The Entity-Relationship Model

Chapter 2

Why Is This Important?

- If you want to use a DBMS, you need to be able to represent your data in it.
- There are many ways to achieve this.
- We will discuss one approach that is traditionally seen as a good and successful one.
- No matter which approach is used, one can end up with a good or a bad database design.

- In a future lecture, we will discuss objective criteria for discovering and fixing bad design choices

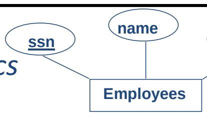
A Picture Is Worth A Thousand Words

- How do we represent data in a database?
- Relational model: store everything in tables (relations)
  - Rows correspond to "records"
  - Columns correspond to fields of these records
- A set of tables is surprisingly expressive.
- Challenge: how to choose the right set of tables
- Can use a graphical model to describe the data
  - Easier for other project participants to understand
- Map the graphical model automatically to a set of tables

Overview of Database Design

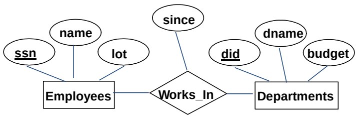
- Conceptual design: (ER Model is used at this stage.)
  - What are the entities and relationships in the enterprise?
  - What information about these entities and relationships should we store in the database?
  - What are the integrity constraints or business rules that hold?
  - A database `schema'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

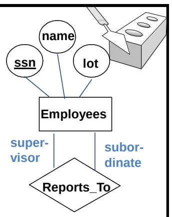
ER Model Basics



- 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 ISA hierarchies.)
  - Each entity set has a key.
  - Each attribute has a domain.

ER Model Basics (Contd.)





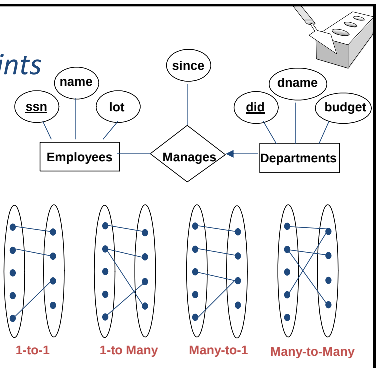
- Relationship: Association among two or more entities, e.g., Attishoo works in Pharmacy department.
  - Can have descriptive attributes that do not belong to any entity
  - Uniquely identified by participating entities (ssn, did)
- Relationship Set: Collection of similar relationships.
  - An n-ary relationship set R relates n entity sets E1,..., En
  - Each relationship in R involves entities e1∈E1,..., en∈En
    - Entity set could participate in different "roles" in same set.

Let's Try To Model Something

- Students in a dorm lend CDs to their friends in the same dorm.
- Each student lives in a dorm room.
- Your friends complain that they cannot remember who has which of their CDs and want you to design a database to keep track of this.
- In particular, the goal is to be able to find out who borrowed a certain CD from a certain person for more than a month, and where the borrower lives.
- What entities, relationships, and attributes are needed?

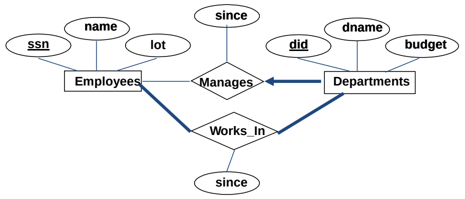
Key Constraints

- 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.



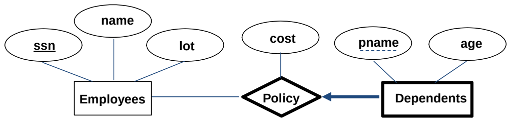
Participation Constraints

- 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).
    - Every Departments entity must appear in an instance of the Manages relationship.

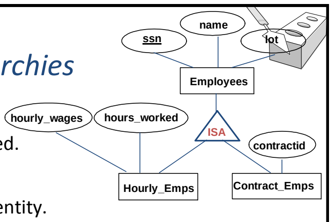


Weak Entities

- 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.



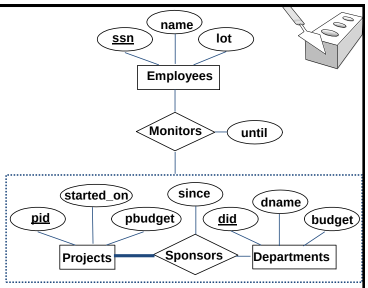
ISA (`is a') Hierarchies



- As in OOPLs, attributes are inherited.
- If we declare A ISA B, every A entity is also considered to be a B entity.
- 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 with another relationship.
  - 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

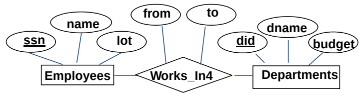
- Design choices:
  - Should a concept be modeled as an entity or an attribute?
  - Should a concept be modeled as an entity or a relationship?
  - Identifying relationships: Binary or ternary? Aggregation?
- Constraints in the ER Model:
  - A lot of data semantics can (and should) be captured.
  - But some constraints cannot be captured in ER diagrams.

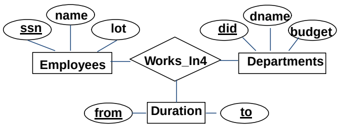
Entity vs. Attribute

- 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.)

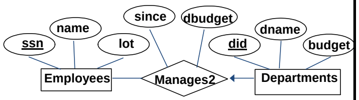
- Works_In4 does not allow an employee to work in a department for two or more periods.
- Similar to the problem of wanting to record several addresses for an employee: We want to record several values of the descriptive attributes for each instance of this relationship.
  - Accomplished by introducing new entity set, Duration.

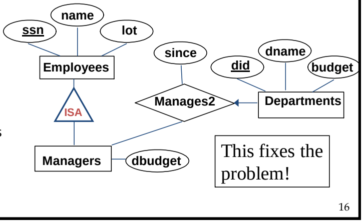




Entity vs. Relationship

- 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: dbudget stored for each dept managed by manager.
  - Misleading: Suggests dbudget associated with department-mgr combination.

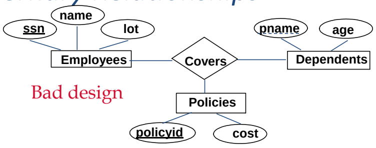




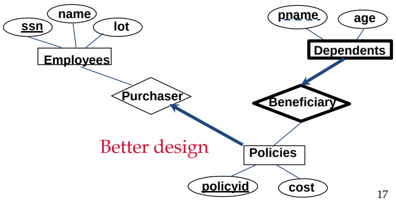
This fixes the problem!

Binary vs. Ternary Relationships

- Requirements:
  - Policy cannot be owned by more than 1 employee.
  - Every policy must be owned by some employee.
  - Dependents is a weak entity set; its key is pname together with policyID.
- What are the problems with the first diagram?



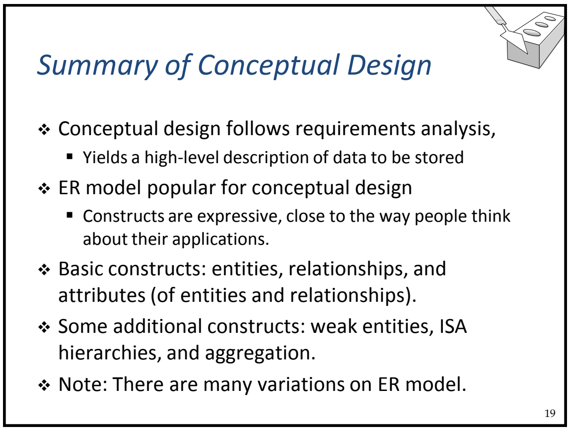
Bad design



Better design

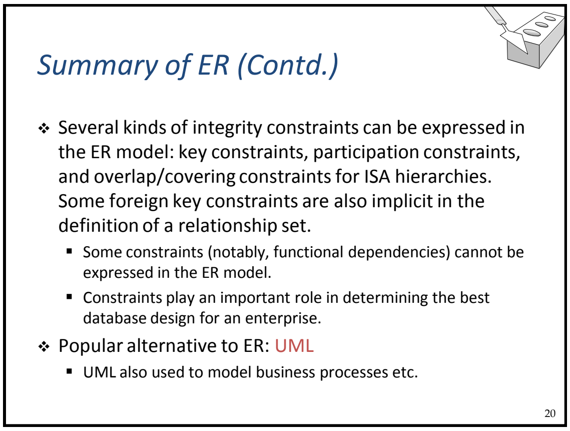
Binary vs. Ternary Relationships (Contd.)

- Previous example illustrated a case when two binary relationships were better than one ternary relationship.
- An example in the other direction: a ternary relationship Contracts relates entity sets Parts, Departments and Suppliers, and has descriptive attribute qty. No combination of binary relationships is an adequate substitute:
  - S “can-supply” P, D “needs” P, and D “deals-with” S does not imply that D has agreed to buy P from S.
  - How do we record qty?



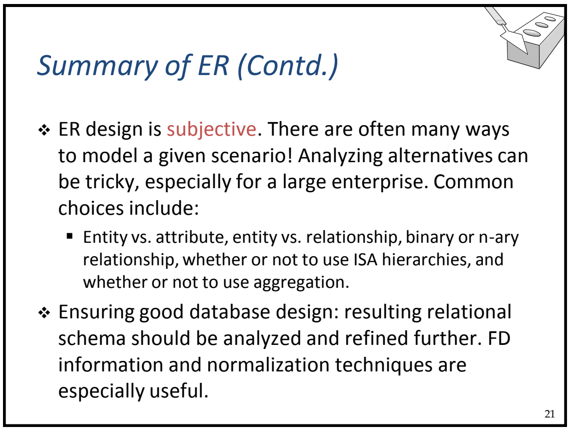
Summary of Conceptual Design

- 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.)

- Several kinds of integrity constraints can be expressed in the ER model: key constraints, participation constraints, and overlap/covering constraints for ISA hierarchies. Some foreign key constraints are also implicit in the definition of a relationship set.
  - Some constraints (notably, functional dependencies) cannot be expressed in the ER model.
  - Constraints play an important role in determining the best database design for an enterprise.
- Popular alternative to ER: UML
  - UML also used to model business processes etc.



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. FD information and normalization techniques are especially useful.