Database Design

- Requirements Analysis
- Conceptual Database Design
  - Creates an abstract model
- Logical Database Design
  - Converts abstract model to concrete schema
- Schema Refinement
  - Improve schema
- Physical Database Design
  - Improve performance
- Application and Security Design
  - Modify schema for application considerations
Entity-Relationship (ER) Model

- Conceptual design model
- Describes the data in terms of entities (things) and their relationships
  - Attempts to model the “real world”
Entities and Attributes

• An *entity* is a thing, distinguishable from other things
  – E.g., a person

• An *entity set* is a collection of similar things
  – E.g., people

• Entities are not necessarily physical objects
  – E.g., a baseball game
Entities and Attributes

• An entity is described by a set of attributes
  – E.g., a person might have a name and social security number
• An attribute has a domain of possible values
  – E.g., name might be a 50 character string
Keys

• The *key* for an entity set is a minimal set of attributes which uniquely identifies each entity
  – Might be single attribute or multiple attributes
  – E.g., social security number for employees

• Might be more than one candidate key
  – If so then one is chosen as the *primary key*
Entity Sets

- Entity sets are represented by rectangles
- Attributes are ovals
Relationships

• A *relationship* is an association between two (or more) entities

• A *relationship set* is the set of relationships between two (or more) entity sets
  – E.g., `works_in` might be a relationship set between the employees entity set and the departments entity set

• A relationship set can have *descriptive attributes*
  – Information about the relationships, not the entities
• Relationship sets are diamonds
Roles

• The entity sets participating in a relationship set need not be unique
  – E.g., a reports_to relationship set might exist between employees and employees

• The entity set then plays different roles in the relationship set
  – E.g., supervisor and underling
Key Constraints

• Relationship sets can be
  – Many-to-many
    • E.g., an employee might work in multiple departments and a department might have multiple employees
  – One-to-many
    • E.g., an employee might manage multiple departments but each department has only one manager
  – One-to-one
    • E.g., each department has only one manager and each employee is only allowed to manage one department
Many-to-Many

• An employee might work in multiple departments and a department might have multiple employees.
One-to-Many

• An employee might manage multiple departments but each department has only one manager.
One-to-Many

- An employee might manage multiple departments but each department has only one manager.
One-to-Many

- An employee might manage multiple departments but each department has only one manager.
One-to-One

- Each department has only one manager and each employee is only allowed to manage one department.
One-to-One

- Each department has only one manager and each employee is only allowed to manage one department
One-to-One

- Each department has only one manager and each employee is only allowed to manage one department.
Participation Constraints

• The participation of an entity set in a relationship set can be
  – Total: every entity must participate in the relationship set
    • E.g., every department must have a manager
  – Partial: not every entity must participate
    • E.g., not every employee manages a department
Participation Constraints

Partial Participation

- 111-11-1111
- 222-22-2222
- 333-33-3333

manages

Total Participation

- employees
- manages
- departments

- 42
- 35
- 76
Participation Constraints

- employees
  - ssn
  - name
- manages
- departments
  - id
  - name

Partial Participation
Total Participation
Weak Entity Sets

- *Weak entity sets* are entity sets whose attributes do not uniquely identify entities without including the primary key of the *identifying owner*
  - The *identifying relationship set* is a one-to-many from the owning entity set to the weak entity set
Weak Entity Sets

- dname does not uniquely identify a dependent
  - Two employees might have dependents with the same name
- Must use the primary key of the identifying owner (the ssn of the employee) together with the dname
Class Hierarchies

- Sometimes, entities are classified into subsets with different attributes
  - Attributes from the parent entity set are inherited by the child entity sets, but the child entity sets have other attributes
    - E.g., the employees entity set might have the subclasses hourly_employees and salaried_employees
  - Defines an *is-a* relationship
Class Hierarchies

- **employees**
  - ssn
  - name
  - hourly_wage
  - salary
- **ISA**
- **hourly_emps**
- **salaried_emps**
Aggregation

- Aggregation allows a relationship set to participate in another relationship set
  - E.g., suppose departments sponsor projects and assign employees to monitor the sponsorships
  - We might want "monitors" to be a relationship set between "employees" and "sponsors", but relationship sets associate two or more entities, not other relationship sets
Design Considerations

• Entity vs Attribute
  – Not always clear if a property should be modeled as an attribute or another entity set
    • E.g., should address be an attribute of employee or a separate entity set
• Entity vs Relationship
  – Sometimes what seems like a relationship set might be better modeled as an entity set
    • E.g., instead of a manages relationship set it might be better to have a managers entity set (as a subclass of employees)
Design Considerations

- Binary vs Ternary Relationships
- Aggregation vs Ternary Relationships