Orange Coast College
Business Division
CS/CIS Department
Fall 2004
CIS 182
Introduction to Database Concepts

Instructor
Dr. Martha Malaty

Text & Original Presentations
Chapter 4

Entity Relationship (ER) Modeling

Database Systems:
Design, Implementation, and Management,
Sixth Edition, Rob and Coronel
In this chapter, you will learn:

• How relationships between entities are defined and refined, and how such relationships are incorporated into the database design process
• How ERD components affect database design and implementation
• How to interpret the modeling symbols for the four most popular ER modeling tools
• That real-world database design often requires that you reconcile conflicting goals
Basic Modeling Concepts

• Good judgment coupled with powerful design tools

• DB Design is both “Art & science”
  – Science:
    • Basic design principles
    • Leads to predictable outcome
  – Art:
    • Judgment call
The Entity Relationship (ER) Model

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

• Refers to the *entity set* and not to a single entity occurrence
• Corresponds to a table (not a row) in the relational environment
• In both the Chen and Crow’s Foot models, an entity is represented by a rectangle containing the entity’s name
• Entity name, a *noun*, is usually written in *capital* letters
Attributes

- Characteristics of entities
- In Chen model, attributes are represented by ovals and are connected to the entity rectangle with a line
- Each oval contains the name of the attribute it represents
- In the Crow’s Foot model, the attributes are simply written in the attribute box below the entity rectangle
Attributes

• Composite attribute
  – Can be subdivided into additional attributes
    • Address into street, city, zip
    • Phone into area code, number, & extension

• Simple attribute
  – Cannot be subdivided
    • Age, gender, marital status

• Single-valued attribute
  – Can have only a single value at any moment of time.
    • Person has one gender, one DOB, ...

• Multivalued attributes
  – Can have many values
    • Person may have several college degrees, several phone numbers...
    • Relational DBMS can’t implement multivalued attributes
The Attributes of the STUDENT Entity

FIGURE 4.1 THE ATTRIBUTES OF THE STUDENT ENTITY

Chen Model

Crow’s Foot Model

STUDENT

STU_NAME

STU_FNAM

STU_EMAIL

STU_PHONE

STU Initial

STU_LNAME

STU_FNAME

STU_E_MAIL

STU_PHONE

STU_LNAME

STU_FNAME

STU_INITIAL

STU_EMAIL

STU_PHONE
Domains

- Attributes have a *domain*:
  - The attribute’s set of possible values in a given type
  - Attributes may share a domain
Primary Keys / Unique ID

• Underlined in the ER diagram
  – In other books could be bold or italic
  – Example:
    ```sql
    TABLE STUDENT(STU_NUM, STU_FNAME, STU_LNAME,
                   STU_INITIAL, STU_E_MAIL, STU_PHONE)
    ```
• Key attributes are also underlined in frequently used table structure shorthand
• Ideally composed of only a single attribute
• Possible to use a composite key:
  – Primary key composed of more than one attribute
  – See TinyCollege Database
A Multivalued Attribute in an Entity

- **CAR_COLOR**
Resolving Multivalued Attribute Problems

• Although the conceptual model can handle multivalued attributes, you should not implement them in the relational DBMS.

• Two possible solutions
  – Within original entity, create several new attributes, one for each of the original multivalued attribute’s components
  – Create a new entity composed of original multivalued attribute’s components
Splitting the Multivalued Attribute into New Attributes

- Create several new attributes, one for each of the original multivalued attribute’s components
  - Car color converted to top color, trim color, ...etc.
- Can lead to major structural problems in the table
  - Waste of memory; If one entity fills all the attribute while others don’t

**Figure 4.4 Splitting the Multivalued Attribute into New Attributes**
Creating a New Entity for the Multivalued Attribute Components

- A better solution: separate the color to be in an entity by itself
  - The new entity is composed of original multivalued attribute’s components

- Example:
  - The designer can now allow different colors to different car parts

<table>
<thead>
<tr>
<th>SECTION</th>
<th>COLOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top</td>
<td>White</td>
</tr>
<tr>
<td>Body</td>
<td>Blue</td>
</tr>
<tr>
<td>Trim</td>
<td>Gold</td>
</tr>
<tr>
<td>Interior</td>
<td>Blue</td>
</tr>
</tbody>
</table>
A New Entity Set Composed of a Multivalued Attribute’s Components

**Figure 4.5** A New Entity Set Composed of a Multivalued Attribute’s Components

**Chen Model**

- **CAR_YEAR**
- **MOD_CODE**
- **CAR_VIN**
- **CAR**
- **CAR_VIN**
- **COL_COLOR**
- **COLOR**
- **COL_SECTION**

**Crow’s Foot Model**

- **CAR**
  - **PK**
  - **CAR_VIN**
  - **MOD_CODE**
  - **CAR_YEAR**
  - **CAR_COLOR**
- **COLOR**
  - **PK, FK1**
  - **CAR_VIN**
  - **COL_SECTION**
  - **COL_COLOR**

Derived Attributes

- Attribute whose value may be calculated (derived) from other attributes
- Need not be physically stored within the database
- Can be derived by using an algorithm
- Example:
  - Employee age can be derived from his DOB
Relationships

- Association between entities
- Participants:
  - Entities that participate in a relationship
- Relationships between entities always operate in both directions
- Relationship classification is difficult to establish if you only know one side
- Examples:
  - A DIVISION is managed by one EMPLOYEE
  - Does the employee manage one or more divisions?
    - 1:M
      - An EMPLOYEE may manage many DIVISIONS
    - 1:1
      - An EMPLOYEE may manage only one DIVISION
Connectivity and Cardinality

• Established by very concise statements known as **business rules**
• Connectivity
  – Used to describe the relationship classification
  – 1:1, 1:M, M:N
• Cardinality
  – Expresses the specific number of entity occurrences associated with one occurrence of the related entity
    • Chen model:
      – Number is written beside the entity using (x,y) format
        » **x represents the minimum of entity occurrences**
        » **y represents the maximum of entity occurrences**
      – The DBMS can’t handle cardinality on table level
        » **“Triggers” (software code) must be written to handle cardinality**
    • Crow’s Foot:
      – Doesn’t display numeric range
        » **Only minimum number for the cardinality information can be represented**
Connectivity & Cardinality in ERD

**FIGURE 4.7** CONNECTIVITY AND CARDINALITY IN AN ERD

Chen Model
Connectivities

1

M

PROFESSOR

teaches

CLASS

Cardinalities

(1,4)

(1,1)

Crow’s Foot Model
Connectivities

PROFESSOR

teaches

CLASS

Cardinalities
Relationship Strength

• Existence dependence (weak/dependent) entity
  – Entity’s existence depends on the existence of one or more other entities
  – Example:
    • Employee vs. dependants

• Existence independence entity
  – Entity can exist apart from one or more related entities
  – Often connected to other independent entities in a 1:M or M:N relationship
  – Example:
    • Course vs. class
Relationship Strength

• Weak (non-identifying) relationships
  – One entity relies on another entity for its existence and identification
  – PK of related entity doesn’t contain PK component of parent entity
• Example:
  – CLASS_CODE is PK for CLASS entity
  – CRS_CODE in CLASS is a FK and not part of the PK of CLASS entity
    • CLASS PK didn’t inherit the PK component from COURSE
Relationship Strength

- **Strong (Identifying) Relationships**
  - Related entities are existence-dependent
  - PK of related entity contains PK component of parent entity

- **Example:**
  - `CLASS_SECTION + CRS_CODE` compose the PK for `CLASS` entity
  - `CRS_CODE` in `CLASS` is a FK and part of the PK of `CLASS` entity
    - `CLASS` PK inherited the PK component from `COURSE`
Relationship Strength

• Example:
  – EMPLOYEE claims DEPENDENT
  – DEPENDENT can’t exist if the EMPLOYEE is deleted
  – EMOPLOYEE can exist if the DEPENDENT is deleted

• The order in which the tables are created, loaded, or deleted is very important

• General rule:
  – To avoid referential integrity problems, always load the “1” side first in an 1:M relationship
Weak (Non-Identifying) Relationship Between COURSE and CLASS

- Example:
  - COURSE table must be created before CLASS table, because a value for the FK in CLASS should exist.
Relationship Participation

• Optional:
  – One entity occurrence does not require a corresponding entity occurrence in a particular relationship
  – Shown by drawing a small circle on side of optional entity on ERD
  – Optional means that minimum cardinality is zero

• Mandatory:
  – One entity occurrence requires a corresponding entity occurrence in a particular relationship
  – Shown by drawing a bar on side of mandatory entity on ERD
  – If no optionality symbol is shown on ERD, it is mandatory
  – Mandatory means that minimum cardinality is 1
Strong (Identifying) Relationship Between COURSE and CLASS

**Figure 4.10** A Strong (Identifying) Relationship Between COURSE and CLASS

### COURSE Table

<table>
<thead>
<tr>
<th>CRS_CODE</th>
<th>DEPT_CODE</th>
<th>CRS_DESCRIPTION</th>
<th>CRS_CREDIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCT-211</td>
<td>ACCT</td>
<td>Accounting I</td>
<td>3</td>
</tr>
<tr>
<td>ACCT-212</td>
<td>ACCT</td>
<td>Accounting II</td>
<td>3</td>
</tr>
<tr>
<td>CIS-220</td>
<td>CIS</td>
<td>Intro. to Microcomputing</td>
<td>3</td>
</tr>
<tr>
<td>CIS-420</td>
<td>CIS</td>
<td>Database Design and Implementation</td>
<td>4</td>
</tr>
<tr>
<td>MATH-243</td>
<td>MATH</td>
<td>Mathematics for Managers</td>
<td>3</td>
</tr>
<tr>
<td>QM-261</td>
<td>CIS</td>
<td>Intro. to Statistics</td>
<td>3</td>
</tr>
<tr>
<td>QM-362</td>
<td>CIS</td>
<td>Statistical Applications</td>
<td>4</td>
</tr>
</tbody>
</table>

### CLASS Table

<table>
<thead>
<tr>
<th>CRS_CODE</th>
<th>CLASS_SECTION</th>
<th>CLASS_TIME</th>
<th>ROOM_CODE</th>
<th>PROF_NUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCT-211</td>
<td>1</td>
<td>MW 11:00-11:50 a.m.</td>
<td>BUS311</td>
<td>105</td>
</tr>
<tr>
<td>ACCT-211</td>
<td>2</td>
<td>MW 11:00-11:50 a.m.</td>
<td>BUS200</td>
<td>105</td>
</tr>
<tr>
<td>ACCT-211</td>
<td>3</td>
<td>MW 11:00-11:50 a.m.</td>
<td>BUS252</td>
<td>342</td>
</tr>
<tr>
<td>ACCT-212</td>
<td>1</td>
<td>MW 11:00-11:50 a.m.</td>
<td>BUS311</td>
<td>301</td>
</tr>
<tr>
<td>ACCT-212</td>
<td>2</td>
<td>MW 11:00-11:50 a.m.</td>
<td>BUS252</td>
<td>301</td>
</tr>
<tr>
<td>CIS-220</td>
<td>1</td>
<td>MW 11:00-11:50 a.m.</td>
<td>KLR209</td>
<td>228</td>
</tr>
<tr>
<td>CIS-220</td>
<td>2</td>
<td>MW 11:00-11:50 a.m.</td>
<td>KLR209</td>
<td>228</td>
</tr>
<tr>
<td>CIS-220</td>
<td>3</td>
<td>MW 11:00-11:50 a.m.</td>
<td>KLR209</td>
<td>228</td>
</tr>
<tr>
<td>CIS-420</td>
<td>1</td>
<td>MW 11:00-11:50 a.m.</td>
<td>KLR209</td>
<td>228</td>
</tr>
<tr>
<td>MATH-243</td>
<td>1</td>
<td>MW 11:00-11:50 a.m.</td>
<td>DRE155</td>
<td>252</td>
</tr>
</tbody>
</table>
Optional CLASS Entity in the Relationship
PROFESSOR teaches CLASS

- CLASS is optional, PROFESSOR is mandatory
COURSE and CLASS in a Mandatory Relationship

- COURSE is mandatory, CLASS is optional
- Two scenarios depending on business rules:
  - CLASS optional (Fig. 4.12):
    - It is possible to create COURSE before CLASS
  - CLASS mandatory (Fig. 4.13):
    - CLASS must be created as the course is created

**FIGURE 4.13 COURSE AND CLASS IN A MANDATORY RELATIONSHIP**
Relationship Strength & Weak Entities

• Weak entity meets the following conditions
  – Existence-dependent; cannot exist without entity with which it has a relationship
  – Has primary key that is partially or totally derived from the parent entity in the relationship

• Representation:
  – In Chen’s model: double-walled rectangle
  – In Crow’s Foot model: short line segments on the four corners of the rectangle

• Database designer usually determines whether an entity can be described as weak based on the business rules
Weak Entity in ERD

**Figure 4.14 A Weak Entity in an ERD**

Chen Model

```
1  has  M
EMPLOYEE  has  DEPENDENT
(0,N)   (1,1)
```

EMP_NUM
EMP_LNAME
EMP_FNAME
EMP_INITIAL
EMP_DOB
EMP_HIREDATE

EMP_NUM
DEP_NUM
DEP_FNAME
DEP_DOB

Crow’s Foot Model

Relationship Degree

• Indicates number of associated entities or participants
• Unary (Recursive) relationship
  – Association is maintained within a single entity
• Binary relationship
  – Two entities are associated
• Ternary relationship
  – Three entities are associated
Three Types of Relationships

**Figure 4.16 Three Types of Relationships**

### Chen Models
- **Unary Relationship**
  - COURSE (M:1) requires PROFESSOR (1:M) teaches CLASS (M:N)
- **Binary Relationship**
  - CONTRIBUTOR (M) contributes to CFR (1:M) receives from RECIPIENT (N)
  - FUND (P)

### Crow’s Foot Models
- **Unary Relationship**
  - COURSE (requires) is prerequisite to COURSE
- **Binary Relationship**
  - PROFESSOR teaches CLASS
- **Ternary Relationship**
  - CONTRIBUTOR contributes to CFR receives from RECIPIENT
  - FUND is distributed in
FIGURE 4.17  THE IMPLEMENTATION OF A TERNARY RELATIONSHIP

Database name: Ch04_MedCo

Table name: CONTRIBUTOR

<table>
<thead>
<tr>
<th>CONTRIB_ID</th>
<th>CONTRIB_LNAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>Brown</td>
</tr>
<tr>
<td>C2</td>
<td>Iglesias</td>
</tr>
<tr>
<td>C3</td>
<td>Smith</td>
</tr>
</tbody>
</table>

Table name: FUND

<table>
<thead>
<tr>
<th>FUND_ID</th>
<th>FUND_NAME</th>
<th>CONTRIB_ID</th>
<th>FUND_AMOUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>f1</td>
<td>Heart</td>
<td>C1</td>
<td>$50,000.00</td>
</tr>
<tr>
<td>F1</td>
<td>Heart</td>
<td>C2</td>
<td>$10,000.00</td>
</tr>
<tr>
<td>F2</td>
<td>Cancer</td>
<td>C1</td>
<td>$10,000.00</td>
</tr>
<tr>
<td>F2</td>
<td>Cancer</td>
<td>C2</td>
<td>$5,000.00</td>
</tr>
<tr>
<td>F2</td>
<td>Cancer</td>
<td>C3</td>
<td>$10,000.00</td>
</tr>
</tbody>
</table>

Table name: RECIPIENT

<table>
<thead>
<tr>
<th>REC_ID</th>
<th>REC_TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>Rogers</td>
</tr>
<tr>
<td>R2</td>
<td>Chen</td>
</tr>
<tr>
<td>R3</td>
<td>Oshanski</td>
</tr>
</tbody>
</table>

Table name: CFR

<table>
<thead>
<tr>
<th>FUND_ID</th>
<th>CON_ID</th>
<th>REC_ID</th>
<th>CFR_AMOUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>f1</td>
<td>c1</td>
<td>r2</td>
<td>$30,000.00</td>
</tr>
<tr>
<td>F1</td>
<td>C1</td>
<td>R3</td>
<td>$20,000.00</td>
</tr>
<tr>
<td>F1</td>
<td>C2</td>
<td>R2</td>
<td>$10,000.00</td>
</tr>
<tr>
<td>F2</td>
<td>C1</td>
<td>R1</td>
<td>$10,000.00</td>
</tr>
<tr>
<td>F2</td>
<td>C2</td>
<td>R1</td>
<td>$5,000.00</td>
</tr>
</tbody>
</table>

Implementation of Ternary Relationship
Recursive Relationships

• Relationship can exist between occurrences of the same entity set
• Naturally found within a unary relationship
ER Representation of Recursive Relationships

**FIGURE 4.18 An ER Representation of Recursive Relationships**

There is no need for two lines for all the relationships here!
1:1 Recursive Relationship
“EMPLOYEE is Married to EMPLOYEE”
Implementation of M:N Recursive “PART Contains PART” Relationship

**Figure 4.21 Implementation of the M:N Recursive “PART Contains PART” Relationship**

<table>
<thead>
<tr>
<th>Table name: COMPONENT</th>
<th>Database name: Ch04_PartCo</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMP_CODE</td>
<td>PART_CODE</td>
</tr>
<tr>
<td>C-130</td>
<td>AA21-6</td>
</tr>
<tr>
<td>C-130</td>
<td>AB-121</td>
</tr>
<tr>
<td>C-130</td>
<td>E129</td>
</tr>
<tr>
<td>C-131A2</td>
<td>E129</td>
</tr>
<tr>
<td>C-130</td>
<td>X10</td>
</tr>
<tr>
<td>C-131A2</td>
<td>X10</td>
</tr>
<tr>
<td>C-130</td>
<td>X34AW</td>
</tr>
<tr>
<td>C-131A2</td>
<td>X34AW</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table name: PART</th>
</tr>
</thead>
<tbody>
<tr>
<td>PART_CODE</td>
</tr>
<tr>
<td>AA21-6</td>
</tr>
<tr>
<td>AB-121</td>
</tr>
<tr>
<td>C-130</td>
</tr>
<tr>
<td>E129</td>
</tr>
<tr>
<td>X10</td>
</tr>
<tr>
<td>X34AW</td>
</tr>
</tbody>
</table>
Implementation of 1:M “EMPLOYEE Manages EMPLOYEE” Recursive Relationship

Figure 4.23 Implementation of the 1:M “EMPLOYEE Manages EMPLOYEE” Recursive Relationship

Table name: EMPLOYEE_V2

<table>
<thead>
<tr>
<th>EMP_CODE</th>
<th>EMP_LNAME</th>
<th>EMP_MANAGER</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>Waddell</td>
<td></td>
</tr>
<tr>
<td>102</td>
<td>Orincon</td>
<td></td>
</tr>
<tr>
<td>103</td>
<td>Jones</td>
<td></td>
</tr>
<tr>
<td>104</td>
<td>Roberts</td>
<td></td>
</tr>
<tr>
<td>105</td>
<td>Robertson</td>
<td></td>
</tr>
<tr>
<td>106</td>
<td>Betina</td>
<td></td>
</tr>
</tbody>
</table>

Database name: Ch04_PartCo
Composite / Bridge Entities

- Composed of the primary keys of each of the entities to be connected
- May also contain additional attributes that play no role in the connective process
- Used to ‘bridge’ between M:N relationships
- No null entries are possible for its key attributes
M:N Relationship Between STUDENT and CLASS

FIGURE 4.25 The M:N Relationship Between STUDENT and CLASS
Composite Entity in ERD

FIGURE 4.26 A COMPOSITE ENTITY IN AN ERD

Chen Model

Crow’s Foot Model

Converting **M:N** Relationship into Two **1:M** Relationships

**Figure 4.24** Converting the M:N Relationship into Two 1:M Relationships

<table>
<thead>
<tr>
<th>Table name: STUDENT</th>
<th>Database name: Ch04_CollegeTry</th>
</tr>
</thead>
<tbody>
<tr>
<td>STU_NUM</td>
<td>STUD_NAME</td>
</tr>
<tr>
<td>+</td>
<td>321452</td>
</tr>
<tr>
<td>+</td>
<td>324257</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table name: ENROLL</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLASS_CODE</td>
</tr>
<tr>
<td>+ 10014</td>
</tr>
<tr>
<td>10014</td>
</tr>
<tr>
<td>10018</td>
</tr>
<tr>
<td>10018</td>
</tr>
<tr>
<td>10021</td>
</tr>
<tr>
<td>10021</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table name: CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLASS_CODE</td>
</tr>
<tr>
<td>+ 10014</td>
</tr>
<tr>
<td>+ 10018</td>
</tr>
<tr>
<td>+ 10021</td>
</tr>
</tbody>
</table>
Entity Supertypes and Subtypes

• Generalization hierarchy
  – Depicts a relationship between a higher-level supertype entity and a lower-level subtype entity
  – Part of the Extended ER Model
    • Were added to ER model after the publication of Chen’s initial paper

• Supertype entity
  – Contains shared attributes

• Subtype entity
  – Contains unique attributes
Nulls Created by Unique Attributes

- If all employee characteristics were described by a single EMPLOYEE entity, we would either have lots of nulls (or dummy entries)

**FIGURE 4.27** Nulls Created by Unique Attributes

<table>
<thead>
<tr>
<th>EMP_NUM</th>
<th>EMP_LNAME</th>
<th>EMP_LICENSE</th>
<th>EMP_RATINGS</th>
<th>EMP_MED_TYPE</th>
<th>EMP_HIRE_DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Kolmycz</td>
<td>ATP</td>
<td>SELMEL1InstrCFII</td>
<td>1</td>
<td>15-Mar-88</td>
</tr>
<tr>
<td>101</td>
<td>Lewis</td>
<td>ATP</td>
<td>SELMEL1InstrCFII</td>
<td>2</td>
<td>25-Apr-89</td>
</tr>
<tr>
<td>102</td>
<td>Vandam</td>
<td>COM</td>
<td>SELMEL1InstrCFII</td>
<td>2</td>
<td>20-Dec-93</td>
</tr>
<tr>
<td>103</td>
<td>Jones</td>
<td>COM</td>
<td>SELMEL1InstrCFII</td>
<td>1</td>
<td>28-Aug-03</td>
</tr>
<tr>
<td>104</td>
<td>Lange</td>
<td>COM</td>
<td>SELMEL1Instr</td>
<td>1</td>
<td>20-Oct-97</td>
</tr>
<tr>
<td>105</td>
<td>Williams</td>
<td>COM</td>
<td>SELMEL1InstrCFII</td>
<td>2</td>
<td>08-Nov-97</td>
</tr>
<tr>
<td>106</td>
<td>Duzak</td>
<td>COM</td>
<td>SELMEL1InstrCFII</td>
<td>2</td>
<td>05-Jan-04</td>
</tr>
<tr>
<td>107</td>
<td>Diante</td>
<td>COM</td>
<td>SELMEL1InstrCFII</td>
<td>1</td>
<td>02-Jul-97</td>
</tr>
<tr>
<td>108</td>
<td>Wiesenthal</td>
<td>COM</td>
<td>SELMEL1InstrCFII</td>
<td>1</td>
<td>18-Nov-95</td>
</tr>
<tr>
<td>109</td>
<td>Travis</td>
<td>COM</td>
<td>SELMEL1InstrCFII</td>
<td>2</td>
<td>14-Apr-01</td>
</tr>
<tr>
<td>110</td>
<td>Genkezi</td>
<td>COM</td>
<td>SELMEL1InstrCFII</td>
<td>1</td>
<td>01-Dec-03</td>
</tr>
</tbody>
</table>
A Generalization Hierarchy

• A relationship between higher-level supertype entity with a lower-level subtype entity
• Subtypes inherit characteristics from the supertype
• Two categories:
  – Disjoined (Non-overlapping)
  – Overlapping

**FIGURE 4.28 A GENERALIZATION HIERARCHY**

![Diagram of a generalization hierarchy]

Supertype

Subtypes

- PILOT
- MECHANIC
- ACCOUNTANT
Disjoint (Non-Overlapping) Subtypes

- Subtypes that contain a subset of the supertype entity set
- Each entity instance (row) of the supertype can appear in only one of the disjoint subtypes
- Indicated with a ‘G’
- Example:
  - An employee who is a “Pilot” can only appear in the PILOT subtype, not in any other disjoint subtype
- Supertype and its subtype(s) maintain a 1:1 relationship
**EMPLOYEE/PILOT**  
Supertype/Subtype Relationship

---

**FIGURE 4.29** The EMPLOYEE/PILOT Supertype/Subtype Relationship

- **Database name:** Ch04_AirCo

**Table name: EMPLOYEE (the supertype)**

<table>
<thead>
<tr>
<th>EMP_NUM</th>
<th>EMP_LNAME</th>
<th>EMP_HIRE_DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Kolmycz</td>
<td>15-Mar-88</td>
</tr>
<tr>
<td>101</td>
<td>Levvis</td>
<td>25-Apr-89</td>
</tr>
<tr>
<td>102</td>
<td>Vandam</td>
<td>20-Dec-93</td>
</tr>
<tr>
<td>103</td>
<td>Jones</td>
<td>28-Aug-03</td>
</tr>
<tr>
<td>104</td>
<td>Lange</td>
<td>20-Oct-97</td>
</tr>
<tr>
<td>105</td>
<td>Williams</td>
<td>08-Nov-97</td>
</tr>
<tr>
<td>106</td>
<td>Duzak</td>
<td>05-Jan-04</td>
</tr>
<tr>
<td>107</td>
<td>Diante</td>
<td>02-Jul-97</td>
</tr>
<tr>
<td>108</td>
<td>Wesenbach</td>
<td>18-Nov-95</td>
</tr>
<tr>
<td>109</td>
<td>Travis</td>
<td>14-Apr-01</td>
</tr>
<tr>
<td>110</td>
<td>Genkazi</td>
<td>01-Dec-03</td>
</tr>
</tbody>
</table>

**Table name: PILOT (the subtype)**

<table>
<thead>
<tr>
<th>EMP_NUM</th>
<th>PIL_LICENSE</th>
<th>PIL_RATINGS</th>
<th>PIL_MED_TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>ATP</td>
<td>SEL/MEL/Instr/CFII</td>
<td>1</td>
</tr>
<tr>
<td>104</td>
<td>ATP</td>
<td>SEL/MEL/Instr</td>
<td>1</td>
</tr>
<tr>
<td>105</td>
<td>COM</td>
<td>SEL/MEL/Instr/CFII</td>
<td>2</td>
</tr>
<tr>
<td>106</td>
<td>COM</td>
<td>SEL/MEL/Instr</td>
<td>2</td>
</tr>
<tr>
<td>109</td>
<td>COM</td>
<td>SEL/MEL/SES/Instr/CFII</td>
<td>1</td>
</tr>
</tbody>
</table>

---
A Generalization Hierarchy with Overlapping Subtypes

- Supertype entity related to several unique and overlapping subtype entities
- Use ‘Gs’ Symbol
- Example:
  - An employee may be a student
  - A professor can be an administrator

**FIGURE 4.30** A Generalization Hierarchy with Overlapping Subtypes
E-R Modeling Symbols Comparison

• Summary of ER model concepts
  – Chen model
    • Moved conceptual design into practical database design arena
    • Became dominant player in CASE tool market late 1980’s & early 1990’s
  – Crow’s Foot model
    • Combines connectivity & cardinality info in a single symbol set
    • Cannot detail all cardinalities
  – Rein85 model
    • Similar to Crow’s Foot
    • Cannot recognize cardinalities explicitly
    • Operates at higher level of abstraction
  – IDEF1X model
    • Acronym for “Integrated DEFition 1X”
    • Derivative of Integrated computer-aided manufacturing (ICAM) studies in the late 1970’s
    • Used by airforce manufacturing
    • Uses fewer symbols
A Comparison of ER Modeling Symbols

**Figure 4.31** A Comparison of ER Modeling Symbols

<table>
<thead>
<tr>
<th>Chen</th>
<th>Crow’s Foot</th>
<th>Rein85</th>
<th>IDEF1X</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entity</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Relationship line</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Relationship</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Option symbol</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>One (1) symbol</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Many (M) symbol</td>
<td>M</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Composite entity</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Weak entity</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The Chen Representation of the Invoicing Problem

FIGURE 4.32 THE CHEN REPRESENTATION OF THE INVOICING PROBLEM

This model may be read as follows:

- Each CUSTOMER may generate one or more INVOICEs.
- Each INVOICE is generated by one CUSTOMER.
- Each INVOICE contains one or more invoice LINEs.
- Each invoice LINE is contained in an INVOICE.
- Each invoice LINE references one PRODUCT.
- Each PRODUCT may be referenced in one or more invoice LINEs.
The Crow’s Foot Representation of the Invoicing Problem

This model may be read as follows:

each CUSTOMER may generate one or more INVOICEs
each INVOICE is generated by one CUSTOMER

each INVOICE contains one or more invoice LINES
each invoice LINE is contained in one INVOICE

each invoice LINE references one PRODUCT
each PRODUCT may be referenced in one or more invoice LINES
The Rein85 Representation of the Invoicing Problem

This model may be read as follows:

- each CUSTOMER may generate one or more INVOICEs
- each INVOICE is generated by one CUSTOMER
- each INVOICE contains one or more invoice LINEs
- each invoice LINE is contained in an INVOICE
- each invoice LINE references one PRODUCT
- each PRODUCT may be referenced in one or more invoice LINEs
The IDEF1X Representation of the Invoicing Problem

This model may be read as follows:

- Each CUSTOMER may generate one or more INVOICEs.
- Each INVOICE is generated by one CUSTOMER.
- Each INVOICE contains one or more invoice LINEs.
- Each invoice LINE is contained in an INVOICE.
- Each invoice LINE references one PRODUCT.
- Each PRODUCT may be referenced in one or more invoice LINEs.
Developing an ER Diagram

- Database design is an iterative rather than a linear or sequential process
  - Based on repetition of processes and procedures
  1. General a detailed narrative of the organization's description of operations.
  2. Identify the business rules based on the description of the operations.
  3. Identify the main entities and relationships from the business rules.
  4. Develop the initial ERD
  5. Identify the attributes and primary keys that describe the entities.
  6. Revise and review the ERD.
- Repeat process until designers and users agree E-R Diagram complete
Case Study: Tiny College

- Step 1: General narrative of organizational operations developed (p. 157)
- Initial interview with “Tiny College” (TC) administration
  - TC is divided into several schools: school of business, school of arts & science, school of education, & school of applied science. Each school is administered by a dean.
- What are the Entities?
- What are the relationships?
- What are the business rules?
Case Study: Tiny College

FIGURE 2.11  TINY COLLEGE ENTITIES

Entities

STUDENT  PROFESSOR  COURSE  CLASS  ROOM

Student number
Social Security number
First name
Last name
Middle initial
Sex
Date of birth
Home address
Home phone
College address
College phone
Major
Hours completed
Grade point average

STUDENT attributes
Case Study: Tiny College

• What are the Entities?
  – DEAN
  – SCHOOL
  – PROFESSOR
  – EMPLOYEE
  – ADMINISTRATOR
  – CLASS
  – COLLEGE
  – COURSE
  – DEPARTMENT
  – STUDENT
  – ROOM
  – BUILDING
  – …
Case Study: Tiny College

- Conceptual model

**Figure 2.12** A Conceptual Model for Tiny College
Case Study: Tiny College

- External model

Figure 2.13: A Division of an Internal Model into External Models
Case Study: Tiny College

• Supertype/Subtype Relationship
• ERD for DEAN/SCHOOL?
  – A dean is an administrator but also is a professor
  – Administrator & professor are employees
  – Other employees are secretaries, custodians, drivers, …
  ⇒ we need to have some supertype/subtype relationship
Case Study: Tiny College

- EMPLOYEE _ PROFESSOR relationship
  - PROFESSOR is optional to EMPLOYEE
  - Not all employees are professors

![Diagram of a supertype/subtype relationship in an ERD](image)
Case Study: Tiny College

- First ERD Segment: SCHOOL-DEPT relationship
  - Each school is composed of several departments.
  - The smallest number of departments operated by a school is one
  - Each department belongs to a single school

**FIGURE 4.38** The First Tiny College ERD Segment
Case Study: Tiny College

- First ERD Segment: PROFESSOR-SCHOOL relationship
  - A professor can be the Dean of a school or not a dean at all
  - Each school must have exactly one dean
  - School is optional to the dean relationship

Figure 4.38 The First Tiny College ERD Segment
Case Study: Tiny College

- Second ERD Segment: DEPT-COURSE relationship
  - Each department offers zero or more courses
  - Each course must belong to a specific (one & only one) department

![The Second Tiny College ERD Segment](image)
Case Study: Tiny College

- Third ERD Segment: COURSE-CLASS relationship
  - A CLASS is a section of a COURSE
  - Each course may offer several classes (sections)
  - A class may not be offered but exist in the college catalog => CLASS is optional to COURSE
Case Study: Tiny College

- Fourth ERD Segment: EMPLOYEE-PROFESSOR-DEPARTMENT relationship
  - Each department has many professors
  - One of the professors is the chair of the department
  - Only one professor can be the department chair
  - No professor is required to accept the chair position
    - => DEPARTMENT is optional to PROFESSOR
Case Study: Tiny College

- Fifth ERD Segment: EMPLOYEE-PROFESSOR-CLASS relationship
  - Each professor may teach up to 4 classes
  - Each class is taught by a professor at a given time in a given place
  - A professor may be on research contract, or sabbatical. In which case he will not be assigned classes

FIGURE 4.42 THE FIFTH TINY COLLEGE ERD SEGMENT
Case Study: Tiny College

• Sixth ERD Segment: STUDENT-ENROLL-CLASS relationship
  – A student may enroll in several classes
  – A class can be taken once during current enrollment period
  – A class can exist even before students are enrolled => STUDENT is optional to CLASS
  – M:N relationship is converted to 2 1:M relationship by adding ENROLL entity
  – A class with no students does not occur in ENROLL
  – ENROLL is a weak entity. It’s existence dependent. Its PK is composed of both PK’s of STUDENT & CLASS

**FIGURE 4.43 THE SIXTH TINY COLLEGE ERD SEGMENT**
Case Study: Tiny College

- Seventh ERD Segment: DEPT-STUDENT relationship
  - A department has many students
  - Many majors are offered by a department
  - A student can only have a single major
    - => A student is associated with a single department
  - It is possible that a student doesn’t have a major
    - => STUDENT is optional to DEPARTMENT
Case Study: Tiny College

- Eighth ERD Segment: PROFESSOR-STUDENT relationship
  - Each student has a PROFESSOR advisor
  - A PROFESSOR advises many students
  - Not every PROFESSOR acts as an advisor
  - \( \Rightarrow \) STUDENT is optional to PROFESSOR
Case Study: Tiny College

- Ninth ERD Segment: BUILDING-ROOM-CLASS relationship
  - A CLASS is taught in a ROOM
  - Each ROOM is located in a BUILDING
  - A BUILDINGS can contain many ROOMs
  - Each ROOM is found in a single BUILDING
  - Some BUILDINGS don’t have classrooms
  - Some ROOMs are not used as CLASSes

**FIGURE 4.46** The Ninth Tiny College ERD Segment
## Components of the ER Model

<table>
<thead>
<tr>
<th>ENTITY</th>
<th>RELATIONSHIP</th>
<th>CONNECTIVITY</th>
<th>ENTITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCHOOL</td>
<td>operates</td>
<td>1:M</td>
<td>DEPARTMENT</td>
</tr>
<tr>
<td>DEPARTMENT</td>
<td>has</td>
<td>1:M</td>
<td>STUDENT</td>
</tr>
<tr>
<td>DEPARTMENT</td>
<td>employs</td>
<td>1:M</td>
<td>PROFESSOR</td>
</tr>
<tr>
<td>DEPARTMENT</td>
<td>offers</td>
<td>1:M</td>
<td>COURSE</td>
</tr>
<tr>
<td>COURSE</td>
<td>generates</td>
<td>1:M</td>
<td>CLASS</td>
</tr>
<tr>
<td>PROFESSOR</td>
<td>is an</td>
<td>1:1</td>
<td>EMPLOYEE</td>
</tr>
<tr>
<td>PROFESSOR</td>
<td>is dean of</td>
<td>1:1</td>
<td>SCHOOL</td>
</tr>
<tr>
<td>PROFESSOR</td>
<td>chairs</td>
<td>1:1</td>
<td>DEPARTMENT</td>
</tr>
<tr>
<td>PROFESSOR</td>
<td>teaches</td>
<td>1:M</td>
<td>CLASS</td>
</tr>
<tr>
<td>PROFESSOR</td>
<td>advises</td>
<td>1:M</td>
<td>STUDENT</td>
</tr>
<tr>
<td>STUDENT</td>
<td>enrolls in</td>
<td>1:M</td>
<td>CLASS</td>
</tr>
<tr>
<td>BUILDING</td>
<td>contains</td>
<td>1:M</td>
<td>ROOM</td>
</tr>
<tr>
<td>ROOM</td>
<td>is used for</td>
<td>1:M</td>
<td>CLASS</td>
</tr>
</tbody>
</table>
Challenge of Database Design: Conflicting Goals

- Database design must conform to design standards. Without design standards, it could be almost impossible to achieve a proper design process, or evaluate or modify an existing design.
- The design should minimize data redundancies and null values.
- High processing speeds are often a top priority in database design. Sometimes this might require the use of 1:1 relationships to avoid nulls, combine tables to avoid additional relationships, or include derived attributes.
- Quest for timely, but possibly complex, information might be the focus of database design. This might require adding more entities and attributes, which in turn will affect transactions speed.
- The cost of a slight loss of transaction speed at the front end and the addition of derived attributes is likely to pay off in the process of report or query generation.
Challenge of Database Design: Conflicting Goals

• Even when the designer is focusing on the entities, attributes, relationships, and constraints, he/she should also consider the effect of update, retrieval, and deletion of data, reporting and query requirements, and end user-requirements such as:
  – Security
  – Performance
  – Shared access
  – Data integrity
  – Changes due to operational requirements
Challenge of Database Design: Conflicting Goals

- You also need to keep in mind that the ERD might require changes either due to problems or operational requirements.
- There might be more than one way to implement a specific concept.
- It is the designer role to use his/her professional judgment to find out the best option.
- Example:
  - There are three possible ways to implement a 1:1 recursive relationship.
Three 1:1 Relationship Implementations

**FIGURE 4.48 VARIOUS IMPLEMENTATIONS OF A 1:1 RECURSIVE RELATIONSHIP**

**Table name: EMPLOYEE_V1**

<table>
<thead>
<tr>
<th>EMP_NUM</th>
<th>EMP_LNAME</th>
<th>EMP_FNAME</th>
<th>EMP_SPOUSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>345</td>
<td>Ramirez</td>
<td>James</td>
<td>347</td>
</tr>
<tr>
<td>346</td>
<td>Jones</td>
<td>Anne</td>
<td>349</td>
</tr>
<tr>
<td>347</td>
<td>Ramirez</td>
<td>Louise</td>
<td>345</td>
</tr>
<tr>
<td>348</td>
<td>Delaney</td>
<td>Robert</td>
<td>346</td>
</tr>
<tr>
<td>349</td>
<td>Shapiro</td>
<td>Anton</td>
<td>346</td>
</tr>
</tbody>
</table>

**Database name: Ch04_PartCo**

First implementation

**Table name: EMPLOYEE**

<table>
<thead>
<tr>
<th>EMP_NUM</th>
<th>EMP_LNAME</th>
<th>EMP_FNAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>345</td>
<td>Ramirez</td>
<td>James</td>
</tr>
<tr>
<td>346</td>
<td>Jones</td>
<td>Anne</td>
</tr>
<tr>
<td>347</td>
<td>Ramirez</td>
<td>Louise</td>
</tr>
<tr>
<td>348</td>
<td>Delaney</td>
<td>Robert</td>
</tr>
<tr>
<td>349</td>
<td>Shapiro</td>
<td>Anton</td>
</tr>
</tbody>
</table>

**Table name: MARRIED_V1**

<table>
<thead>
<tr>
<th>EMP_NUM</th>
<th>EMP_SPOUSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>345</td>
<td>347</td>
</tr>
<tr>
<td>346</td>
<td>349</td>
</tr>
<tr>
<td>347</td>
<td>345</td>
</tr>
<tr>
<td>349</td>
<td>346</td>
</tr>
</tbody>
</table>

Second implementation

**Table name: MARRIAGE**

<table>
<thead>
<tr>
<th>MAR_NUM</th>
<th>MAR_DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>04-Mar-03</td>
</tr>
<tr>
<td>2</td>
<td>02-Feb-99</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMP_NUM</th>
<th>EMP_LNAME</th>
<th>EMP_FNAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>345</td>
<td>Ramirez</td>
<td>James</td>
</tr>
<tr>
<td>346</td>
<td>Jones</td>
<td>Anne</td>
</tr>
<tr>
<td>347</td>
<td>Ramirez</td>
<td>Louise</td>
</tr>
<tr>
<td>348</td>
<td>Delaney</td>
<td>Robert</td>
</tr>
<tr>
<td>349</td>
<td>Shapiro</td>
<td>Anton</td>
</tr>
</tbody>
</table>

**Table name: MARPART**

<table>
<thead>
<tr>
<th>MAR_NUM</th>
<th>EMP_NUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>345</td>
</tr>
<tr>
<td>2</td>
<td>346</td>
</tr>
</tbody>
</table>

The Relational Schema for the Third Implementation

**Third implementation**

<table>
<thead>
<tr>
<th>MARRIAGE</th>
<th>1</th>
<th>MAR_DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAR_NUM</td>
<td>0</td>
<td>MAR_DATE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MARPART</th>
<th>1</th>
<th>EMP_NUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAR_NUM</td>
<td>1</td>
<td>EMP_NUM</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EMPLOYEE</th>
<th>EMP_NUM</th>
<th>EMP_LNAME</th>
<th>EMP_FNAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>345</td>
<td>Ramirez</td>
<td>James</td>
<td></td>
</tr>
<tr>
<td>346</td>
<td>Jones</td>
<td>Anne</td>
<td></td>
</tr>
<tr>
<td>347</td>
<td>Ramirez</td>
<td>Louise</td>
<td></td>
</tr>
<tr>
<td>348</td>
<td>Delaney</td>
<td>Robert</td>
<td></td>
</tr>
<tr>
<td>349</td>
<td>Shapiro</td>
<td>Anton</td>
<td></td>
</tr>
</tbody>
</table>
Three 1:1 Relationship Implementations

- EMPLOYEE_V1 uses a recursive relationship. It might cause data anomalies
  - If a couple divorce, more than one record need to be updated, otherwise inconsistent data
    will occur.
  - This implementation might have too many null values.
  - There is no way to prevent entering data about one employee being married to several employees.
- The second implementation (EMPLOYEE and MARRIED_V1) eliminates the null values for unmarried employees, but
  - Duplicate data for married couples and are likely to produce inconsistent data
  - Cannot prevent having one employee married to several other employees
- The third implementation (MATRRIAGE, MAPART, and EMPLOYEE) seems to be the best option but
  - Need to use a unique index on EMP_NUM attribute in the MAPART table to ensure that an employee occurs only once in any given marriage.
Documentation

• Documentation, documentation, documentation!!!
  – Documentation is very important part of the designer's job
  – Put all design activities in writing
  – The development of organizational documentation standards is a very important aspect of ensuring data compatibility and coherence
Summary

• Entity relationship (ER) model
  – Uses ER diagrams to represent conceptual database as viewed by the end user
  – Three main components
    • Entities
    • Relationships
    • Attributes
  – Includes connectivity and cardinality notations
• Connectivities and cardinalities are based on business rules
Summary (continued)

- ER symbols are used to graphically depict the ER model’s components and relationships
- ERDs may be based on many different ER models
- Entities can also be classified as supertypes and subtypes within a generalization hierarchy
- Database designers are often forced to make design compromises