THE ENTITY–RELATIONSHIP MODEL AND EXTENSIONS

Data Modeling Using the ER Model and its extensions

• Main Phases of DB Design
• ER Diagrams- Notation
• 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
• Relationships of Higher Degree
• Extended Entity-Relationship (EER) Model

Notation is based on :
Main Phases of Database Design

- Fig 7.1
- Four Phases
  - Requirements Document and Analysis
  - Conceptual Design
  - Logical Design
  - Physical Design

ER-DIAGRAM NOTATION

- **Symbol**
  - ENTITY TYPE
  - WEAK ENTITY TYPE
  - RELATIONSHIP TYPE
  - IDENTIFYING RELATIONSHIP TYPE
  - ATTRIBUTE
  - KEY ATTRIBUTE
  - MULTIVALUED ATTRIBUTE
  - COMPOSITE ATTRIBUTE
  - DERIVED ATTRIBUTE

- **Meaning**
  - TOTAL PARTICIPATION OF E₂ IN R
  - CARDINALITY RATIO 1:N FOR E₁:E₂ IN R
  - STRUCTURAL CONSTRAINT (min, max) ON PARTICIPATION OF E IN R
Example COMPANY Database

- Requirements of the Company (oversimplified for illustrative purposes)
  - 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 name, number and is located at a single location.

Example COMPANY Database (Cont.)

- Store each EMPLOYEE’s social security number, address, salary, sex, and birth date. 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 employee.
ER Model Concepts: Entities and Attributes

- **Entities**: objects or *things* in the mini-world that are represented in the database.
  - E.g., the STUDENT John Smith, the Research DEPARTMENT, the COMPANY 3M, etc.

- **Attributes**: *properties* used to describe an entity
  - an STUDENT entity may have a Name, StudentID, Address, Class, Major

- A *specific entity* will have a value for each of its attributes; for example, a specific student entity may have
  Name='John Smith’, StudentId=‘22211333’, Address=‘731 Renner, Richardson, TX’, Class=‘Master’, Major=‘CS’
Types of Attributes

• **Simple**: Each entity has a *single atomic* value for the attribute
  – E.g., StudentID or Class

• **Composite**: The attribute may be composed of several components
  – Address (Apt#, House#, Street, City, State, ZipCode, Country) or Name(FirstName, MiddleName, LastName)
  – Composition may form a hierarchy where some components are themselves composite.

• **Multi-valued**: An entity may have *multiple values* for that attribute
  – E.g., Color of a CAR or PreviousDegrees of a STUDENT
  – Denoted as {Color} or {PreviousDegrees}.

• 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)}.

Entity Types and Key Attributes

• Entities with the same basic attributes are grouped into an **entity type**.
  – E.g., Mary Lee, John Smith, entities of type EMPLOYEE
  – CIA, FBI, entities of type Government Agency

• **Key** attribute: An attribute of an entity type for which each entity must have a *unique value* is called a **key attribute** of the entity type.
  – For example, SSN of EMPLOYEE
  – StudentID of STUDENT

• A key attribute may be composite.
  – VehicleTagNumber is a key of the CAR entity type with components (Number, State).

• An entity type may have more than one key. For example, the CAR entity type may have two keys:
  – VehicleIdentificationNumber (popularly called VIN) and
  – VehicleTagNumber (Number, State), also known as license_plate number.
ENTITY TYPE CAR WITH ATTRIBUTES

CAR
Registration(RegistrationNumber, State), VehicleID, Make, Model, Year, {Color}

car_1
((ABC 123, TEXAS), TK629, Ford Mustang, convertible, 1989, {red, black})
car_2
((ABC 123, NEW YORK), WP9872, Nissan Sentra, 2-door, 1992, {blue})
car_3
((VSY 720, TEXAS), TD729, Chrysler LeBaron, 4-door, 1993, {white, blue})

Entity Type Vs Entity Set

• Entity Type: aggregated entities
  – Share same structure
  – Described by entity type’s name and attributes
  – Intension

• Entity Set: The collection of all entities of a specific entity type
  – Extension
A relationship relates two or more distinct entities with a specific meaning:
- STUDENT John Smith taking the CS6360 COURSE
- EMPLOYEE Franklin Wang 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
- the MANAGES relationship type in which EMPLOYEES and DEPARTMENTs participate.
• The **degree** of a relationship type is the **number of participating** entity types.
  – Both MANAGES and WORKS_ON are binary relationships.
  – Recursive, ternary
• More than one relationship type can exist with the same participating entity types
  – MANAGES and WORKS_FOR are distinct relationships between EMPLOYEE and DEPARTMENT participate.

---

**Relationship Set**

Think of the “value” of a relationship set as a table.
• One column for each of the connected entity sets.
• One row for each list of entities, one from each set, that are connected by the relationship.

<table>
<thead>
<tr>
<th>Students</th>
<th>Courses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sally</td>
<td>CS180</td>
</tr>
<tr>
<td>Sally</td>
<td>CS111</td>
</tr>
<tr>
<td>Joe</td>
<td>CS180</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Students — Taking — Courses
Weak Entity Types

• An entity that does not have a key attribute
• A weak entity must participate in an identifying relationship type with an owner or identifying entity type
• Entities are identified by the combination of:
  – A partial key of the weak entity type
  – The particular entity they are related to in the identifying entity type

Example:
Suppose that a DEPENDENT entity is identified by the dependent’s first name and birthdate, and the specific EMPLOYEE that the dependent is related to. DEPENDENT is a weak entity type with EMPLOYEE as its identifying entity type via the identifying relationship type DEPENDENT_OF

ER Model and Data Abstraction

ABSTRACTION
• Classification
• Aggregation
• Identification
• Generalization

ER Model Concept
• Entity Type - a grouping of member entities
• Relationship Type - a grouping of member relationships
• Relationship Type is an aggregation of (over) its participating entity types
• Weak Entity Type
• ?????????
Constraints on Aggregation

- Cardinality Constraints on Relationship Types
  - (Also known as ratio constraints)
  - Maximum Cardinality
    - One-to-one
    - One-to-many
    - Many-to-one
    - Many-to-many
  - Minimum Cardinality (also called participation or existence dependency constraints)
    - zero (optional participation, not existence-dependent)
    - one or more (mandatory, existence-dependent)

One-to-many (1:N) or Many-to-one (N:1)
Structural Constraints – one way to express semantics of relationships

Structural constraints on relationships:

- **Cardinality ratio** (of a binary relationship): 1:1, 1:N, N:1, or M:N
  
  SHOWN BY PLACING APPROPRIATE NUMBER ON THE LINK.

- **Participation constraint** (on each participating entity type): total (called *existence dependency*) or partial.
  
  SHOWN BY DOUBLE LINING THE LINK

NOTE: These are easy to specify for Binary Relationship Types. Do not be misled by obscure notations to specify above constraints for higher order relationships
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): min=0, max=n
- Must have min≤max, min≥0, max ≥1
- Derived from the knowledge of mini-world 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 higher order relationship type constraints

What does it mean to put m:n:p on the three arms of the relationship?
It is essentially meaningless. The (min,max) notation “looking away” from the entity is the best to use.
Multi-way Relationships

Usually binary relationships (connecting two E.S.) suffice.

- However, there are some cases where three or more E.S. must be connected by one relationship.
- Example: relationship among students, courses, TA's (and graders).

Possibly, this E/R diagram is OK:

Example

- This binary relationship OK if each TA (or grader) is a TA of all students. Connection student-TA is \textit{only} via the course.
- But what if students were divided into sections, each headed by a TA?
  - Then, a student in CS 6360 would be related to only one of the TA's for CS6360. Which one?
- Need a 3-way relationship to tell.
Beers-Bars-Drinkers Example

Drinkers have favorite Beers

Students | Courses | TAs
---|---|---
Ann | CS6360 | Jan
Sue | CS6360 | Pat
Bob | CS6360 | Jan
Linda | CS6360 | Mike
... | ... | ...
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

TERNARY RELATIONSHIPS

Problem with constraints on higher order relationship types

What does it mean to put m:n:p on the three arms of the relationship? It is essentially meaningless.

The (min,max) notation for higher order relationship type constraints

A Teacher can offer min 1 and max 2 Offerings
A Course may have 1 to 3 Offerings
A Student may enroll in from 1 to 5 Offerings
Roles played by Entity Types in Relationship types

- In a recursive relationship two entities of the same entity type are related; for example, a SUPERVISION relationship type relates one EMPLOYEE (in the role of supervisee) to another EMPLOYEE (in the role of supervisor).

- Similarly, the same entity type may play different roles in different relationships. E.g., Employee plays the role of employee.

- 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.
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
Problems with Current Modeling Tools

- **DIAGRAMMING**
  - Poor conceptual meaningful notation.
  - To avoid the problem of layout algorithms and aesthetics of diagrams, they prefer boxes and lines and do nothing more than represent (primary-foreign key) relationships among resulting tables. (a few exceptions)

- **METHODOLGY**
  - lack of built-in methodology support.
  - poor tradeoff analysis or user-driven design preferences.
  - poor design verification and suggestions for improvement.

<table>
<thead>
<tr>
<th>COMPANY</th>
<th>TOOL</th>
<th>FUNCTIONALITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Embarcadero Technologies</td>
<td>ER Studio</td>
<td>Database Modeling in ER and IDEF1X</td>
</tr>
<tr>
<td></td>
<td>DB Artisan</td>
<td>Database administration and space and security management</td>
</tr>
<tr>
<td>Oracle</td>
<td>Developer 2000 and Designer 2000</td>
<td>Database modeling, application development</td>
</tr>
<tr>
<td>Popkin Software</td>
<td>System Architect 2001</td>
<td>Data modeling, object modeling, process modeling, structured analysis/design</td>
</tr>
<tr>
<td>Platinum Technology</td>
<td>Platinum Enterprise</td>
<td>Data, process, and business component modeling</td>
</tr>
<tr>
<td></td>
<td>Modeling Suite: Erwin, BPWin, Paradigm Plus</td>
<td></td>
</tr>
<tr>
<td>Persistence Inc.</td>
<td>Pwrtier</td>
<td>Mapping from O-O to relational model</td>
</tr>
<tr>
<td>Rational</td>
<td>Rational Rose</td>
<td>Modeling in UML and application generation in C++ and JAVA</td>
</tr>
<tr>
<td>Rogue Ware</td>
<td>RW Metro</td>
<td>Mapping from O-O to relational model</td>
</tr>
<tr>
<td>Resolution Ltd.</td>
<td>Xcase</td>
<td>Conceptual modeling up to code maintenance</td>
</tr>
<tr>
<td>Sybase</td>
<td>Enterprise Application Suite</td>
<td>Data modeling, business logic modeling</td>
</tr>
<tr>
<td>Visio</td>
<td>Visio Enterprise</td>
<td>Data modeling, design and reengineering Visual Basic and Visual C++</td>
</tr>
</tbody>
</table>
PROBLEM with ER notation

THE ENTITY RELATIONSHIP MODEL IN ITS ORIGINAL FORM DID NOT SUPPORT THE GENERALIZATION ABSTRACTION.
Extended Entity-Relationship (EER) Model

- Incorporates Set-subset relationships
- Incorporates Generalization Hierarchies
- LIMITATIONS OF THE ER MODEL:
  - No relationship may be defined between an entity type and a relationship type

NEXT SECTION OF THIS Presentation ILLUSTRATES HOW THE ER MODEL CAN BE EXTENDED WITH
- Set-subset relationships and Generalization Hierarchies
  and how we can impose further notation on them.

![EER Diagram](image)

Figure 8.1
EER diagram notation to represent subclasses and specialization.

Three specializations of EMPLOYEE:
- [SECRETARY, TECHNICIAN, ENGINEER]
- [MANAGER]
- [HOURLY_EMPLOYEE, SALARIED_EMPLOYEE]
8.3 Constraints and Characteristics of Specialization and Generalization Hierarchies

(a)

No_of_passengers
Max_speed
Vehicle_id
License_plate_no
CAR
Price

No_of_axles
Tonnage
Vehicle_id
License_plate_no
TRUCK

(b)

Vehicle_id
Price
License_plate_no
VEHICLE

Max_speed
No_of_passengers
CAR

No_of_axles
Tonnage
TRUCK

Figure 8.3
Generalization. (a) Two entity types, CAR and TRUCK. (b) Generalizing CAR and TRUCK into the superclass VEHICLE.

Figure 8.5
EER diagram notation for an overlapping (nondisjoint) specialization.

The notation of using single or double lines is similar to that for partial or total participation of an entity type in a relationship type, as described in Chapter 7.
Figure 8.6
A specialization lattice with shared subclass ENGINEERING_MANAGER.

Figure 8.7
A specialization lattice with multiple inheritance for a UNIVERSITY database.