CS W4111.001
Introduction to Databases
Spring 2017

Computer Science Department
Columbia University

Conceptual Design Using the
Entity-Relationship (ER) Model
Overview of Database Design

❖ **Conceptual design:** (ER Model is used at this stage.)
  - What are the *entities* and *relationships* in the enterprise?
  - What information about these entities and relationships should we store in the database?
  - What are the *integrity constraints* or *business rules* that hold?
  - A database ‘schema’ in the ER Model can be represented pictorially (*ER diagrams*).
  - Can map an ER diagram into a relational schema.

❖ **Schema Refinement:** (Normalization) Check relational schema for redundancies and related anomalies.

❖ **Physical Database Design and Tuning:** Consider typical workloads and further refine the database design.

---

ER Model Basics

❖ **Entity:** Real-world object distinguishable from other objects.
  An entity is described (in DB) using a set of *attributes*.

❖ **Entity Set:** A collection of similar entities. E.g., all employees.
  - All entities in an entity set have the same set of attributes. (Until we consider IS-A hierarchies, anyway!)
  - Each entity set has a *key*.
  - Each attribute has a *domain*.
  - Can map entity set to a relation easily.
Relationship: Association among 2 or more entities. E.g., Attishoo works in Pharmacy department.

Relationship Set: Collection of similar relationships. An n-ary relationship set R relates n entity sets E₁ ... Eₙ; each relationship in R involves entities e₁, ..., eₙ. Same entity set could participate in different relationship sets, or in different “roles” in same set.

Relationship sets can also have descriptive attributes (e.g., the since attribute of Works_In).
Key Constraints

- Consider Works_In: An employee can work in many departments; a dept can have many employees.
- In contrast, each dept has at most one manager, according to the key constraint on Manages.

Participation Constraints

Does every department have a manager?
If so, this is a participation constraint: the participation of Departments in Manages is said to be total (vs. partial)
**Weak Entities**

A *weak entity* can be identified uniquely only by considering the primary key of another (*owner*) entity.

- Owner entity set and weak entity set must participate in a one-to-many relationship set (one owner, many weak entities).
- Weak entity set must have total participation in this *identifying* relationship set.

![Weak Entity Diagram]

**ISA (‘is a’) Hierarchies**

- As in C++, or other PLs, attributes are inherited.
- If we declare A *ISA* B, every A entity is also considered to be a B entity. (Query answers should reflect this: *unlike C++!*)

  - **Overlap constraints:** Can Joe be an Hourly_Emps as well as a Contract_Emps entity? *(Allowed/disallowed)*
  - **Covering constraints:** Does every Employees entity also have to be an Hourly_Emps or a Contract_Emps entity? *(Yes/no)*

- Reasons for using ISA:
  - To add descriptive attributes specific to a subclass.
  - To identify entities that participate in a relationship.
Aggregation

- Used when we have to model a relationship involving (entity sets and) a relationship set.
- Aggregation allows us to treat a relationship set as an entity set for purposes of participation in (other) relationships.

Aggregation vs. ternary relationship:
- Monitors is a distinct relationship, with a descriptive attribute.
- Also, can say that each sponsorship is monitored by at most one employee.

Conceptual Design Using the ER Model

- Design choices:
  - Should a concept be modeled as an entity or an attribute?
  - Should a concept be modeled as an entity or a relationship?
  - Identifying relationships: Binary or ternary? Aggregation?

- Constraints in the ER Model:
  - A lot of data semantics can (and should) be captured.
  - But some constraints cannot be captured in ER diagrams.

- Need for further refining the schema:
  - Relational schema obtained from ER diagram is a good first step. But ER design subjective & can’t express certain constraints; so this relational schema may need refinement.
**Entity vs. Attribute**

- Should *address* be an attribute of Employees or an entity (connected to Employees by a relationship)?
- Depends upon the use we want to make of address information, and the semantics of the data:
  - If we have several addresses per employee, *address* must be an entity (since attributes cannot be set-valued).
  - If the structure (city, street, etc.) is important, e.g., we want to retrieve employees in a given city, *address* must be modeled as an entity (since attribute values are atomic).

**Entity vs. Attribute (Contd.)**

- Works_In2 does not allow an employee to work in a department for two or more periods.
**Entity vs. Relationship**

- First ER diagram OK if a manager gets a separate discretionary budget for each dept.
- What if a manager gets a discretionary budget that covers all managed depts?
  - Redundancy of \textit{dbudget}, which is stored for each dept managed by the manager.
  - Misleading: suggests \textit{dbudget} tied to managed dept.

**Binary vs. Ternary Relationships**

- If each policy is owned by just 1 employee:
  - Key constraint on Policies would mean policy can only cover 1 dependent!
- What are the additional constraints in the 2nd diagram?
Summary of Conceptual Design

❖ Conceptual design follows requirements analysis,
  – Yields a high-level description of data to be stored
❖ ER model popular for conceptual design
  Constructs are expressive, close to the way people think about their applications.
❖ Basic constructs: entities, relationships, and attributes (of entities and relationships).
❖ Some additional constructs: weak entities, ISA hierarchies, and aggregation.
❖ Note: There are many variations on ER model.

Summary of ER (Contd.)

❖ ER design is subjective. There are often many ways to model a given scenario! Analyzing alternatives can be tricky, especially for a large enterprise. Common choices include:
  Entity vs. attribute, entity vs. relationship, binary or n-ary relationship, whether or not to use ISA hierarchies, and whether or not to use aggregation.
❖ Ensuring good database design: resulting relational schema should be analyzed and refined further. More on this later...