Three Schema Model

- ANSI/SPARC introduced the three schema model in 1975
- It provides a framework describing the role and purpose of data modeling
Three Schema Model (cont.)

- **External schema or user view**
  - Representation of how users view the database

- **Conceptual schema**
  - A logical view of the database containing a description of all the data and relationships
  - Independent of any particular means of storing the data
  - One conceptual schema usually contains many different external schemas

- **Internal schema**
  - A representation of a conceptual schema as physically stored on a particular product
  - A conceptual schema can be represented by many different internal schemas
E-R Model

- **Entity-Relationship model** is a set of concepts and graphical symbols that can be used to create conceptual schemas

- **Four versions**
  - Original E-R model by Peter Chen (1976)
  - Extended E-R model: the most widely used model
  - Information Engineering (IE) by James Martin (1990)
  - IDEF1X national standard by the National Institute of Standards and Technology
  - Unified Modeling Language (UML) supporting object-oriented methodology
The Extended E-R Model

Figure 2.15 Example Entity-Relationship Diagram
Example: E-R Diagram

Figure 2.10  Showing Attributes in an Entity-Relationship Diagram
Entities

- Something that can be identified and the users want to track
  - *Entity class* is a collection of entities described by the entity format in that class
  - *Entity instance* is the representation of a particular entity
- There are usually many instances of an entity in an entity class
**Example: Entity**

**Figure 2.5** CUSTOMER: An Example of an Entity

CUSTOMER
entity contains:
- CustNumber
- CustName
- Address
- City
- State
- Zip
- ContactName
- PhoneNumber

Two instances of CUSTOMER:

12345
Ajax Manufacturing
123 Elm St
Memphis
TN
32455
P. Schwartz
223-5567

67890
Jefferson Dance Club
345-10th Avenue
Boston
MA
01234
Frita Bellingsley
210-8896
Attributes

- Description of the entity’s characteristics
- All instances of a given entity class have the same attributes
  - Composite attribute: attribute consisting of the group of attributes
  - Multi-value attributes: attribute with more than one possible value
Identifiers

- Identifiers are attributes that name, or identify, entity instances
- The identifier of an entity instance consists of one or more of the entity’s attributes
- An identifier may be either unique or non-unique
  - Unique identifier: the value identifies one and only one entity instance
  - Non-unique identifier: the value identifies a set of instances
- Composite identifiers: Identifiers that consist of two or more attributes
Relationships

- Entities can be associated with one another in relationships
  - Relationship classes: associations among entity classes
  - Relationship instances: associations among entity instances
- Relationships can have attributes
- A relationship class can involve many entity classes
- Degree of the relationship is the number of entity classes in the relationship
Example: Degree of the relationship

Relationships of degree 2 are very common and are often referred to by the term binary relationships
Binary Relationships

- 1:1
- 1:N
- N:M

Figure 2.7 Three Types of Binary Relationships
(a) 1:1 Binary Relationship; (b) 1:N Binary Relationship
and (c) N:M Binary Relationship
Recursive Relationship

- Recursive relationships are relationships among entities of a single class.

Figure 2.9 Recursive Relationship (a) E-R Diagram and (b) Sample Data

Diagram showing recursive relationships among entities.
Cardinality

- **Maximum cardinality** indicates the maximum number of entities that can be involved in a relationship.
- **Minimum cardinality** indicate that there may or may not be an entity in a relationship.

**Figure 2.8** Relationship with Minimum Cardinality Shown

![Diagram showing the relationship between DORMITORY and STUDENT with minimum cardinality indicated](image)
Weak Entities

- **Weak entities** are those that must logically depend on another entity.
- Weak entities cannot exist in the database unless another type of entity (strong entity) also exists in the database.
  - ID-dependent entity: the identifier of one entity includes the identifier of another entity.
Example: Weak Entities

Figure 2.11  Weak Entities (a) Weak, but Not ID-Dependent and (b) ID-Dependent

(a)
- **EMPLOYEE** 1:N **DEPENDENT**
  - Identifier: EmployeeNumber
  - Identifier: Social Security Number

(b)
- **BUILDING** 1:N **APARTMENT**
  - Identifier: BuildingName
  - Identifier: {BuildingName, ApartmentNumber}
Example: Weak Entities

Figure 2.12 Examples of Required Entities

(a) ORDER N:1 SALESPERSON

(b) PATIENT 1:N PRESCRIPTION

(c) ASSIGNMENT N:1 PROJECT

Identifier:
{ProjectName, TaskName}

Identifier:
ProjectName
Subtype Entities

- **Subtype entity** is an entity that represents a special case of another entity, called *supertype*
- Sometimes called an **IS-A relationship**
- Entities with an IS-A relationship should have the same identifier
Example: Subtype Entities

Figure 2.14a  Subtype Entities — CLIENT Without Subtype Entities

CLIENT Contains

ClientNumber
ClientName
AmountDue
Address
SocialSecurityNumber
ManagingPartnerName
TaxIdentificationNumber
ContactPerson
Phone

(a)
Example: Subtype Entities

**Figure 2.14b** Subtype Entities — CLIENT with Subtype Entities

- CLIENT
  - INDIVIDUAL-CLIENT
  - PARTNERSHIP-CLIENT
  - CORPORATE-CLIENT

**CLIENT Contains**
- ClientNumber
- ClientName
- AmountDue

**INDIVIDUAL-CLIENT Contains**
- Address
- SocialSecurityNumber

**PARTNERSHIP-CLIENT Contains**
- ManagingPartnerName
- Address
- TaxIdentificationNumber

**CORPORATE-CLIENT Contains**
- ContactPerson
- Phone
- TaxIdentificationNumber

(b)
Example: Subtype Entities

Figure 2.14c Subtype Entities — Non-Exclusive Subtypes with Optional Supertype

- CLIENT
- CLIENT USING Windows
- CLIENT USING Unix
- CLIENT USING Linux

(c)
IDEF1X Standard

- **IDEF1X** (Integrated Definition 1, Extended) was announced as a national standard in 1993.
- It defines entities, relationships, and attributes in more specific meanings.
- It changed some of the E-R graphical symbols.
- It includes definition of domains, a component not present in the extended E-R model.
- **Four Relationship Types**
  - Non-Identifying Connection Relationships
  - Identifying Connection Relationships
  - Non-Specific Relationships
  - Categorization Relationships
- **Products supporting IDEF1X**: ERWin, Visio, Design/2000
Example: IDEF1X

Figure 2.17a Levels of Detail in IDEF1X Models — Entities Only
Example: IDEF1X

Figure 2.17b  Levels of Detail in IDEF1X Models — Entities and Primary Keys
Example: IDEF1X

Figure 2.17c Levels of Detail in IDEF1X Models — Entities and Attributes
Non-Identifying Connection Relationships

- Represent relationship with a dashed line from a parent to a child entity
- Default cardinality is 1:N with a mandatory parent and an optional child
  - 1 indicates exactly one child is required
  - Z indicates zero or one children
  - P indicates one or more child entities are required
  - ◊ indicates the parent is optional
Non-Identifying Connection Relationships

Figure 2.19 Non-Identifying Connection Relationships
Identifying Connection Relationships

- Same as ID-dependent relationships in the extended E-R model
- Parent’s identifier is always part of the child’s identifier
- Relationship are indicated with solid lines, child entities are shown with rounded corners (ID-dependent entities only)
Identifying Connection Relationships

**Figure 2.20** Identifying Connection Relationship

<table>
<thead>
<tr>
<th>BUILDING</th>
<th>OFFICE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BuildingNumber</td>
<td>OfficeNumber</td>
</tr>
<tr>
<td>ReceptionPhone</td>
<td>BuildingNumber (FK)</td>
</tr>
<tr>
<td>NumberFloors</td>
<td>NetPortNumber</td>
</tr>
<tr>
<td>ParkingLot</td>
<td>PhonePortNumber</td>
</tr>
<tr>
<td></td>
<td>MaxCapacity</td>
</tr>
</tbody>
</table>
Non-Specific Relationships

- Simply a many-to-many relationship
- Relationships are shown with a filled-in circle on each end of the solid relationship line
- Cannot set minimum cardinalities of a non-specific relationship

![Diagram of Non-Specific Relationships](a)
Non-Specific Relationships

Figure 2.21b Non-Specific Relationships — Model Showing Missing Entity

EMPLOYEE
- SSN
- Name
- Phone
- JobCode

SKILL
- SkillName
- Description

SKILL_LEVEL
- Proficiency
- Description
- LastExamDate
- LastExamScore

Note: Identifier of SKILL_LEVEL is (SSN, SkillName)
Categorization Relationships

- A relationship between a generic entity and another entity called a **category entity**
- Called specialization of generalization/subtype relationships (IS-A relationships) in the extended E-R model
- Within category clusters, category entities are mutually exclusive
- Two types of category clusters:
  - **Complete**: every possible type of category for the cluster is shown (denoted by two horizontal lines with a gap in-between)
  - **Incomplete**: at least one category is missing (denoted by placing the category cluster circle on top of a single line, no gap between horizontal lines)
Example: Categorization Relationships

Figure 2.23 Incomplete and Complete Category Clusters
Example: IDEF1X Model With Relationship Names

Figure 2.24 IDEF1X Model Showing Relationship Names
Example: IDEF1X Model With Relationship Names

Figure 2.25 Using Names for Multiple Relationships between the Two Entities

CUSTOMER
- CustomerNumber
- Name
- Street
- City
- State
- Zip
- Phone
- Email

SALE
- InvoiceNumber
- Date
- Description
- NegotiatedPrice
- Tax
- Total
- Commission

SALESPERSON
- Name
- Phone
- BrokerName

Buys / Purchased By
Sells / Sold By
Sells / Sold By
Domains

- A **domain** is a named set of values that an attribute can have.
- It can be a specific list of values or a pre-defined data characteristic, e.g. character string of length less than 75.
- Domains reduce ambiguity in data modeling and are practically useful.
- Two types of domains:
  - **Base domain**: have a data type and possibly a value list or range definition.
  - **Type domain**: a subset of a base domain or a subset of another type domain.
Example: Domain Hierarchy

Figure 2.26 Example of Domain Hierarchy

DEPARTMENT_NAMES Domain

Accounting, Biology, Chemistry, Computer Science, Information Systems, Management, Physics, Zoology

SCIENCE Domain

Biology, Chemistry, Physics, Zoology

BUSINESS Domain

Accounting, Information Systems, Management

COMPUTING Domain

Computer Science, Information Systems

LIFE_SCIENCE Domain

Biology, Zoology

Base Domain

Type Domain
Example: IDEF1X

Figure 2.17c Levels of Detail in IDEF1X Models — Entities and Attributes
Alternative ER Notations

ER diagram for the COMPANY schema, with all role names included and with structural constraints on relationships specified using alternative notation (min, max).
Summary of ER Diagram

Notation for ER Schemas

- ENTITY TYPE (strong entity type)
- WEAK ENTITY TYPE -- no key!
- RELATIONSHIP TYPE
- IDENTIFYING RELATIONSHIP TYPE -- (between weak entity and identifying/owner entity type)
- ATTRIBUTE
- KEY ATTRIBUTE
- MULTIVALUED ATTRIBUTE -- double oval!
- COMPOSITE ATTRIBUTE
- DERIVED ATTRIBUTE

TOTAL PARTICIPATION OF E₁ IN R (i.e., the double line)
CARDINALITY RATIO 1:N FOR E₁:E₂ IN R
STRUCTURAL CONSTRAINT (min, max) ON PARTICIPATION OF E IN R
UML-style E-R Diagrams

- The Unified Modeling Language (UML) is a set of structures and techniques for modeling and designing object-oriented programs (OOP) and applications.
- The concept of UML entities, relationships, and attributes are very similar to those of the extended E-R model.
- Several OOP constructs are added:
  - `<Persistent>` indicates that the entity class exist in the database
  - UML allows entity class attributes
  - UML supports visibility of attributes and methods
  - UML entities specify constraints and methods in the third segment of the entity classes
- Currently, the object-oriented notation is of limited practical value.
Example: UML

Figure 2.27a  UML Representation of a 1:1 Relationship

<table>
<thead>
<tr>
<th>EMPLOYEE</th>
<th>AUTO-ASSIGNMENT</th>
<th>AUTO</th>
</tr>
</thead>
<tbody>
<tr>
<td>EmployeeID</td>
<td>0..1</td>
<td>LicenseNumber</td>
</tr>
<tr>
<td>Name</td>
<td>1..1</td>
<td>VIN</td>
</tr>
<tr>
<td>Title</td>
<td></td>
<td>Make</td>
</tr>
<tr>
<td>Phone</td>
<td></td>
<td>Model</td>
</tr>
<tr>
<td>SkillCode</td>
<td></td>
<td>Year</td>
</tr>
<tr>
<td>Constraints and methods named here</td>
<td></td>
<td>Constraints and methods named here</td>
</tr>
</tbody>
</table>

(a)
Example: UML

**Figure 2.27b** UML Representation of a 1:N Relationship

<table>
<thead>
<tr>
<th>DORMITORY</th>
<th>DORM-OCUPANT</th>
<th>STUDENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>0..1</td>
<td>StudentNumber</td>
</tr>
<tr>
<td>CampusAddress</td>
<td></td>
<td>StudentName</td>
</tr>
<tr>
<td>Capacity</td>
<td></td>
<td>Phone</td>
</tr>
<tr>
<td>HousePhone</td>
<td></td>
<td>Class</td>
</tr>
<tr>
<td>Constraints and methods</td>
<td>1..*</td>
<td>AssignedRoom</td>
</tr>
<tr>
<td>named here</td>
<td></td>
<td>Constraints and methods</td>
</tr>
<tr>
<td></td>
<td></td>
<td>named here</td>
</tr>
</tbody>
</table>
Example: UML

Figure 2.27c UML Representation of an N:M Relationship

<table>
<thead>
<tr>
<th>STUDENT</th>
<th>STUDENT-CLUB</th>
<th>CLUB</th>
</tr>
</thead>
<tbody>
<tr>
<td>StudentNumber</td>
<td>0..*</td>
<td>ClubNumber</td>
</tr>
<tr>
<td>StudentName</td>
<td>0..*</td>
<td>BudgetCode</td>
</tr>
<tr>
<td>Phone</td>
<td></td>
<td>Description</td>
</tr>
<tr>
<td>Class</td>
<td></td>
<td>President</td>
</tr>
<tr>
<td>AssignedRoom</td>
<td></td>
<td>PresidentPhone</td>
</tr>
<tr>
<td>Constraints and methods named here</td>
<td></td>
<td>Constraints and methods named here</td>
</tr>
</tbody>
</table>

(c)
Figure 2.28a  UML Representation of Weak Entities — Non-ID-Dependent Weak Entity

PATIENT
Name
Phone
InsuranceCompany
InsuranceNumber
Identifier: Name
Methods

PATIENT-PRESCRIPTION
<non-identifying>
1

PRESCRIPTION
Date
PrescriptionID
Doctor
Drug
Quantity
Identifier: PrescriptionID
Methods

(a)

Figure 2.28b  UML Representation of Weak Entities — ID-Dependent Weak Entity

PROJECT
ProjectName
StartDate
EndeDate
BudgetCode
Identifier: ProjectName
Methods

PROJECT-ASSIGN
<identifying>
1

ASSIGNMENT
TaskName
TaskStart
EstimatedHours
ActualHours
EmployeeName
Identifier: TaskName
Methods

(b)
UML: Subtypes

Figure 2.29 UML Representation of Subtypes

- **CLIENT**
  - ClientNumber
  - ClientName
  - AmountDue
  - Identifier: ClientNumber
  - Methods

- **ClientType**
  - **INDIVIDUAL**
    - Address
    - SocialSecurityNumber
    - Methods
  - **PARTNERSHIP**
    - ManagingPartnerName
    - Address
    - TaxIdentificationNumber
    - Methods
  - **CORPORATE**
    - ContactPerson
    - Phone
    - TaxIdentificationNumber
    - Methods