Conceptual Data Modeling

(Oppel Chapter 5)

Slides courtesy of Andy Oppel

"Essentially, all models are wrong, but some are useful"

George E. P. Box

Statistician

10/18/1919 – 3/28/2013

The Conceptual Modeling Process

- Preparation
  - Establish Objectives
  - Collect and Evaluate Inputs
    - Requirements
    - Patterns
    - Generic or standard models
  - Establish Conceptual Modeling Guidelines
    - Inclusion of attributes
    - Naming conventions
    - Generalization versus specialization
    - Subject area selection

The Conceptual Modeling Process (2)

- Solution Design
  - Sort out various design alternatives
  - First opportunity to review a model and adjust/correct
  - Validation of requirements
  - Review entities and relationships

Conceptual versus Logical

Criteria	Conceptual Model	Logical Model
Attributes included?	Optional	Always
Generalized or specialized?	Commonly generalized	Generalization to match expected physical model
Model normalized?	Seldom; M:N relationships common	Always for OLTP models; somewhat for OLAP
Multiple alternatives shown?	This is the best place to show them	Seldom
Layers of detail shown?	Multiple levels of detail aimed at different audiences are common	Most practitioners strive for a consistent level of detail throughout

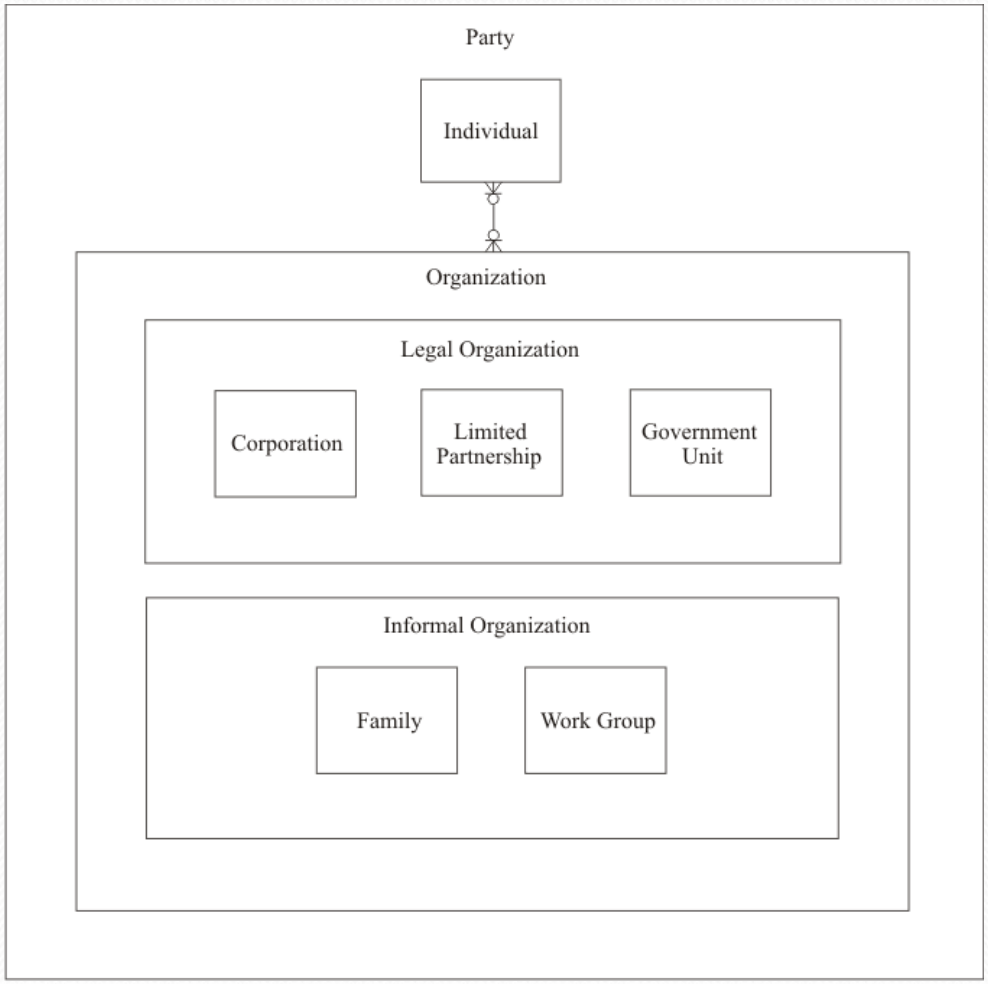
Creating the Model

- Skipping the Conceptual Modeling step will likely cost you more time in the long run
- Important to involve:
  - SMEs
  - Project Team members
  - Project Sponsor
  - Others who can offer additional perspectives

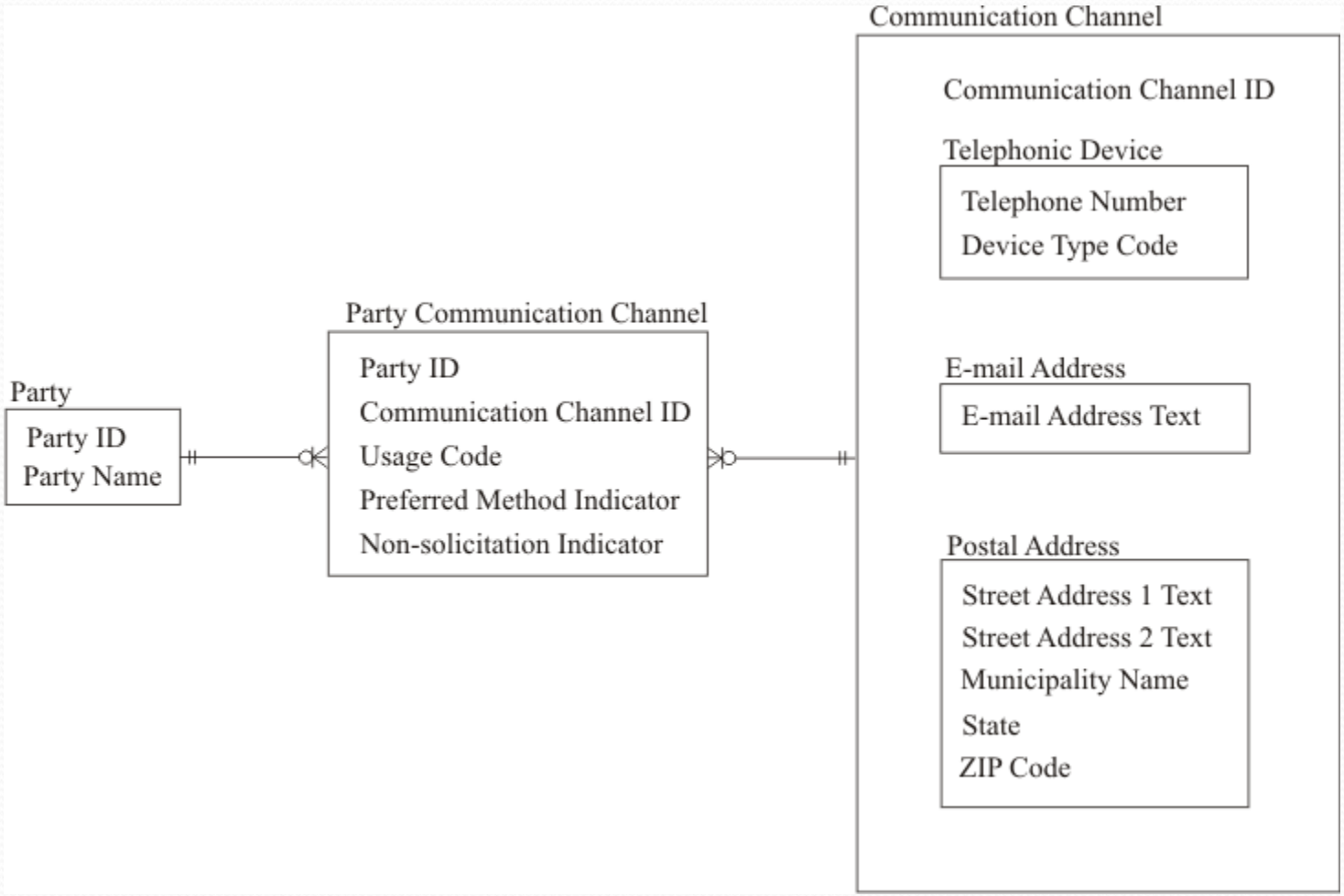
Generic Models and Patterns

- Few models are created completely from scratch
- Practitioners rely on common patterns, existing models, industry-specific models, and generic models such as Len Silverston's Universal Data Model (*adapted*, not *copied verbatum*)
- The next several slides show two commonly used patterns: The Party structure and Communication Channel structure.

The Party Structure



The Communication Channel Structure



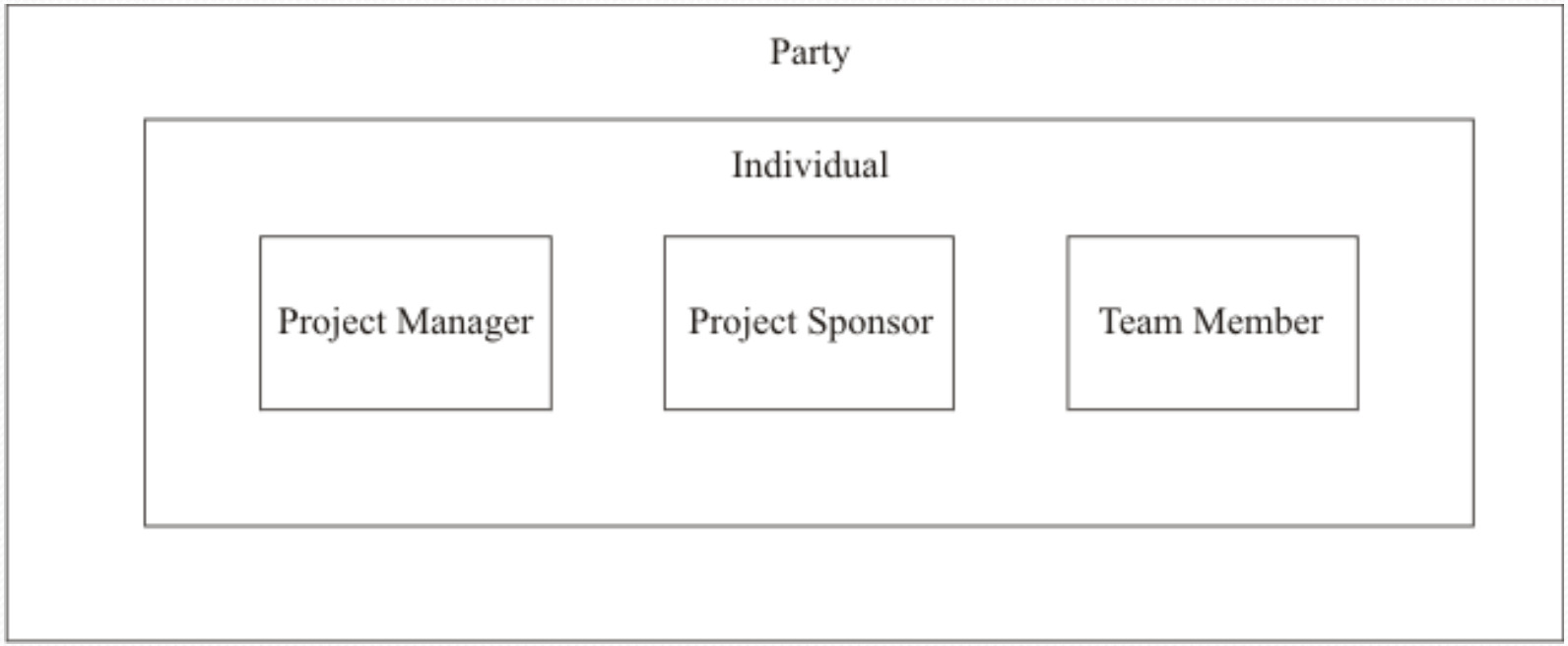
First Cut Diagrams

- Expect revisions
- Principles:
  - Include the most important entities
  - Relationships show maximum cardinality and are unnamed
  - Attributes included when necessary for clarity
  - Stay at a high level, excluding logs, audit data, exception handling, history
  - Show subclasses where data/relationships are markedly different, or when the business handles the subclasses significantly differently

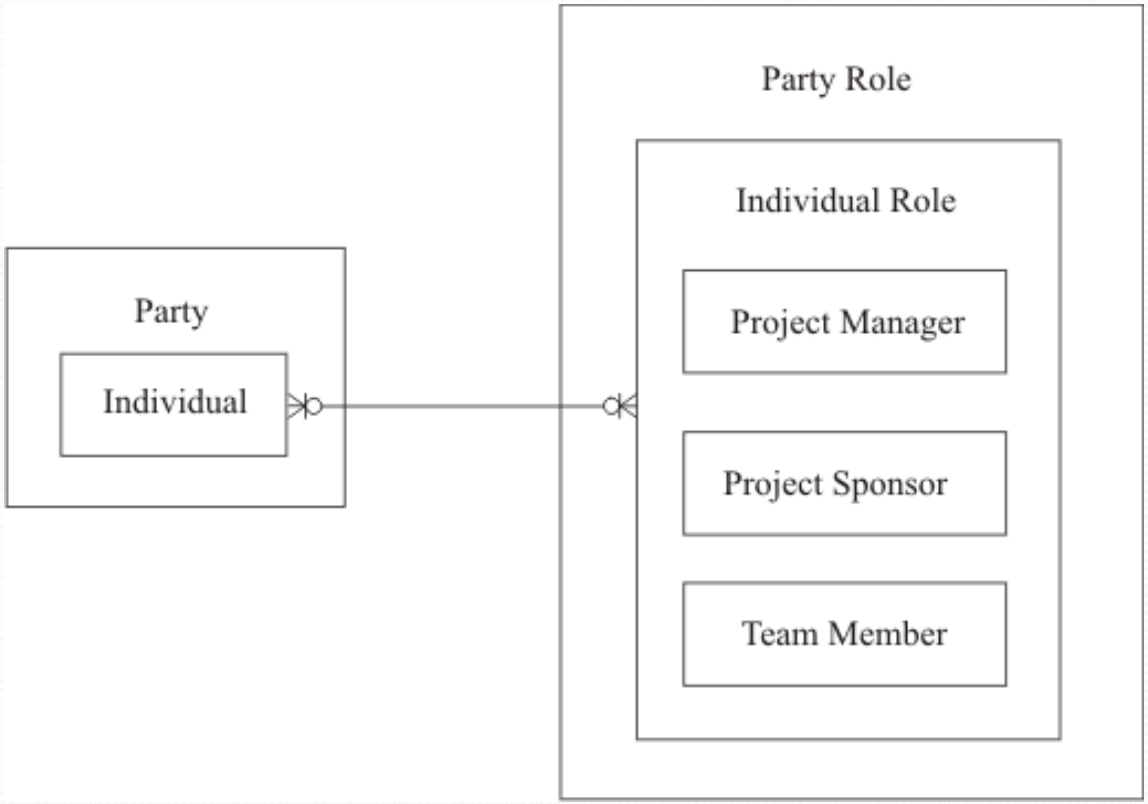
Roles versus Subtypes

- Subtypes:
  - Should not overlap (every instance of supertype should map to only one subtype)
  - Must cover all possible cases
- Roles:
  - Any instance can have multiple roles
  - Only the most important roles need to be spelled out on the model
  - More flexible:
    - New roles require only the insert of a row for the role type
    - New instances require only an insert into the intersection

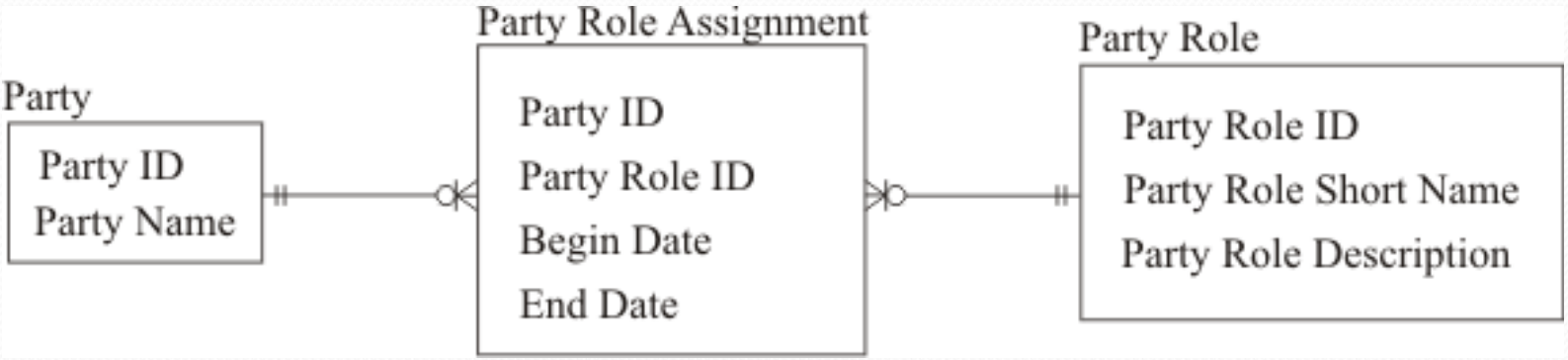
Party Structure with Subtypes



Party Structure with Roles



Party Role Logical Model



Party Role Tables

Party

Party ID	Party Name
101	Tiana C. Steger
102	Andrew T. Guay
103	Sherry L. Johnson

Party Role Assignment

Party ID	Party Role ID	Begin Date	End Date
101	PS	01/05/2009	12/31/2010
101	PM	01/05/2009	06/30/2009
102	TM	01/05/2009	12/31/2010
103	PM	07/01/2009	12/31/2010
103	TM	07/01/2009	12/31/2010

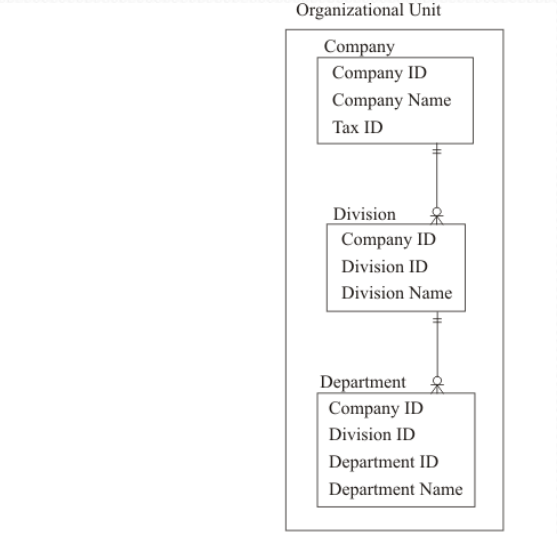
Party Role

Party Role ID	Party Role Short Name	Party Role Description
PS	Prj Spnsr	Project Sponsor
PM	Prj Mgr	Project Manager
TM	Mbr	Project Team Member

Modeling Hierarchies

- Characteristics of Hierarchies:
  - Parents can have many child entity classes
  - Each child entity class can have only one parent
- Specialized Model Issues:
  - Each new layer requires a new entity and relationship
  - If some instances have missing layers, dummy records are required
  - Rigid (inflexible)
- Generalized Model Issues:
  - More difficult for business users to understand
  - Queries more confusing/awkward

Rigid (Specialized) Hierarchy



Company

Company ID	Company Name	Tax ID
M100	Acme Industries	07-7542678
M110	Northwest Manufacturing	07-9426104

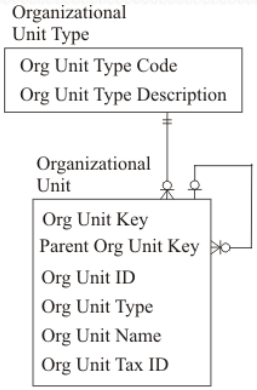
Division

Company ID	Division ID	Division Name
M100	01	Rocketry
M100	02	Safety Equipment

Department

Company ID	Division ID	Department ID	Department Name
M100	01	M1	Manufacturing
M100	01	S1	Government Sales
M100	01	S2	Non-Government Sales
M100	02	S1	Government Sales
M100	02	S5	Retail Sales

Flexible (Generalized) Hierarchy



Organizational Unit Type

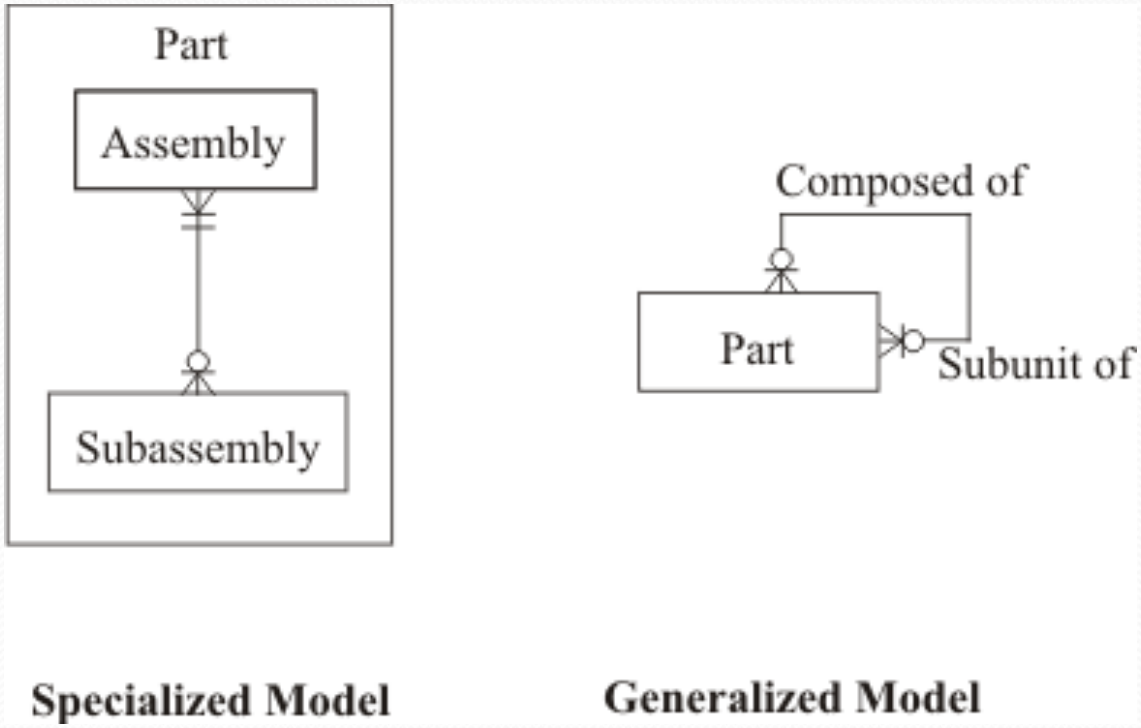
Org Unit Type Code	Org Unit Type Description
CO	Company
DIV	Division
DEPT	Department

Organizational Unit Type

Org Unit Key	Parent Org Unit Key	Org Unit ID	Org Unit Type Code	Org Unit Name	Org Unit Tax ID
1001		M100	CO	Acme Industries	07-7542678
1002	1001	01	DIV	Rocketry	
1003	1002	M1	DEPT	Manufacturing	
1004	1002	S1	DEPT	Government Sales	
1005	1002	S2	DEPT	Non-Government Sales	
1006	1001	02	DIV	Safety Equipment	
1007	1006	S1	DEPT	Government Sales	
1008	1006	S5	DEPT	Retail Sales	
1009		M110	CO	Northwest Manufacturing	07-9426104

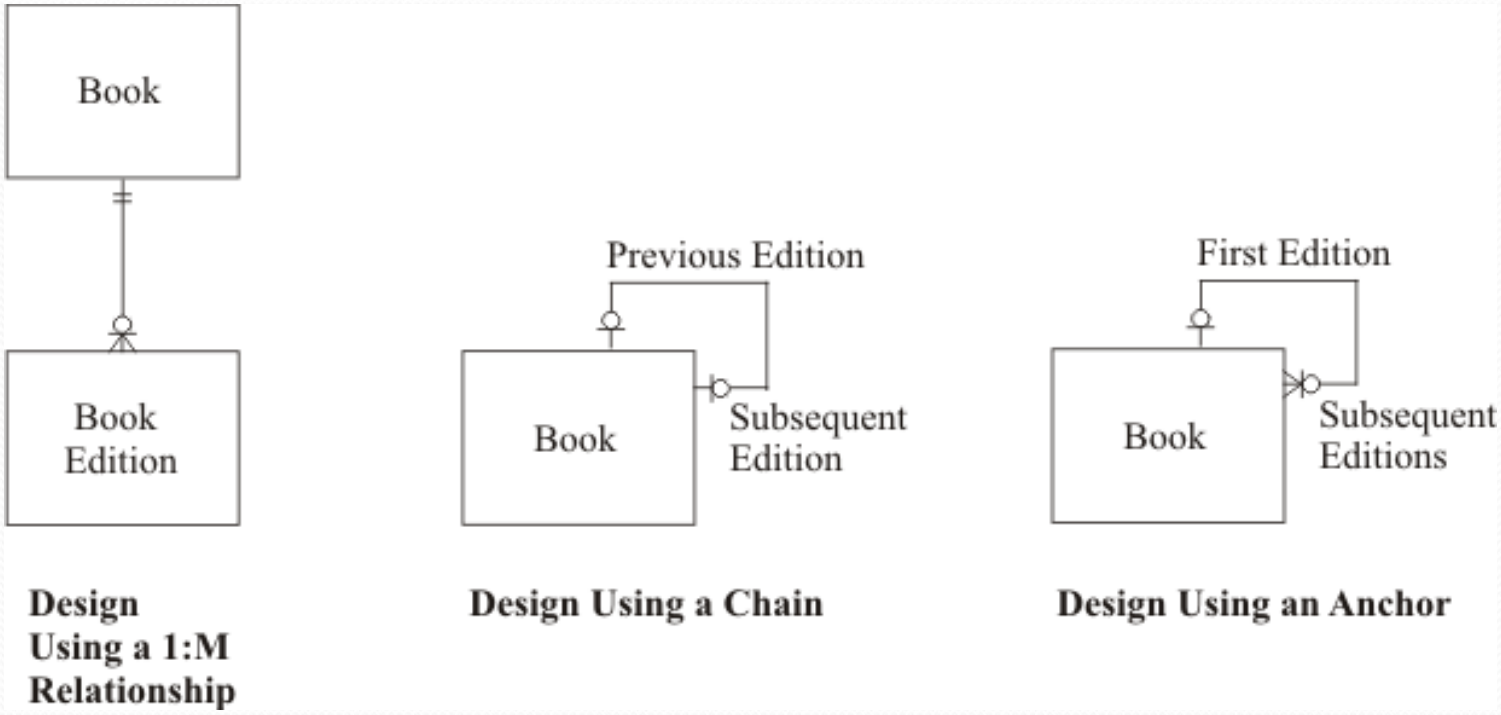
Network Structures

- Like hierarchy, but without single-parent restriction
- Common example is bill of materials in manufacturing



Linked List Structures

- Commonly called chains
- Each entity instance is linked to one other entity instance in one direction or both directions



Design Using a 1:M Relationship

Design Using a Chain

Design Using an Anchor

Bottom Up vs. Top Down Modeling

- Bottom Up
  - Start with a literal interpretation that is close to a physical table
  - Work up to more generalized structures
  - Works well in workshop-style design sessions
  - Note that normalization is a bottom up approach
- Top Down
  - Start with a general concept and work downwards
  - Works well in groups only if participants are abstract thinkers (literal thinkers will have difficulty following)

Subject Areas

- Use to break large problems into smaller ones
- Facilitate parallel development of models
- Avoid selecting subject areas based on business processes or components of your organization
- Synthesize models so any shared entities are identical in all subject area models
- Standard colors for each subject area highly recommended

What Makes a Good Conceptual Model?

- Completeness
- Conciseness
- Precision
- Balance
- Process support (processes map to the model)
- Understandability