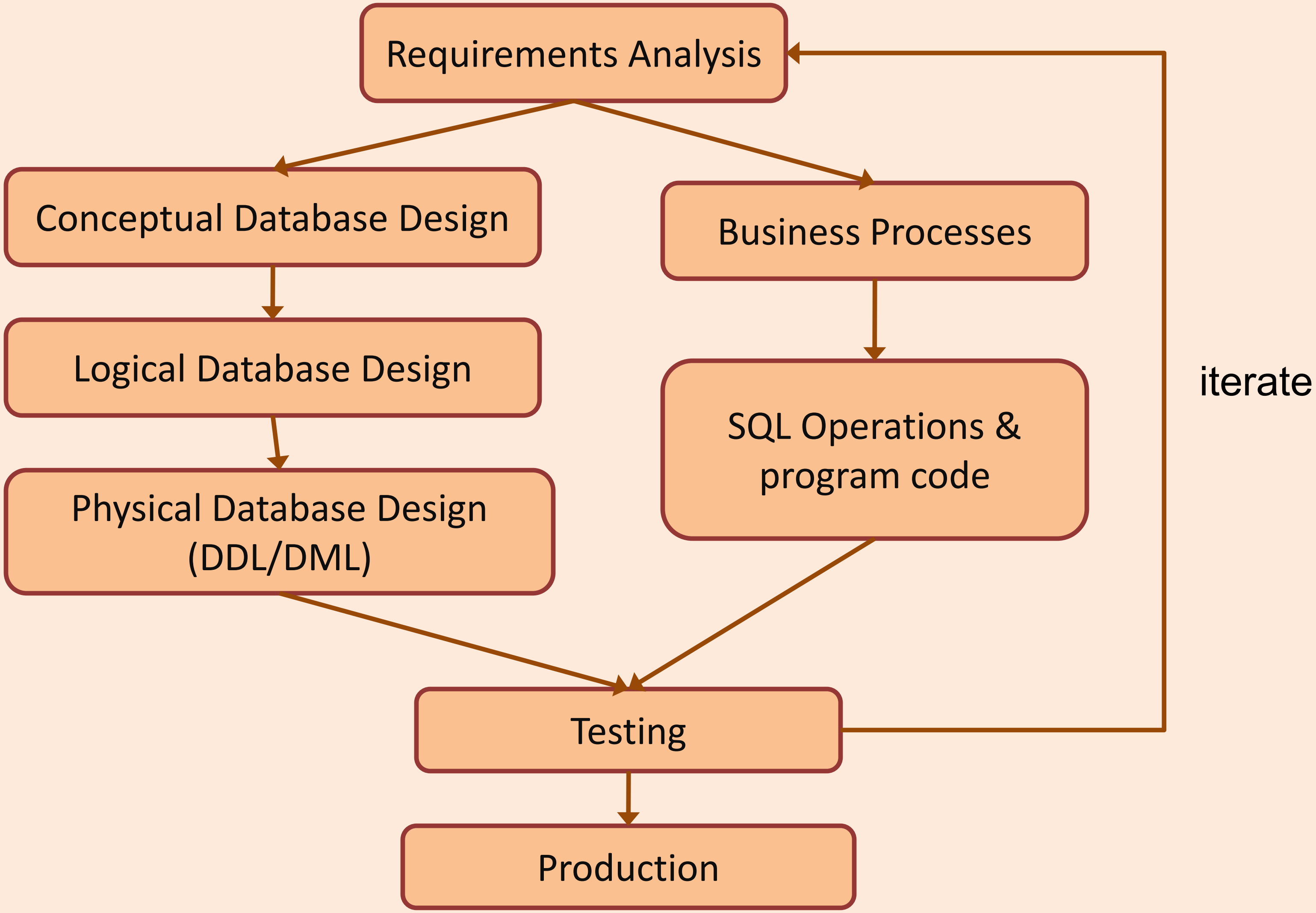
ICS 321 Spring 2011
High Level Database Models

Asst. Prof. Lipyeow Lim

Information & Computer Science Department

University of Hawaii at Manoa

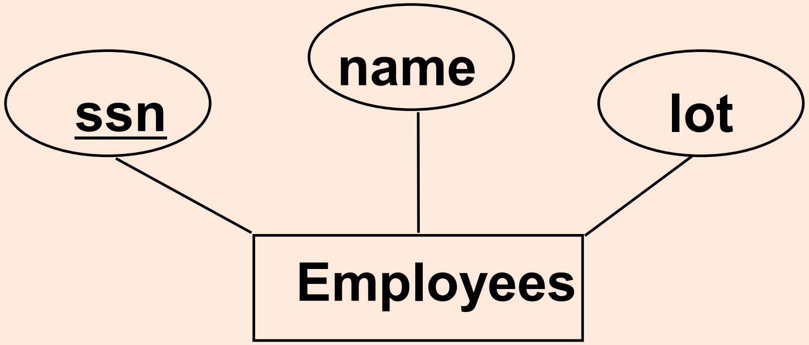
Database Design & Deployment


Requirements Analysis
Conceptual Database Design
Business Processes
Logical Database Design
SQL Operations & program code
iterate
Physical Database Design (DDL/DML)
Testing
Production

Overview Database Design

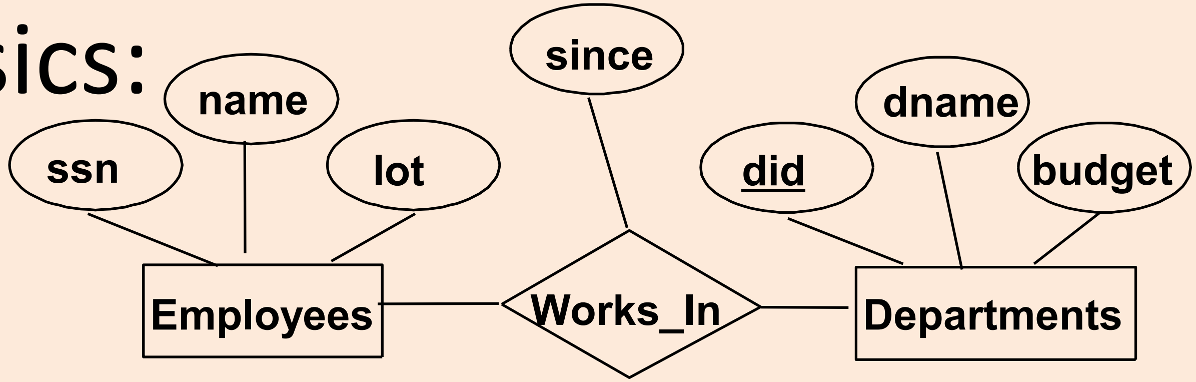
- Conceptual Design
  - Use entity-relationship (aka ER) model represented pictorially as ER diagrams
  - Map ER model to relational schema
- Questions to ask yourself
  - What are the entities and relationships in the application?
  - What information about these entities and relationships should we store in the database?
  - What are the integrity constraints or business rules that hold?

ER Model Basics: Entities



- *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, anyway!)
  - Each entity set has a *key*.
  - Each attribute has a *domain*.

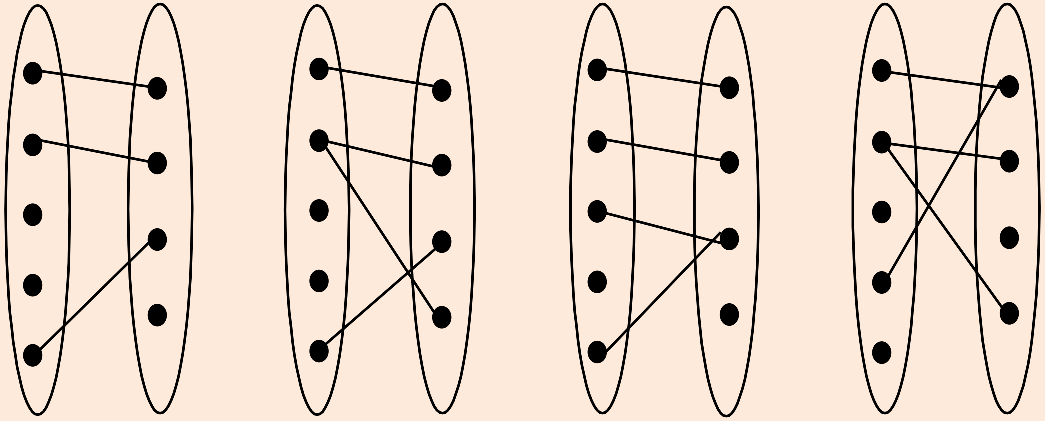
ER Model Basics: Relationships



- *Relationship*: Association among two or more entities.
- *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
  - Same entity set could participate in different relationship sets, or in different “roles” in same set.

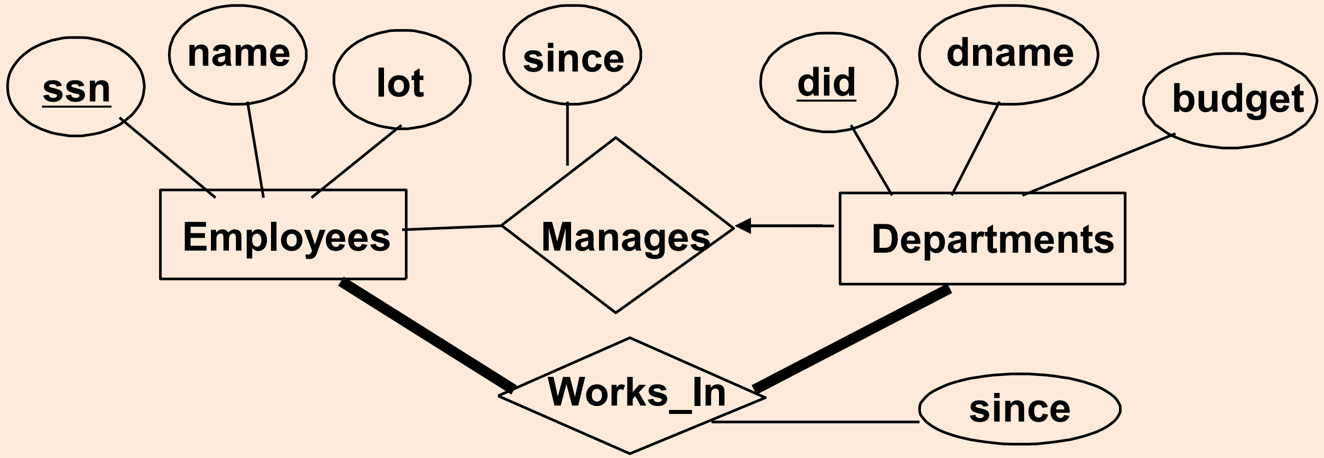
Cardinality Ratios of Relationships

- Consider binary relationships, i.e., between two entity sets
- Alternate notation: 1:1, 1:M, M:1, M:N



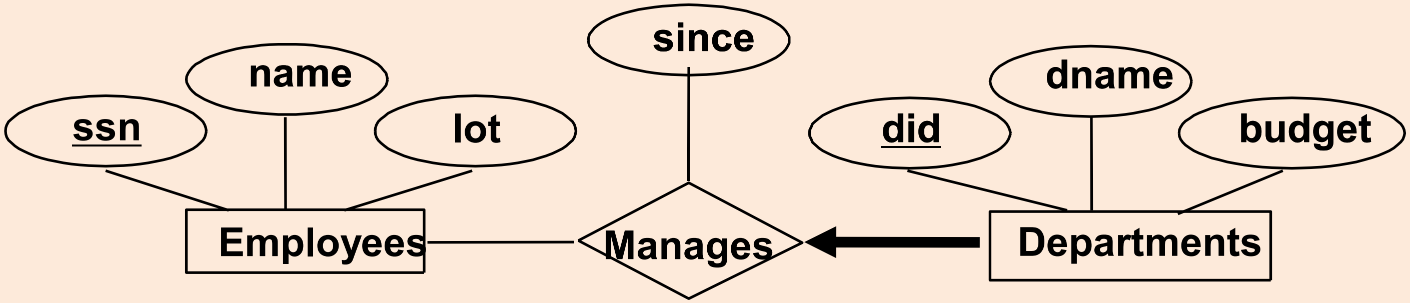
1-to-1 **1-to Many** **Many-to-1** **Many-to-Many**

Key Constraints



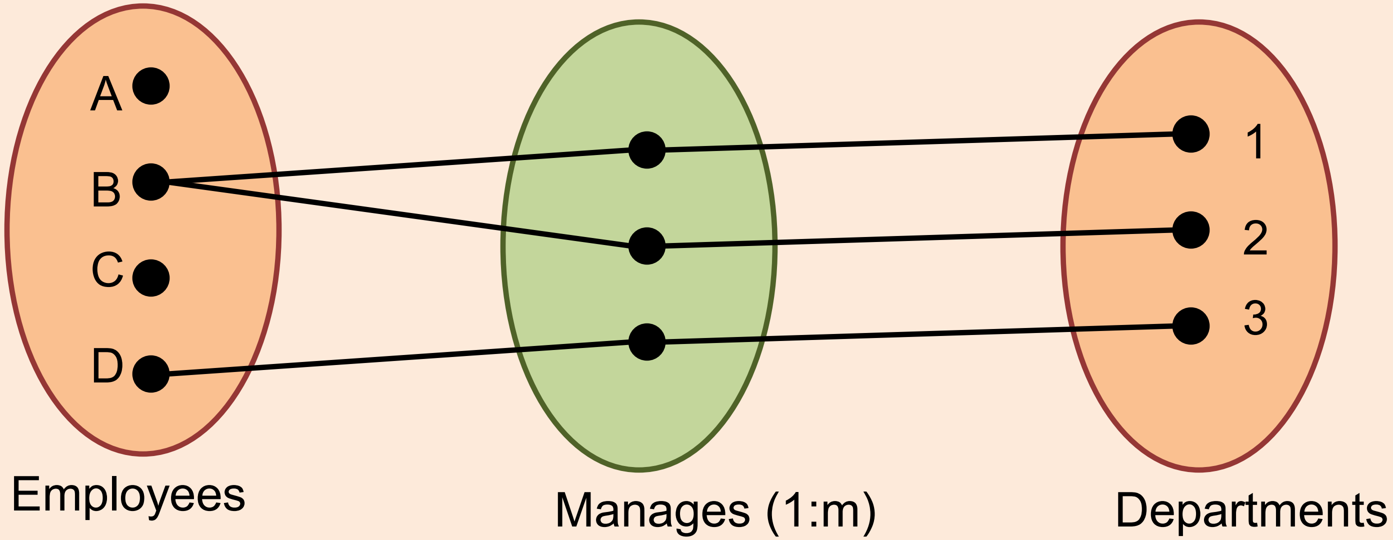
- Consider Works_In: An employee can work in many depts; a dept can have many employees : m-to-m
- Consider Manages: each dept has at most one manager
- Dept has a *key constraint* on Manages: each instance of dept appears in at most one instance of manages
- Denoted by an arrow: given a dept entity we can uniquely identify the manages relationship in which it appears

Participation constraints



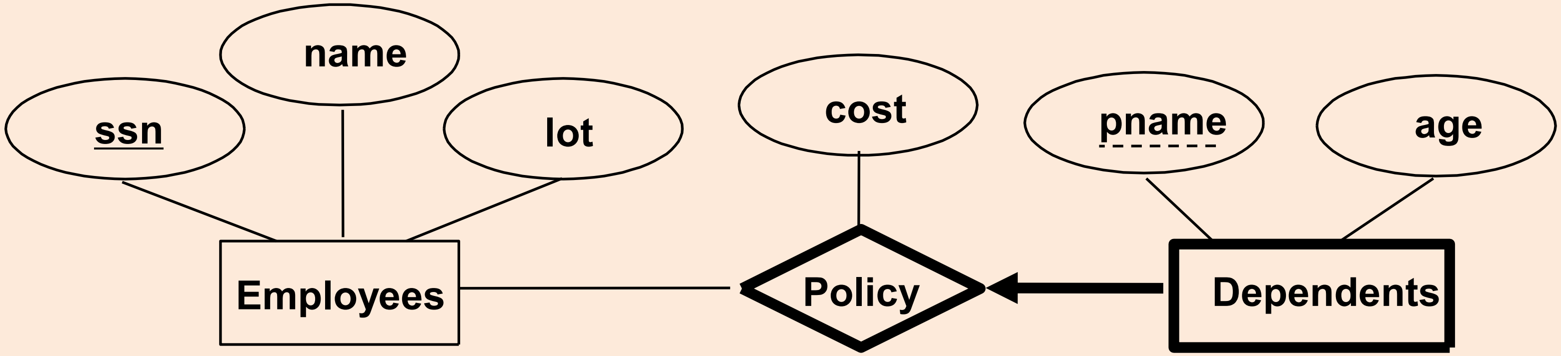
- Does every dept have a manager?
- If so, this is a *participation constraint*: the participation of dept in Manages is said to be *total* (vs. *partial*). Denoted by thick/double line
- Meaning that every Dept entity must appear in an instance of the Manages relationship

Set Theoretic Formulation



- Partial Partiticipation: Not all members of the Employees entity set take part in the manages relations
- Total Partiticipation: All members of the Dept entity set take part in the manages relationship
- Dept has a key constraint on Manages: each member of the dept entity set takes part in at most one member of the manages relationship set

Weak Entities

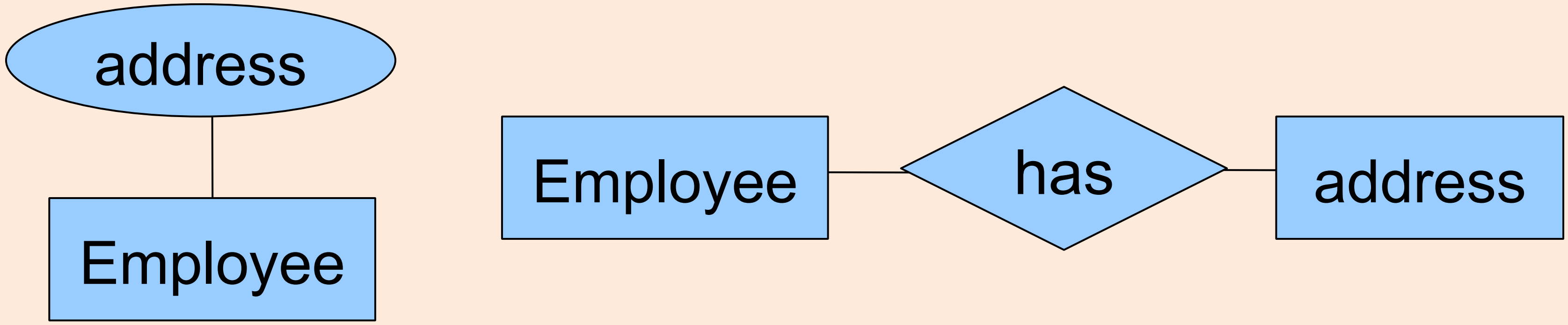


- A *weak entity* can be identified uniquely only by considering the primary key of another (*owner*) entity.
- Owner entity set and weak entity set must participate in a one-to-many relationship set (one owner, many weak entities).
- Weak entity set must have total participation in this *identifying* relationship set.
- Denoted by a box with double or thick lines

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?
- How much semantics to capture in the form of constraints ?

Entity vs. Attribute



- Depends upon how we want to use the 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).