CSC 321/621
1/24/2012 Lecture #1 Outline

Purpose of the reading and lecture

- Big concept: Entity-relationship model and associated diagram
- Goal: Model the information needed to represent a real-world system, particularly items that exist in that system and how they interact with each other
- Note this is all going to be descriptive –we are in a **design** phase!
- Note this is NOT specific to databases – we could employ this in OO design as well.
- Expectations:
    - Be able to understand an ER diagram
    - Be able to create an ER diagram from a description of a real-world system

Review DreamHome setup

Summary of ER diagrams

- Previously discussed notion of entity, attributes, and relationships
    - Entity: Similar to a class in OO design (meta-concept, template)
    - Attributes: Descriptive properties of that entity (similar to member variables/fields in a class)
    - Relationships: Interactions between entities (in OO, typically requires a member variable relating to that relationship)
        - Can also have attributes (describing the interaction)
        - Can be arbitrarily complex (ternary, quaternary, recursive, etc)
    - We are designing classes, not instantiating objects of a particular class
    - Another way of looking at this: Nouns (entities), Descriptors (attributes/properties), Verbs (relationships)
- Representing such information is done with a diagramming tool built for object-oriented design – see book Figure 12.1 -- called a UML diagram
    - Rectangles: top box – entity, rest of box – attributes
    - Lines between rectangles: relationships -- line only if binary, diamond if ternary or higher
        - If binary, label on edge, with an arrow (usually indicating meaningful in one direction – take the more "active" direction [a manager SUPERVISES staff instead of a staff member IS_SUPERVISED_BY a manager])
        - If ternary or higher, place label in diamond
        - If has attributes, place in box like entity, but without name on the box, use a dashed line to connect to relationship line
        - May provide role names at edges of relationship
- Attribute issues we may need to be concerned with: simple or composite [composite: can be broken down further [like address], single or multi-valued [multi-valued: an office can have 2 phone numbers], base or derived [derived: constructed from other base or derived attributes: deposit = 2x rent]
    - For now, let's target: simple, single-valued, base

- Some attributes become particularly important when identify particular entity occurrences
    - Sets of attributes (1 or more) that provide a unique ID for that entity – called "keys"
        - Candidate keys – any set of attributes that uniquely identifies and can't be broken down further
        - Primary key – a candidate key selected to represent the "ID" mechanism (maybe chosen because it required fewest attributes, or least amount of information)
            - Attributes that define a primary key are often labeled {PK} in the diagram
            - There is a notion that if a PK for an entity doesn't exist, then an entity occurrence it is only uniquely identifiable within a relationship (so it relies on another entities primary key) – called a weak (child, dependent, subordinate) entity; ones with a PK are "strong entity" or (parent, owner, dominant)
                - Weak entities occurrences should disappear when associated strong entity occurrences disappear (as they exist within the context of the associated strong entity)
                - Player (name, number) PLAYS_ON Team (name)
                    - Player is WEAK
    - Relationships – can associate structural constraints on relationships
        - Multiplicity – defining the number or range of possible occurrences of an entity type that may relate to a single occurrence of an associated entity type
            - Set one one side at a time
            - Typically write in a range, even if it is a set value (1..1 is written when want to say 1)
            - Label both (all) ends of relationship arrows – start with the direction the relationship is labeled.
            - For ternary or higher relationships: fix n-1 Entities to specific values, what is number of values of nth Entity that can be set?
        - Cardinality: developed from high end of ranges on both sides – max number of possible relationship occurrences, summarized in one of three ways: one-to-one (1:1), one-to-many (1:*), many-to-many (*:*)
        - Participation: low end of range – participation is optional if 0, mandatory if 1 or higher; note you interpret from values on opposite side of relationship: staff oversees property is labeled 0..100 on property side - means staff could not participate at all if overseeing 0 properties [so, not required to oversee properties ➔ staff has optional participation in that relationship]

        - Note: CARDINALITY OF A RELATIONSHIP (1:many) [descriptive of relationship], PARTICIPATION OF AN ENTITY IN AN A RELATIONSHIP (MANDATORY or OPTIONAL) [descriptive of entity], MULTIPLICITY: RANGE OF ENTITIES THAT CAN PARTICIPATE [descriptive of entity]
- A semantic net demonstrates actual entity and relationship occurrences (like instantiations of objects) with lines representing how two (or more) entities interaction in a relationship

Good vs bad in design…

Nouns: entities, Verbs: relationships

Avoid redundancy – don't re-represent information in diagram

Don’t represent information for an entity that is part of the information of an entity that it is related to

Don’t include ownerName as an attribute of Dog if there is an Owns relationship with an Owner entity

Don’t include attributes that are essentially descriptors of other attributes and will be repeated: Beer having attributes of name, manufacturer, and manufacturer address – manufactur address is really a descriptor of manufacturer, not of Beer → break into a Manufactures relationship and Manufacturer Entity

Try to avoid weak entity sets – usually there is a key we can come up with

Relationship attributes – I have seen having directly as relationship attributes as well as adding an additional entity with attributes that reflects the relationship (see example answers – such as “VideoAgreement” Registration entity)