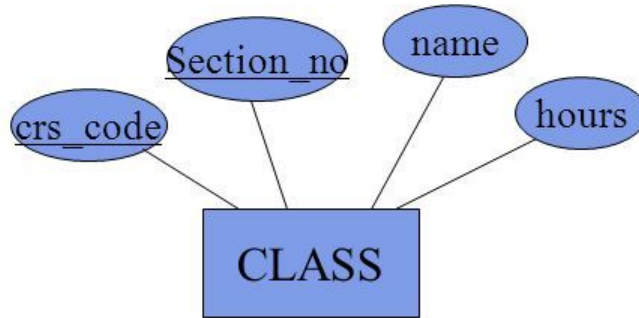


(Each car is identified by a unique vehicle identification number, or CAR_VIN)



Composite Identifiers

Primary key composed of more than one attribute



Composite primary key is the combination of CRS_CODE and CLASS_SECTION



The CLASS table (entity) components and contents

CLASS_CODE	CRS_CODE	CLASS_SECTION	CLASS_TIME	ROOM_CODE	PROF_NUM
10012	ACCT-211	1	MWF 8:00-8:50 a.m.	BUS311	105
10013	ACCT-211	2	MWF 9:00-9:50 a.m.	BUS200	105
10014	ACCT-211	3	TTh 2:30-3:45 p.m.	BUS252	342
10015	ACCT-212	1	MWF 10:00-10:50 a.m.	BUS311	301
10016	ACCT-212	2	Th 6:00-8:40 p.m.	BUS252	301
10017	CIS-220	1	MWF 9:00-9:50 a.m.	KLR209	228
10018	CIS-220	2	MWF 9:00-9:50 a.m.	KLR211	114
10019	CIS-220	3	MWF 10:00-10:50 a.m.	KLR209	228
10020	CIS-420	1	W 6:00-8:40 p.m.	KLR209	162
10021	QM-261	1	MWF 8:00-8:50 a.m.	KLR200	114
10022	QM-261	2	TTh 1:00-2:15 p.m.	KLR200	114
10023	QM-362	1	MWF 11:00-11:50 a.m.	KLR200	162
10024	QM-362	2	TTh 2:30-3:45 p.m.	KLR200	162
10025	MATH-243	1	Th 6:00-8:40 p.m.	DRE155	325



Partial key attribute



PARTIAL KEY

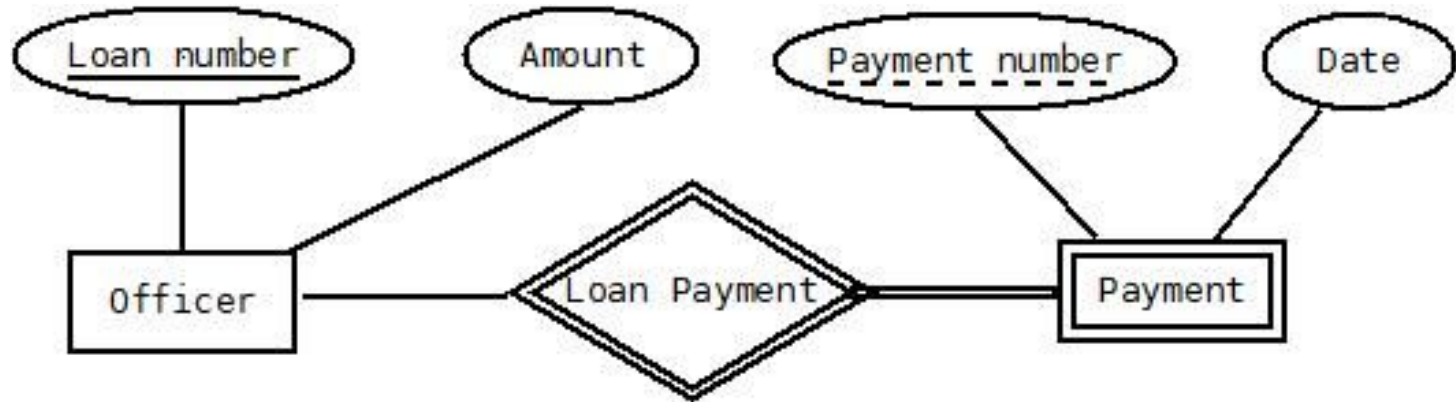
ATTRIBUTE

An attribute that, when combined with the key attribute of the owner entity, provides a unique identification for the weak entity.

We underline the discriminator with a dashed line.



Weak Entity & Partial Key Attribute



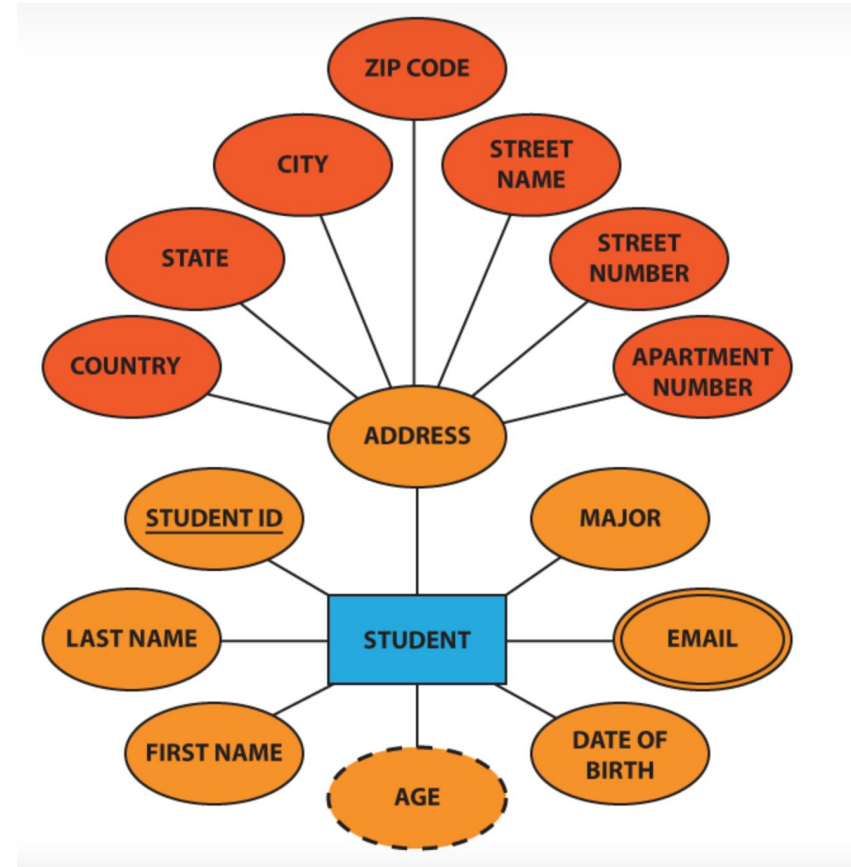
Attributes (cont'd.)

- **Composite attribute** can be subdivided
 - ADDRESS can be subdivided into street, city, state, and zip code.
 - PHONE_NUMBER can be subdivided into area code and exchange number
- **Simple attribute** cannot be subdivided
 - age, sex, and marital status




Attributes (cont'd.)

The composite attribute “address” can be subdivided into street name, street number, apartment number, city, state, zip code, and country.

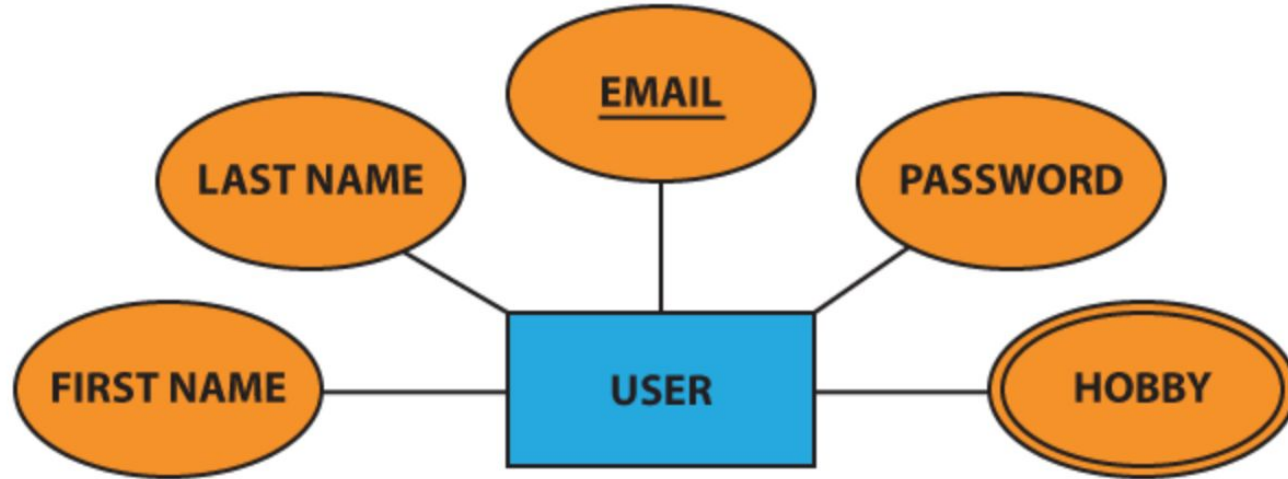


Attributes (cont'd.)

- **Single-value attribute** can have only a single value
 - a person can have only one National Identification Number (or NIC)
 - a manufactured part can have only one serial number.
- **Multivalued attributes** can have many values 
 - a person may have several college degrees
 - a household may have several different phones



Attributes (cont'd.)



“hobby” can be considered as a multivalued attribute for the “user” entity



Types of Attributes

- In general, composite and multi-valued attributes may be nested arbitrarily to any number of levels, although this is rare.
 - For example, PreviousDegrees of a STUDENT is a composite multi-valued attribute denoted by {PreviousDegrees (College, Year, Degree, Field)}
 - Multiple PreviousDegrees values can exist
 - Each has four subcomponent attributes:
 - College, Year, Degree, Field

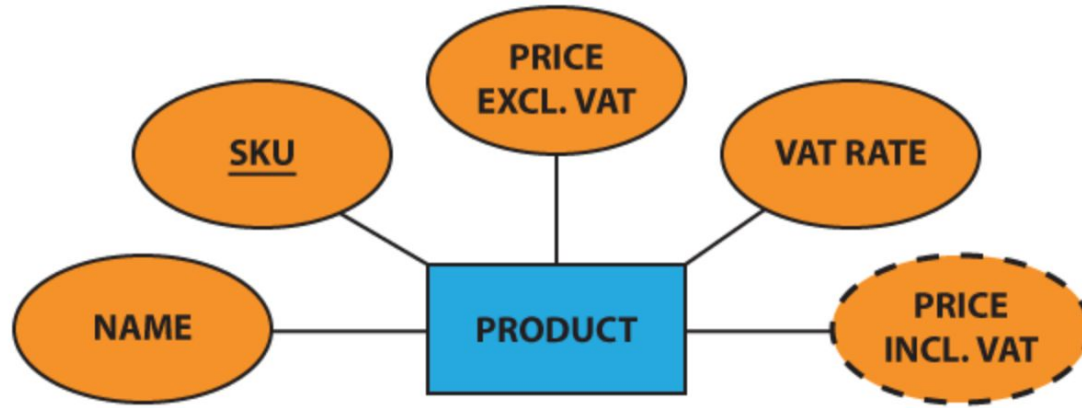


Attributes (cont'd.)



- **Derived attribute (computed attributes):** value may be calculated from other attributes
 - Need not be physically stored within database
 - an employee's age may be found by computing the difference between the current date and the EMP_DOB
 - total cost of an order can be derived by multiplying the quantity ordered by the unit price





Having given the price excluding VAT and the VAT rate, we can calculate the price including VAT



TABLE 4.2 Advantages and Disadvantages of Storing Derived Attributes

	DERIVED ATTRIBUTE	
	STORED	NOT STORED
Advantage	Saves CPU processing cycles Saves data access time Data value is readily available Can be used to keep track of historical data	Saves storage space Computation always yields current value
Disadvantage	Requires constant maintenance to ensure derived value is current, especially if any values used in the calculation change	Uses CPU processing cycles Increases data access time Adds coding complexity to queries



Entity Type CAR with two keys and a corresponding Entity Set

(a)

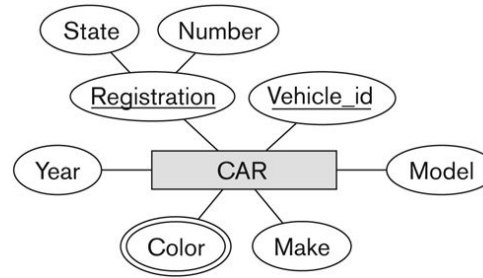


Figure 3.7

The CAR entity type with two key attributes, Registration and Vehicle_id. (a) ER diagram notation. (b) Entity set with three entities.

(b)

CAR
Registration (Number, State), Vehicle_id, Make, Model, Year, {Color}

CAR₁
((ABC 123, TEXAS), TK629, Ford Mustang, convertible, 2004 {red, black})

CAR₂
((ABC 123, NEW YORK), WP9872, Nissan Maxima, 4-door, 2005, {blue})

CAR₃
((VSY 720, TEXAS), TD729, Chrysler LeBaron, 4-door, 2002, {white, blue})

⋮



COMPANY Database- Example

- We need to create a database schema design based on the following (simplified) **requirements** of the COMPANY Database:
 - The company is organized into DEPARTMENTS. Each department has a name, number and an employee who *manages* the department. We keep track of the start date of the department manager. A department may have several locations.
 - Each department *controls* a number of PROJECTs. Each project has a unique name, unique number and is located at a single location.



Example COMPANY Database

- We store each EMPLOYEE's social security number, address, salary, sex, and birthdate.
 - Each employee *works for* one department but may *work on* several projects.
 - We keep track of the number of hours per week that an employee currently works on each project.
 - We also keep track of the *direct supervisor* of each employee.
- Each employee may *have* a number of DEPENDENTS.
 - For each dependent, we keep track of their name, sex, birthdate, and relationship to the employee.



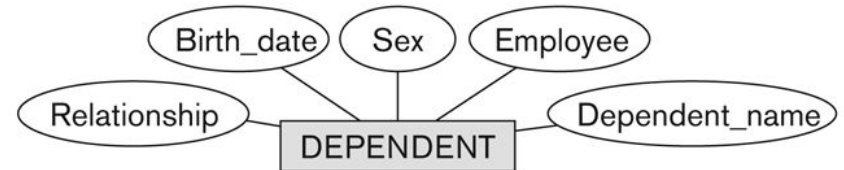
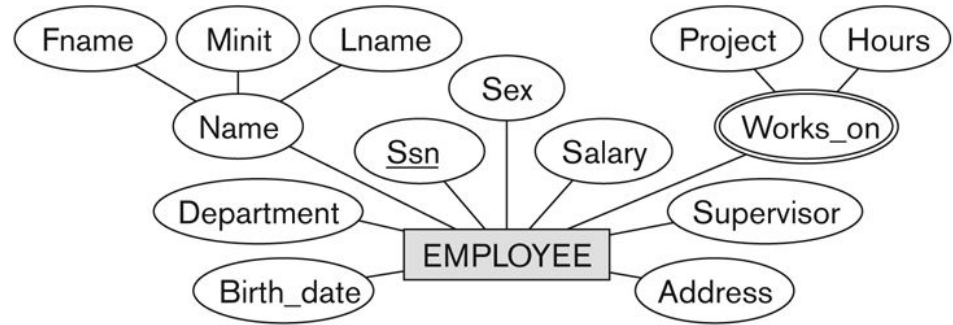
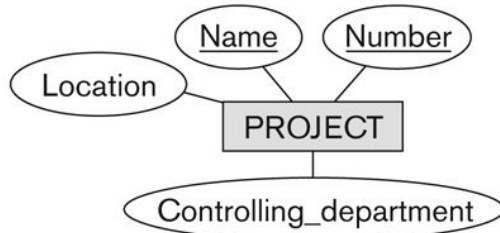
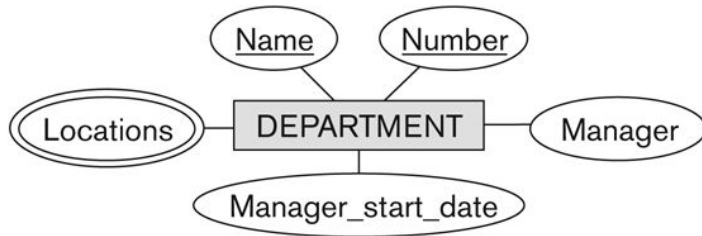
Initial Design of Entity Types for the COMPANY Database Schema

- Based on the requirements, we can identify four initial entity types in the COMPANY database:
 - DEPARTMENT
 - PROJECT
 - EMPLOYEE
 - DEPENDENT



Initial Design of Entity Types:

EMPLOYEE, DEPARTMENT, PROJECT, DEPENDENT



Relationships



Relationships and Relationship Types

- A **relationship** relates two or more distinct entities with a specific meaning. For example,
 - EMPLOYEE John Smith **works** on the ProductX PROJECT, or
 - EMPLOYEE Franklin Wong **manages** the Research DEPARTMENT.
- Relationships of the same type are grouped or typed into a **relationship type**.
 - The **WORKS_ON** relationship type in which EMPLOYEES and PROJECTs participate, or
 - The **MANAGES** relationship type in which EMPLOYEES and DEPARTMENTs participate.



Relationships and Relationship Types

- The degree of a relationship type is **the number of participating entity types**.
 - Both MANAGES and WORKS_ON are *binary* relationships.



Relationship instances of the WORKS_FOR

Relationship between EMPLOYEE and DEPARTMENT

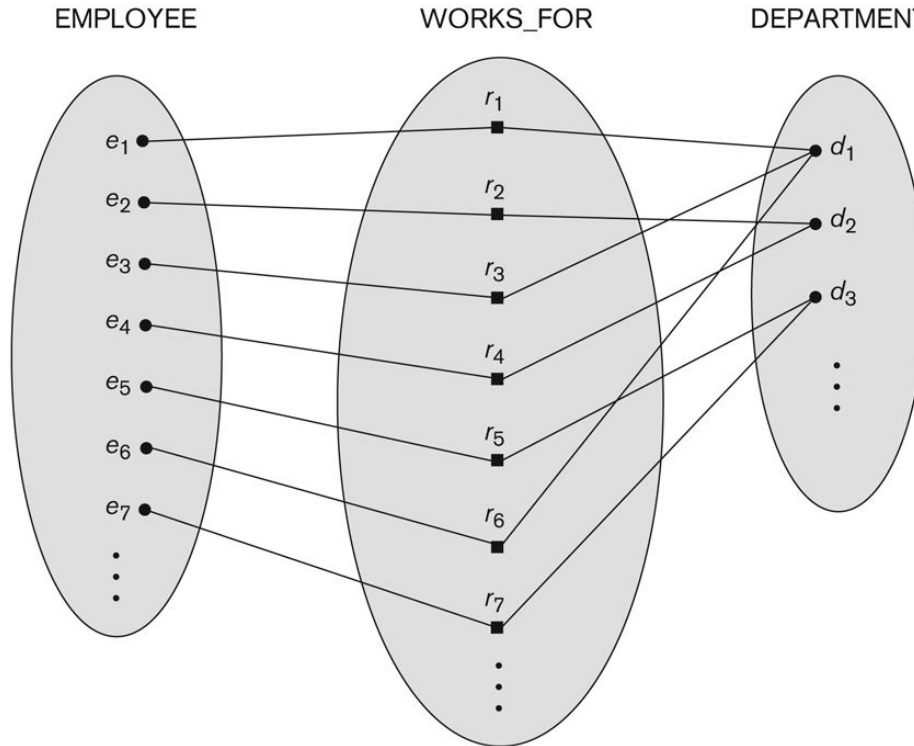


Figure 3.9

Some instances in the WORKS_FOR relationship set, which represents a relationship type WORKS_FOR between EMPLOYEE and DEPARTMENT.



Relationship type vs. Relationship set

- Relationship Type:
 - Is the schema description of a relationship
 - Identifies the relationship name and the participating entity types
 - Also identifies certain relationship constraints
- Relationship Set:
 - The current set of relationship instances represented in the database
 - The current *state* of a relationship type



Relationship type vs. Relationship set

- In ER diagrams, we represent the *relationship type* as follows:
 - **Diamond-shaped box** is used to display a relationship type
 - Connected to the participating entity types via straight lines



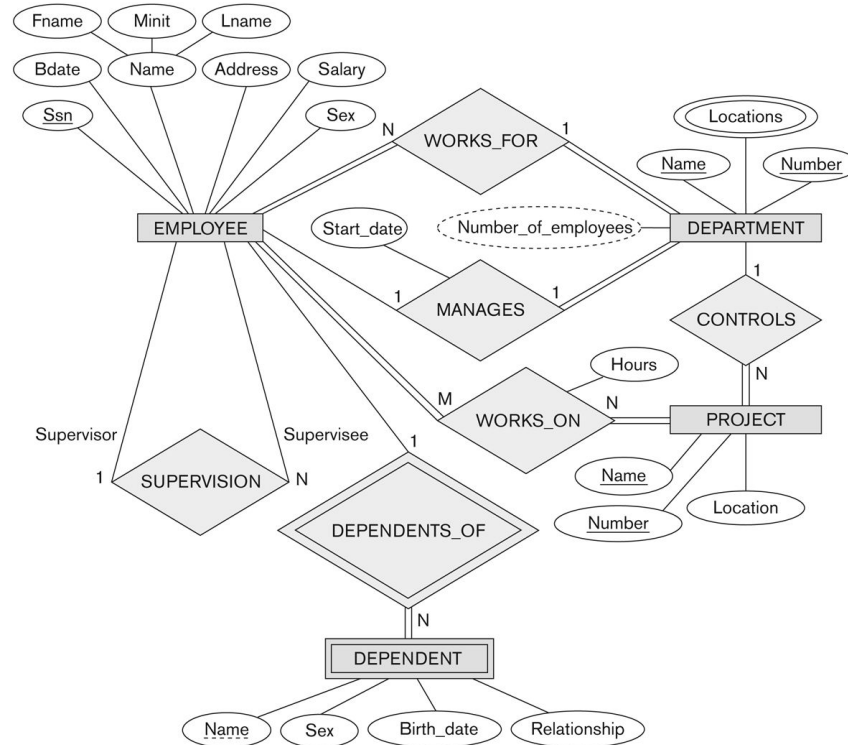
Refining the COMPANY database schema by introducing relationships

- By examining the requirements, six relationship types are identified
- All are *binary* relationships(degree 2)
- Listed below with their participating entity types:
 - WORKS_FOR (between EMPLOYEE, DEPARTMENT)
 - MANAGES (also between EMPLOYEE, DEPARTMENT)
 - CONTROLS (between DEPARTMENT, PROJECT)
 - WORKS_ON (between EMPLOYEE, PROJECT)
 - SUPERVISION (between EMPLOYEE (as subordinate), EMPLOYEE (as supervisor))
 - DEPENDENTS_OF (between EMPLOYEE, DEPENDENT)



ER DIAGRAM – Relationship Types are:

WORKS_FOR, MANAGES, WORKS_ON, CONTROLS, SUPERVISION, DEPENDENTS_OF



Discussion on Relationship Types

- In the refined design, some attributes from the initial entity types are refined into relationships:
 - Manager of DEPARTMENT -> MANAGES
 - Works_on of EMPLOYEE -> WORKS_ON
 - Department of EMPLOYEE -> WORKS_FOR
 - etc
- In general, more than one relationship type can exist between the same participating entity types
 - MANAGES and WORKS_FOR are distinct relationship types between EMPLOYEE and DEPARTMENT
 - Different meanings and different relationship instances.



Constraints on Relationships

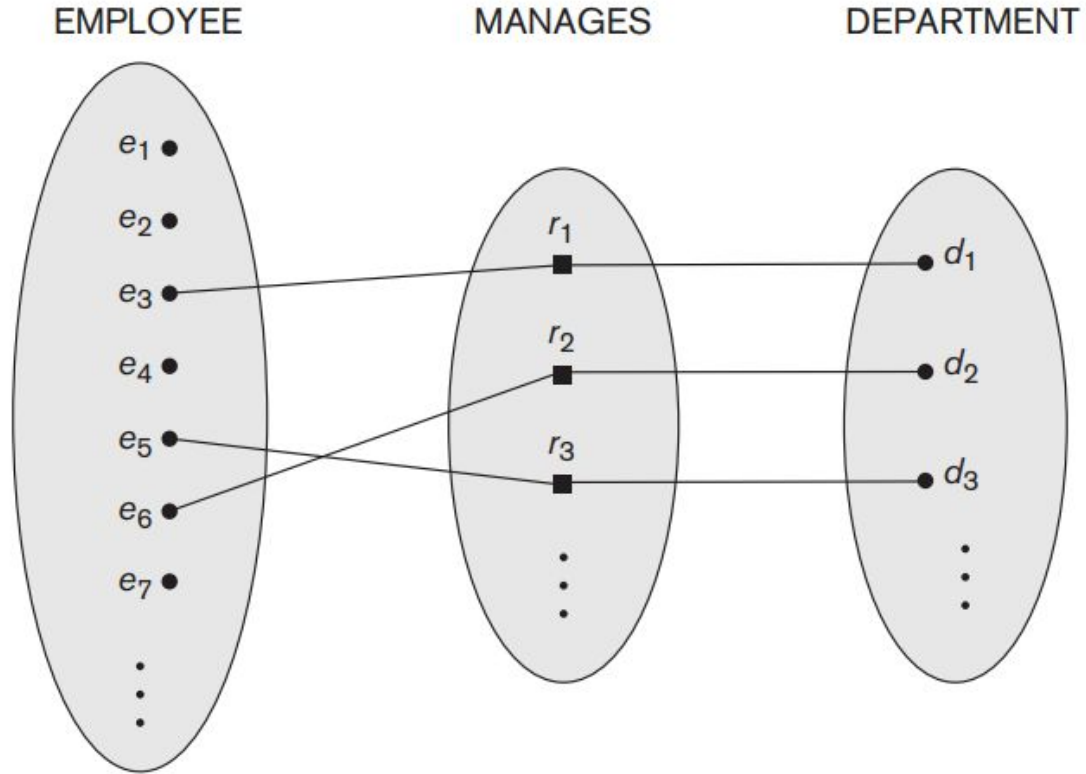
- Constraints on Relationship Types
 - (Also known as ratio constraints)
 - **Cardinality Ratio** (specifies *maximum* participation)
 - One-to-one (1:1)
 - One-to-many (1:N) or Many-to-one (N:1)
 - Many-to-many (M:N)
 - **Existence Dependency Constraint** (specifies *minimum* participation) (also called participation constraint)
 - zero (optional participation, not existence-dependent)
 - one or more (mandatory participation, existence-dependent)



One-to-one (1:1) Relationship

Figure 3.12

A 1:1 relationship,
MANAGES.



Many-to-one (N:1) Relationship

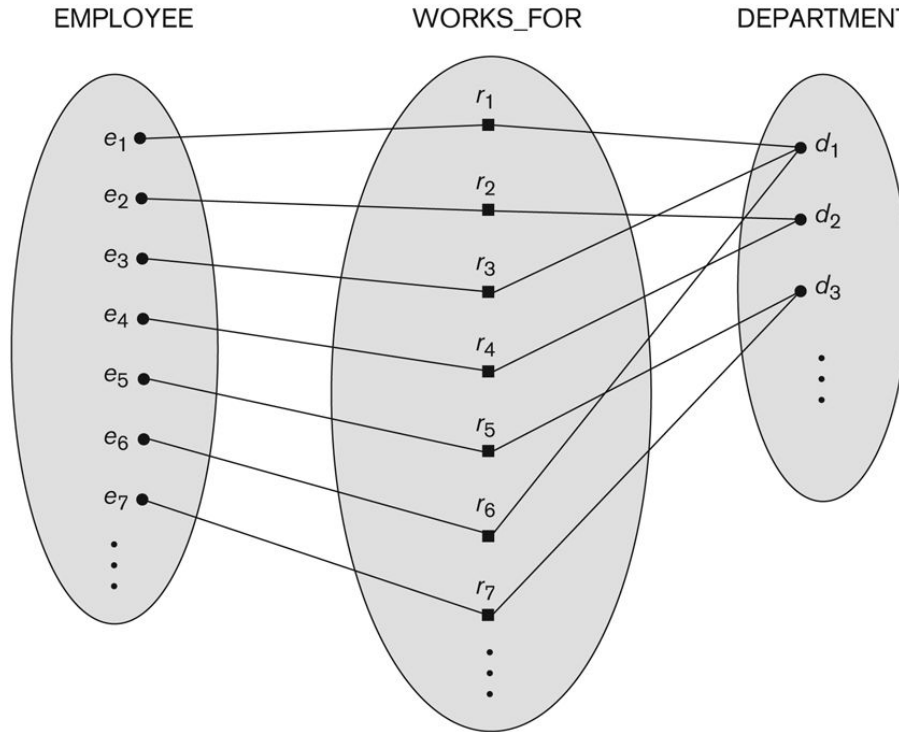


Figure 3.9

Some instances in the WORKS_FOR relationship set, which represents a relationship type WORKS_FOR between EMPLOYEE and DEPARTMENT.



Many-to-many (M:N) Relationship

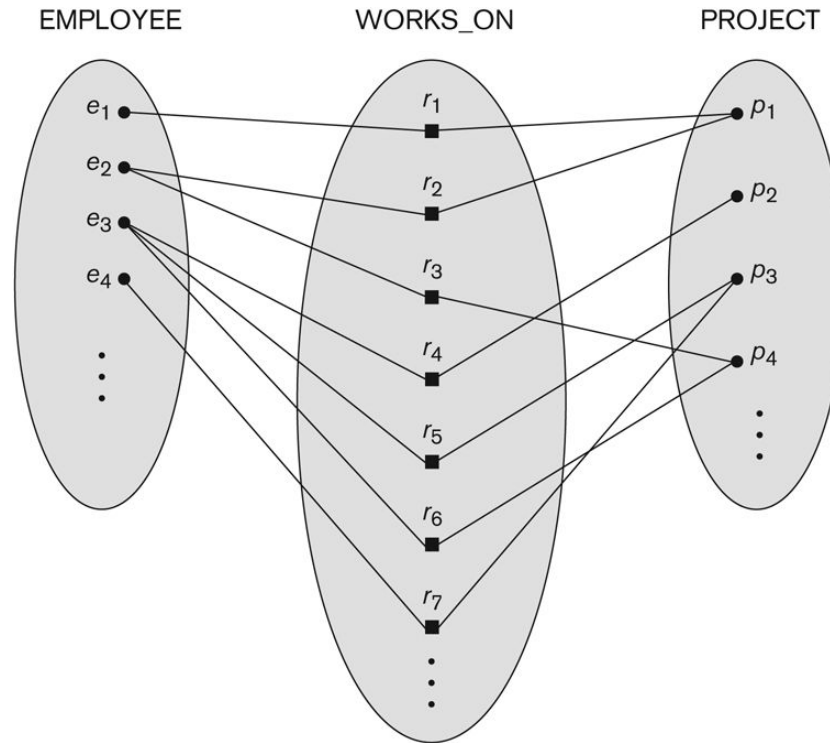


Figure 3.13
An M:N relationship,
WORKS_ON.



Notation for Constraints on Relationships

- Cardinality ratio (of a binary relationship): 1:1, 1:N, N:1, or M:N
 - Shown by placing appropriate numbers on the relationship edges.
 - Participation constraint (on each participating entity type): total (called existence dependency) or partial.
 - Total shown by double line, partial by single line.
- NOTE: These are easy to specify for Binary Relationship Types.



Cardinality ratio

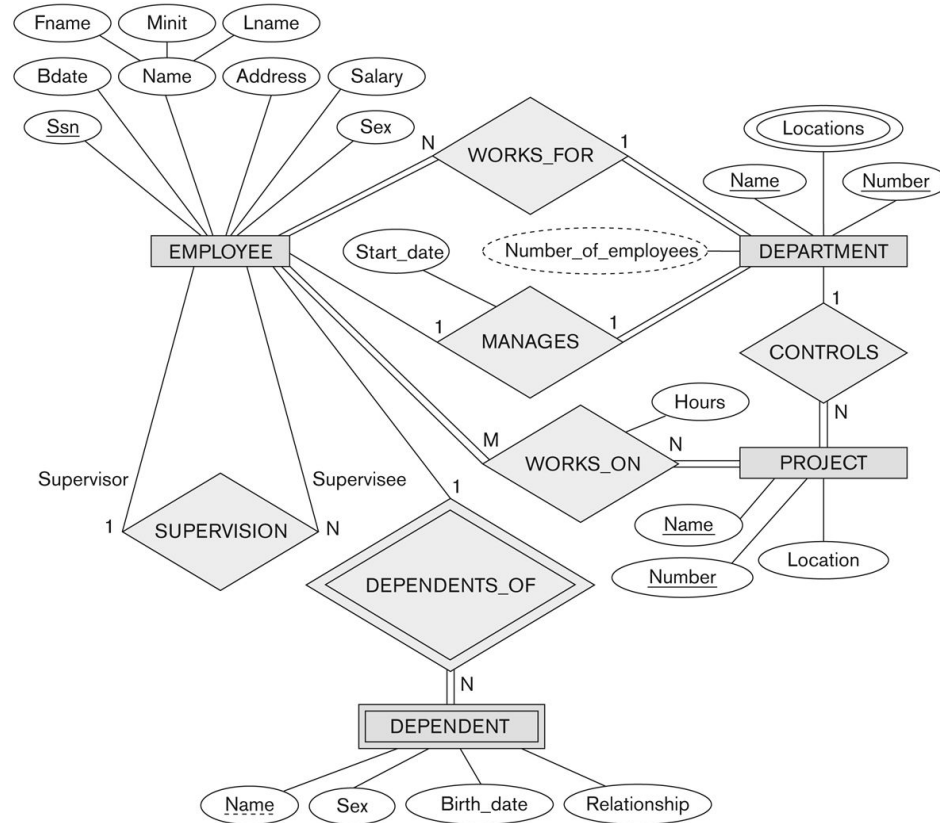


Figure 3.2

An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter.



Alternative (min, max) notation for relationship structural constraints:

- Specified on each participation of an entity type E in a relationship type R
- Specifies that each entity e in E participates in at least *min* and at most *max* relationship instances in R
- Default(no constraint): $\text{min}=0$, $\text{max}=\infty$ (signifying no limit)
- Must have $\text{min} \leq \text{max}$, $\text{min} \geq 0$, $\text{max} \geq 1$
- Derived from the knowledge of mini-world constraints



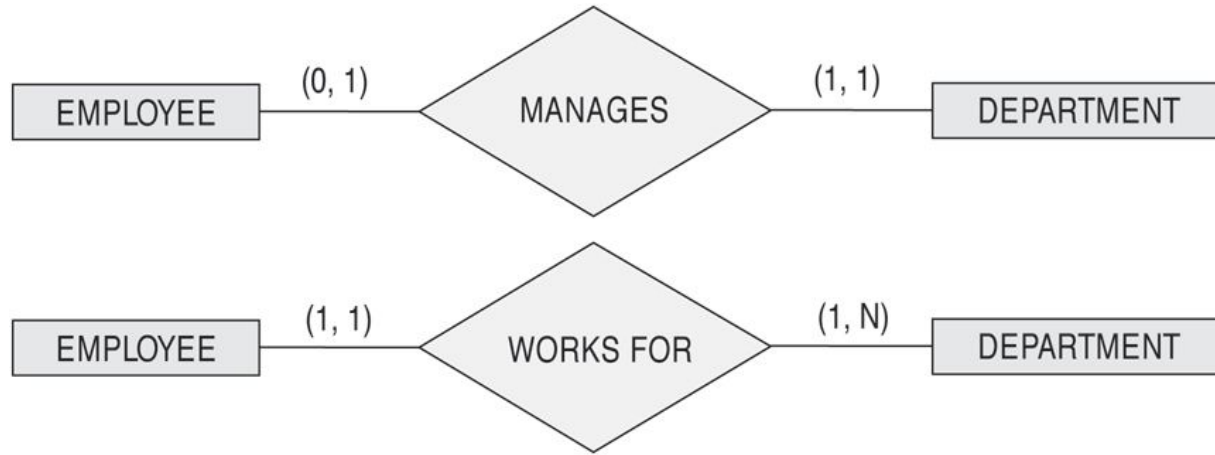
Alternative (min, max) notation for relationship structural constraints:

● Examples:

- A department has exactly one manager and an employee can manage at most one department.
 - Specify (0,1) for participation of EMPLOYEE in MANAGES
 - Specify (1,1) for participation of DEPARTMENT in MANAGES
- An employee can work for exactly one department but a department can have any number of employees.
 - Specify (1,1) for participation of EMPLOYEE in WORKS_FOR
 - Specify (0,n) for participation of DEPARTMENT in WORKS_FOR



The (min,max) notation for relationship constraints



Read the min,max numbers next to the entity type and looking **away from** the entity type



Recursive Relationship Type

- An relationship type whose with the same participating entity type in **distinct roles**
 - Example: the SUPERVISION relationship
- EMPLOYEE participates twice in two distinct roles:
 - supervisor (or boss) role
 - supervisee (or subordinate) role
- Each relationship instance relates two distinct EMPLOYEE entities:
 - One employee in *supervisor* role
 - One employee in *supervisee* role



Displaying a recursive relationship

- In a recursive relationship type.
 - Both participations are same entity type in different roles.
 - For example, SUPERVISION relationships between EMPLOYEE (in role of supervisor or boss) and (another) EMPLOYEE (in role of subordinate or worker).
- In following figure, first role participation labeled with 1 and second role participation labeled with 2.
- In ER diagram, need to display role names to distinguish participations.



A Recursive Relationship Supervision`

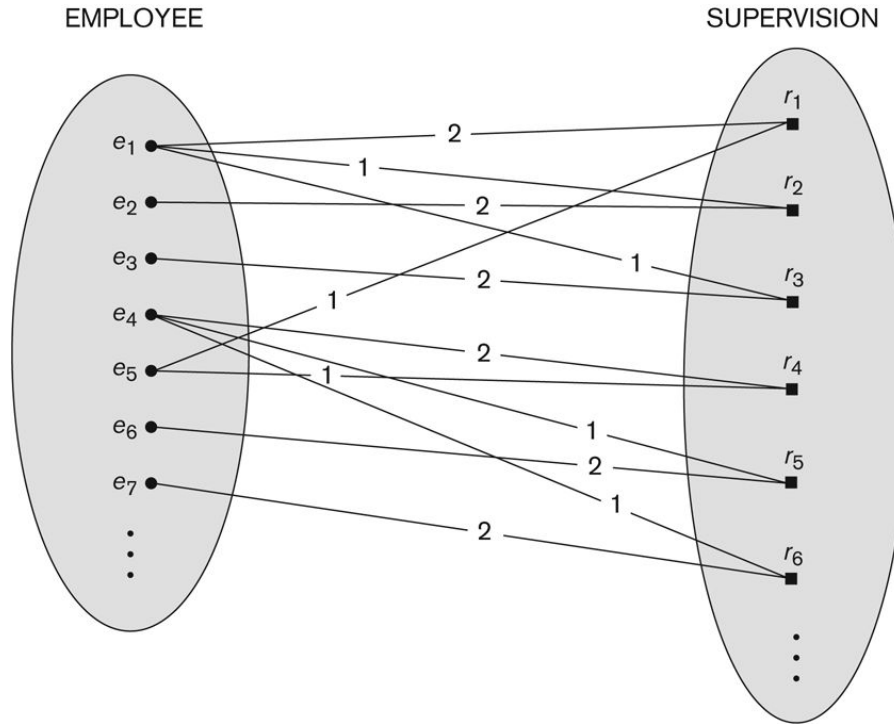


Figure 3.11

A recursive relationship SUPERVISION between EMPLOYEE in the *supervisor* role (1) and EMPLOYEE in the *subordinate* role (2).



Recursive Relationship Type is: SUPERVISION (participation role names are shown)

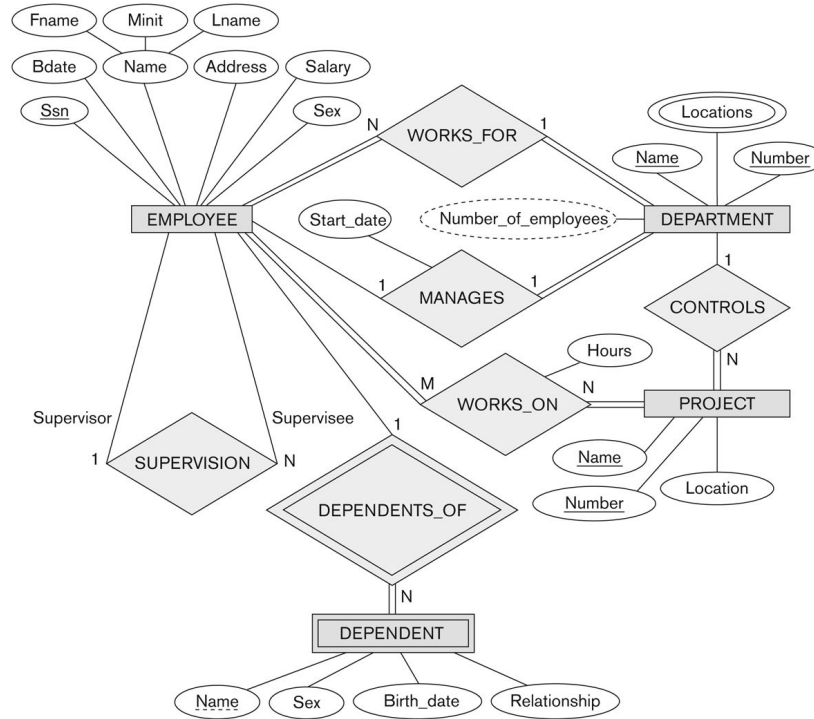


Figure 3.2

An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter.



Attributes of Relationship types

- A relationship type can have attributes:
 - For example, HoursPerWeek of WORKS_ON
 - Its value for each relationship instance describes the number of hours per week that an EMPLOYEE works on a PROJECT.
 - A value of HoursPerWeek depends on a particular (employee, project) combination
 - Most relationship attributes are used with M:N relationships
 - In 1:1 relationships, they can be transferred to the either one of the relationship
 - In 1:N relationships, they can be transferred to the entity type on the N-side of the relationship



Example Attribute of a Relationship Type: Hours of WORKS_ON

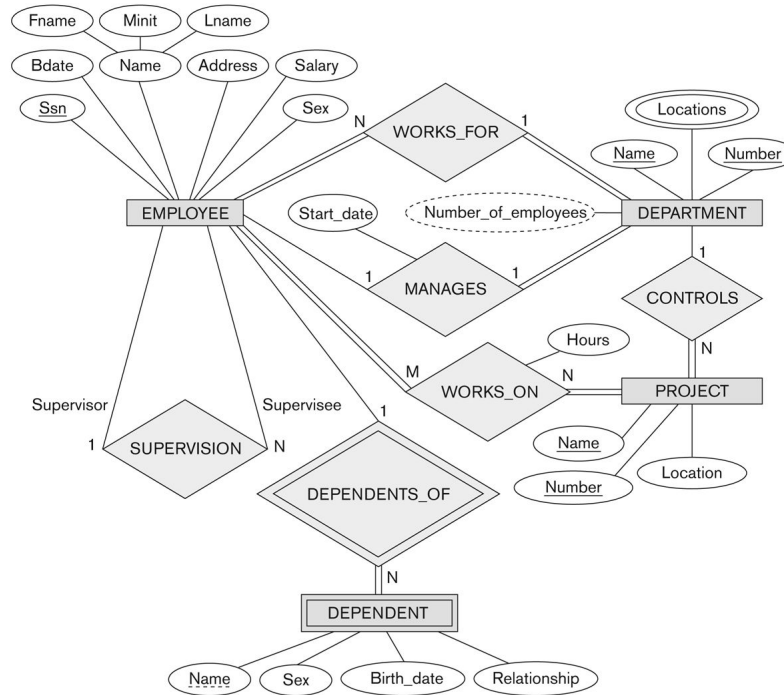
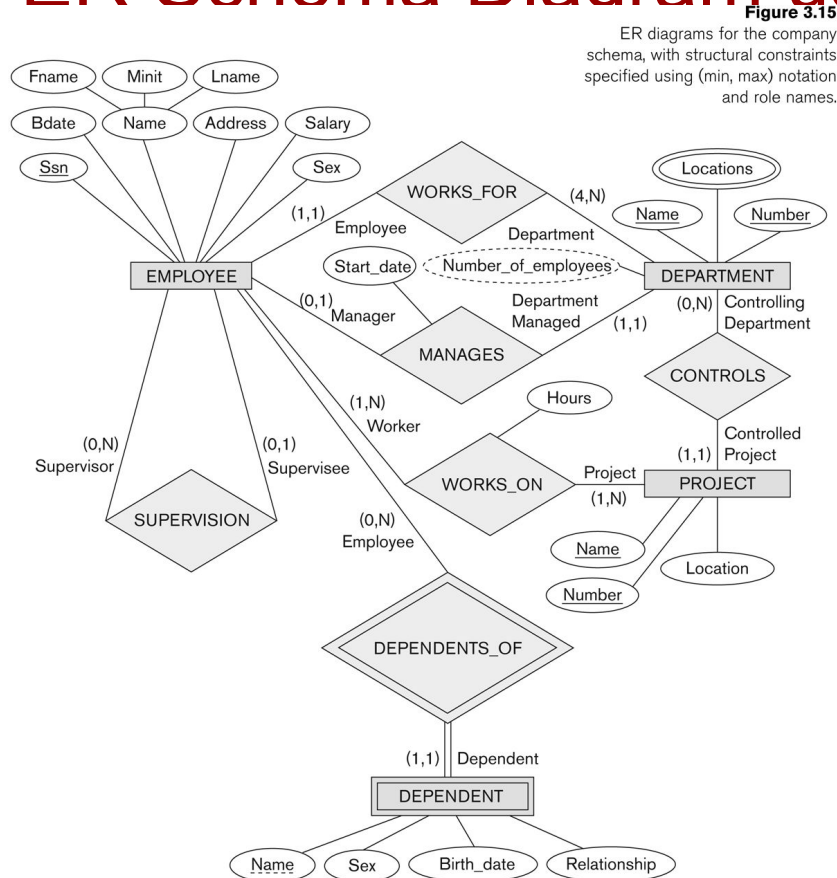


Figure 3.2

An ER schema diagram for the COMPANY database. The diagrammatic notation is introduced gradually throughout this chapter.








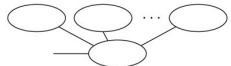

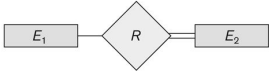




COMPANY ER Schema Diagram using (min, max) notation



Summary of notation for ER diagrams

Figure 3.14
Summary of the
notation for ER
diagrams.

Symbol	Meaning
	Entity
	Weak Entity
	Relationship
	Identifying Relationship
	Attribute
	Key Attribute
	Multivalued Attribute
	Composite Attribute
	Derived Attribute
	Total Participation of E_2 in R
	Cardinality Ratio 1: N for $E_1:E_2$ in R
	Structural Constraint (min, max) on Participation of E in R



Alternative diagrammatic notation

- ER diagrams is one popular example for displaying database schemas
- Many other notations exist in the literature and in various database design and modeling tools
- UML class diagrams is representative of another way of displaying ER concepts that is used in several commercial design tools



UML class diagrams

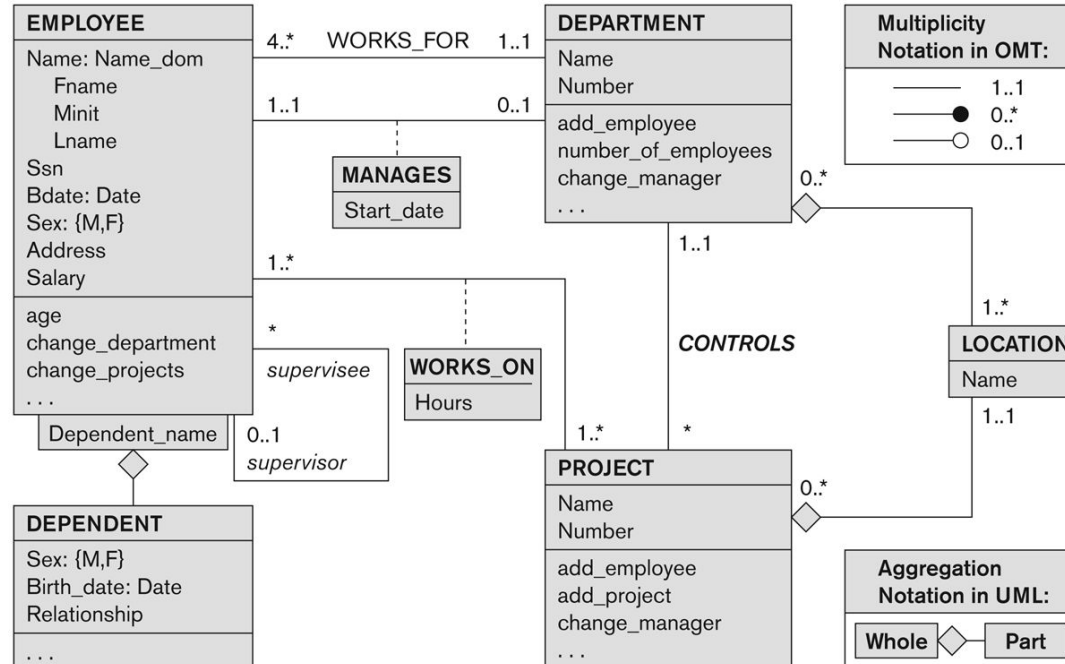
- Represent classes (similar to entity types) as large rounded boxes with three sections:
 - Top section includes entity type (class) name
 - Second section includes attributes
 - Third section includes class operations (operations are not in basic ER model)
- Relationships (called associations) represented as lines connecting the classes
 - Other UML terminology also differs from ER terminology
- Used in database design and object-oriented software design



UML class diagram for COMPANY database schema

Figure 3.16

The COMPANY conceptual schema in UML class diagram notation.



Relationships of Higher Degree

- Relationship types of degree 2 are called binary
- Relationship types of degree 3 are called ternary and of degree n are called n -ary
- In general, an n -ary relationship is not equivalent to n binary relationships
- Constraints are harder to specify for higher-degree relationships ($n > 2$) than for binary relationships



Discussion of n-ary relationships ($n > 2$)

- In general, 3 binary relationships can represent different information than a single ternary relationship (see Figure 3.17a and b on next slide)
- If needed, the binary and n-ary relationships can all be included in the schema design (see Figure 3.17a and b, where all relationships convey different meanings)
- In some cases, a ternary relationship can be represented as a weak entity if the data model allows a weak entity type to have multiple identifying relationships (and hence multiple owner entity types) (see Figure 3.17c)



Example of a ternary relationship

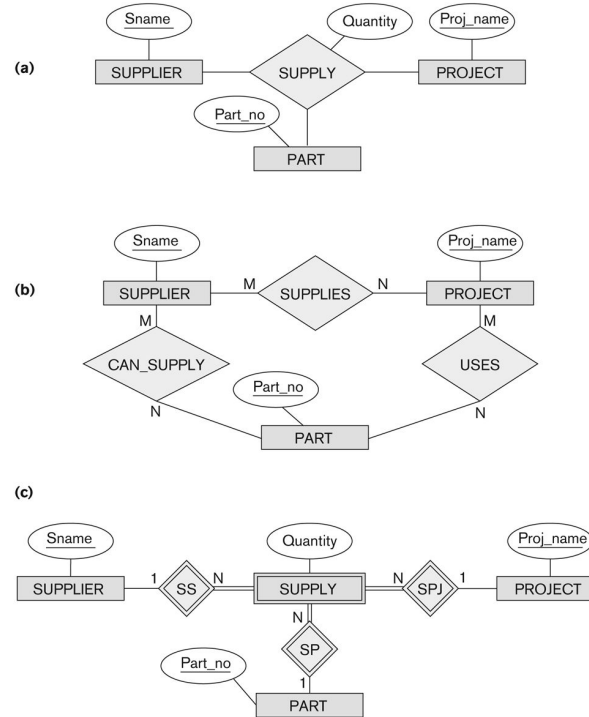


Figure 3.17

Ternary relationship types. (a) The SUPPLY relationship. (b) Three binary relationships not equivalent to SUPPLY. (c) SUPPLY represented as a weak entity type.

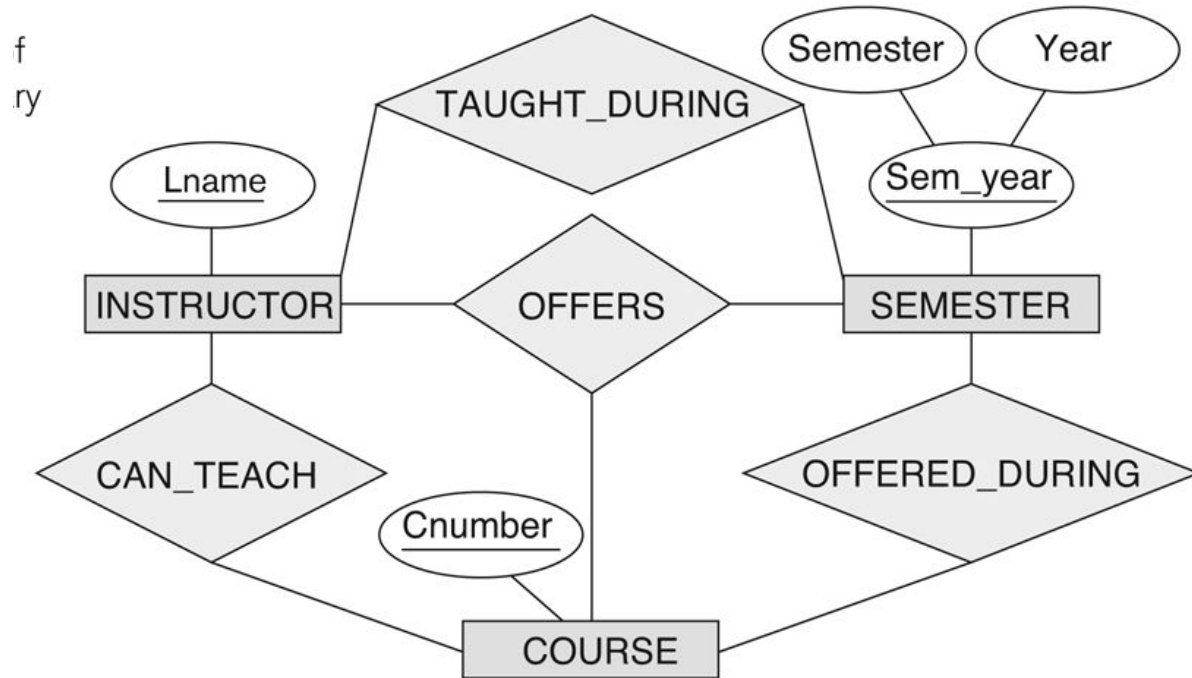


Discussion of n-ary relationships ($n > 2$)

- If a particular binary relationship can be derived from a higher-degree relationship at all times, then it is redundant
- For example, the TAUGHT_DURING binary relationship in Figure 3.18 (see next slide) can be derived from the ternary relationship OFFERS (based on the meaning of the relationships)



Another example of a ternary relationship



Displaying constraints on higher-degree relationships

- The (min, max) constraints can be displayed on the edges – however, they do not fully describe the constraints
- Displaying a 1, M, or N indicates additional constraints
 - An M or N indicates no constraint
 - A 1 indicates that an entity can participate in at most one relationship instance *that has a particular combination of the other participating entities*
- In general, both (min, max) and 1, M, or N are needed to describe fully the constraints



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 ER-based design



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 20A01	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++



Chapter Summary

- ER Model Concepts: Entities, attributes, relationships
- Constraints in the ER model
- Using ER in step-by-step conceptual schema design for the COMPANY database
- ER Diagrams - Notation
- Alternative Notations – UML class diagrams, others



Next Lesson

Extended Entity-Relationship (EER) Model

- The entity relationship model in its original form did not support the specialization and generalization abstractions
- Next chapter illustrates how the ER model can be extended with
 - Type-subtype and set-subset relationships
 - Specialization/Generalization Hierarchies
 - Notation to display them in EER diagrams



