Dr Renata Borovica Gajic

David Eccles

THE UNIVERSITY OF MELBOURNE

# INFO 90002 Database Systems & Information Modelling

Lecture 3

Introduction to Data Modelling

# News

- Time to form a team!
  - Teams of 4
  - Register your team under the Assignment Groups
  - Complete your Group Agreement
  - Arrange a time for your kick-off meeting with a tutor or lecturer (Check LMS announcements for timeslots)
- Password change on INFO90002db.eng.unimelb.edu.au
- Lecturer Consultation Times
  - Monday 1400H (2pm) AEST
  - Wednesday 1000H (10am) AEST Lecture Recap, Q&A
  - Thursday 1100H (11am) AEST Ask me anything about the subject assignments, tutorials, lectures
  - **NOT** RECORDED – Turn up!

THE UNIVERSITY OF
MELBOURNE

# Lecture 2: Verb Noun Analysis

Solution

# Noun-verb analysis solution

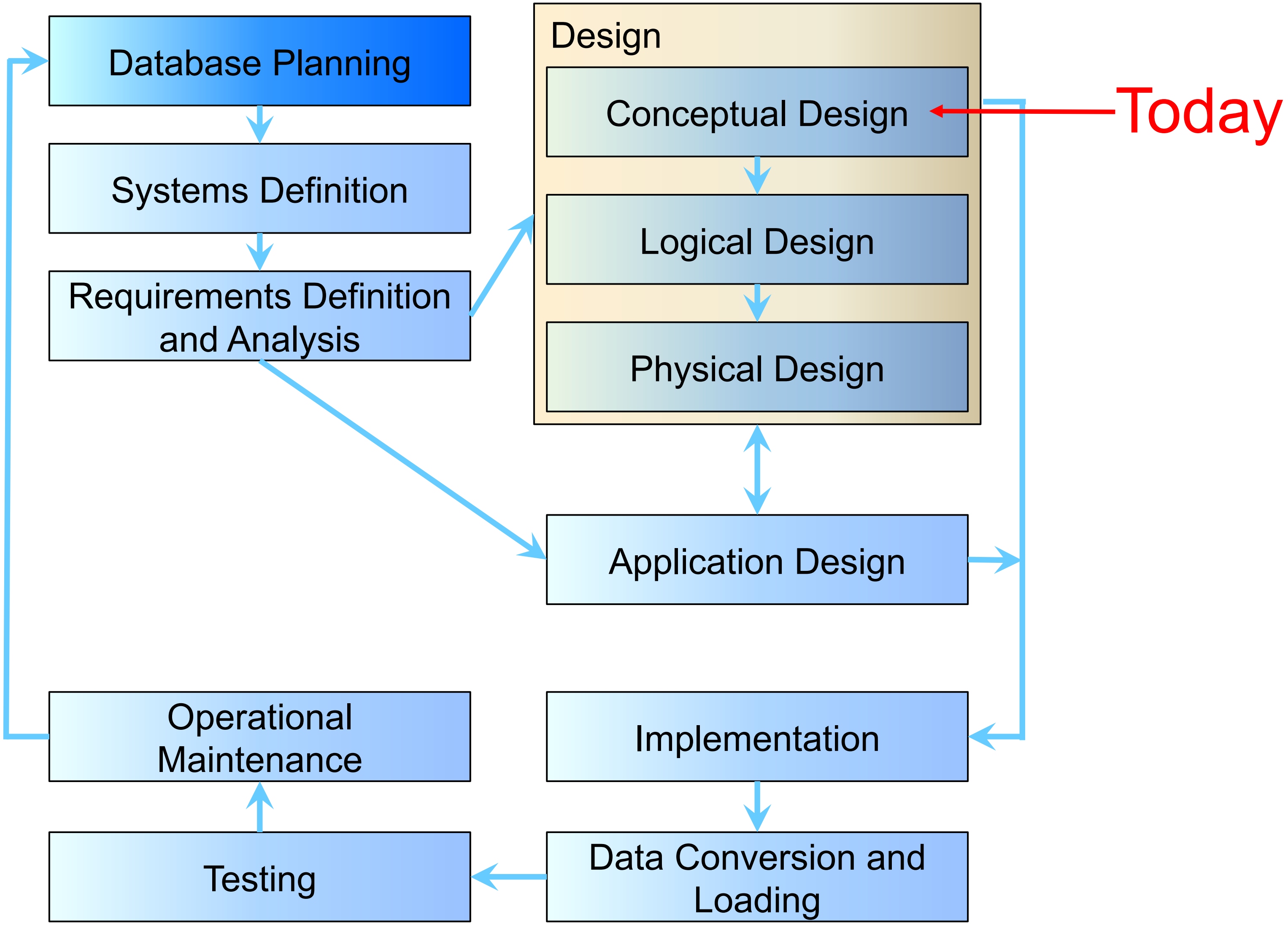
- An investment bank has a number of branches. Within each branch a number of departments operate and are structured in a hierarchical manner. The bank employs around 3000 staff who are assigned to work in the various departments across the branches. There are essentially three types of special employees where extra details required by the system. There are dealers who carry out investments who have limits imposed upon them for how much they can spend. There are IT compliance managers who's Basel2 role is required to be stored and there are HR managers that need have their assessment number recorded (along with other details not specified here).
- We need a database to record staff details including which department and branch they are assigned...

THE UNIVERSITY OF
MELBOURNE

Dr Renata Borovica Gajic
David Eccles

# 3. Introduction to ER Modelling

# Database Development Lifecycle: Review



Database Planning

Systems Definition

Requirements Definition and Analysis

Design

Conceptual Design

Logical Design

Physical Design

Today

Application Design

Operational Maintenance

Implementation

Testing

Data Conversion and Loading

# The Entity-Relationship Model

- Basic ER modeling concepts
- Constraints
- Conceptual Design

*Readings: Chapter 2, Ramakrishnan & Gehrke, Database Systems*

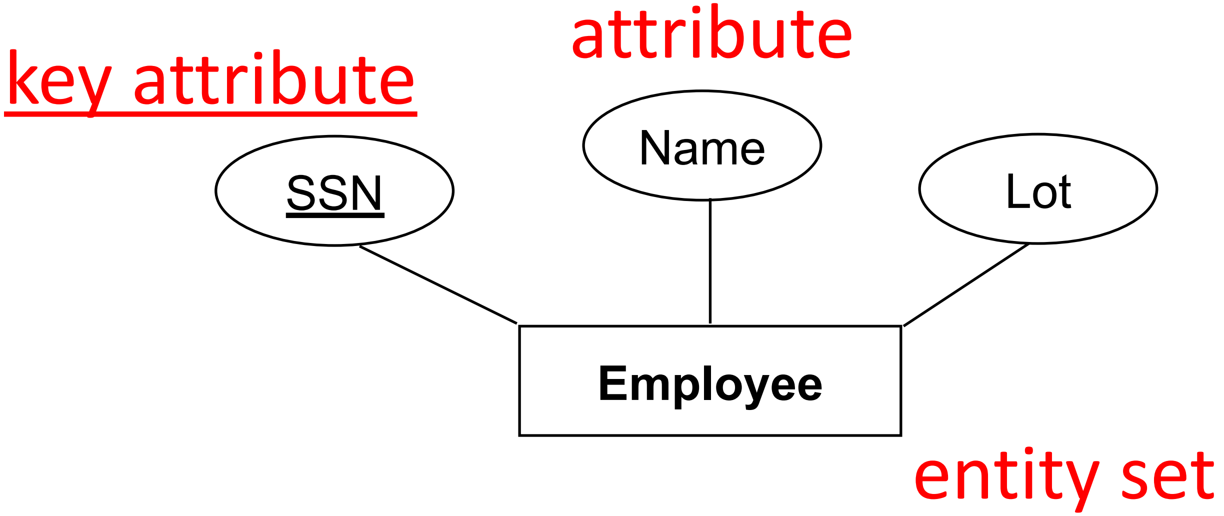
# Conceptual Design: Objectives

- 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* that hold?

# ER Model: Entity & its attributes

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

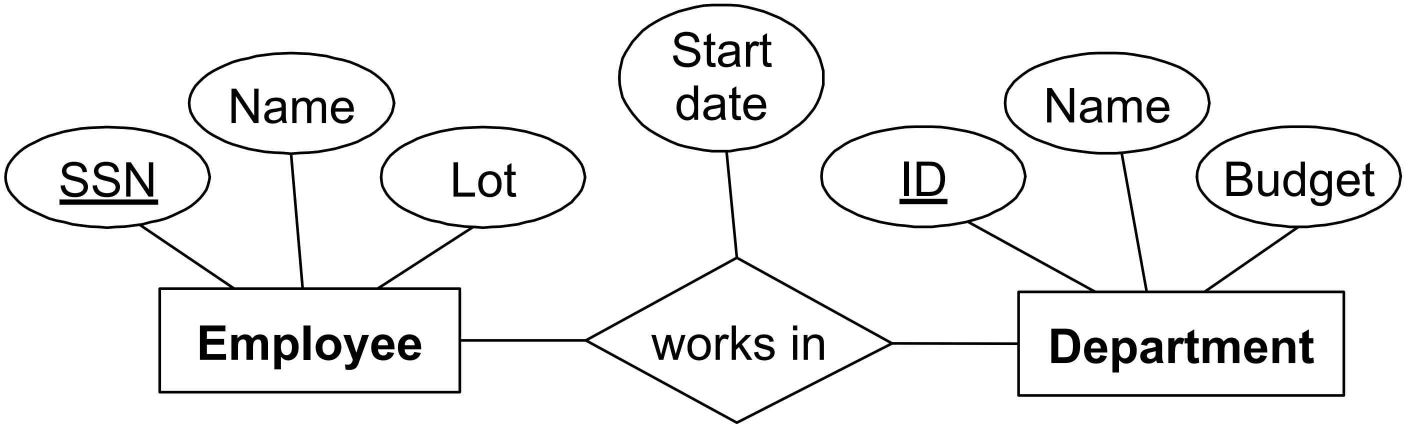
- ***Entity Set***: A collection of entities of the same type (e.g. *all employees*)
  - All entities in an entity set have the same set of attributes
  - Each entity has a *key (underlined)*



# ER Model: Relationship

- ***Relationship***: Association among two or more entities. Relationships can have their own attributes.
  - Example: Fred *works in* the Pharmacy department.

- ***Relationship Set***: Collection of relationships of the same type.
  - Example: Employees *work in* departments.

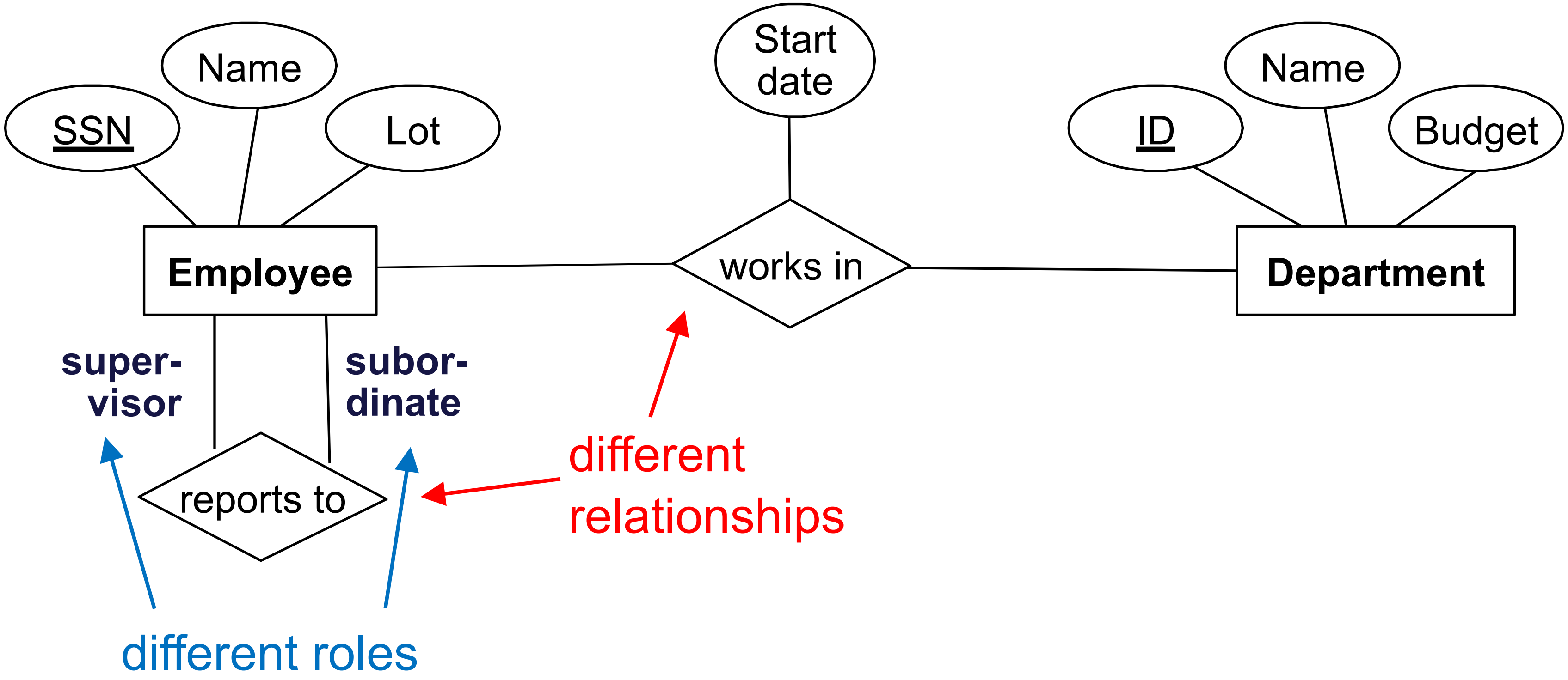


relationship set
(with an attribute)

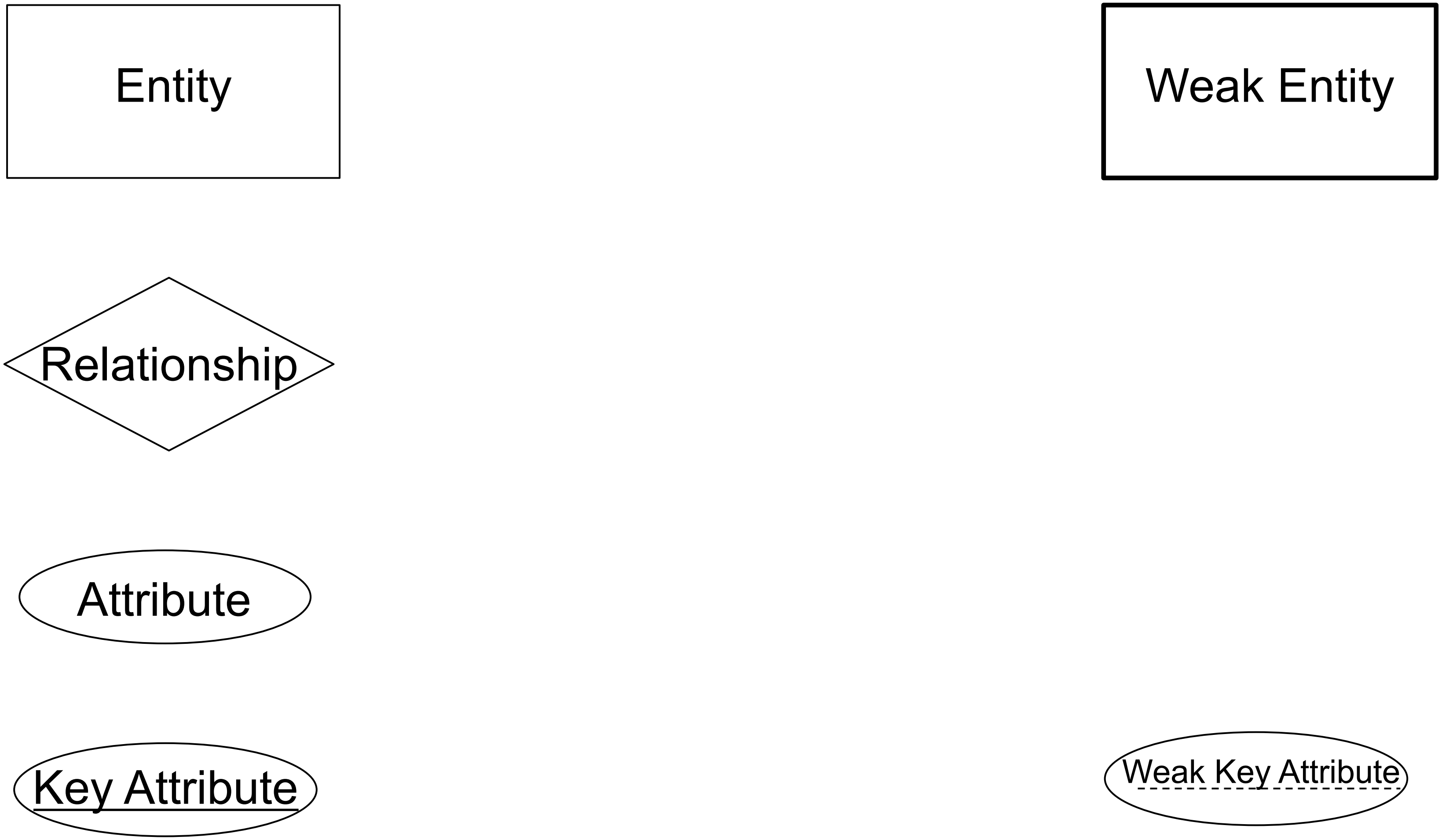
# ER Model: Relationship roles

Same entity set can participate in:

- *different* relationship sets, or even
- *different “roles”* in the same set



# Chen notation key shapes



Entity

Weak Entity

Relationship

Attribute

Key Attribute

Weak Key Attribute

# The Entity-Relationship Model

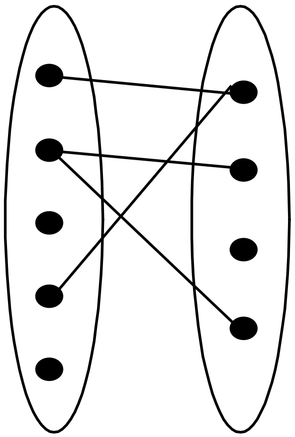
- Basic ER modeling concepts
- Constraints
- Conceptual Design

*Readings: Chapter 2, Ramakrishnan & Gehrke, Database Systems*

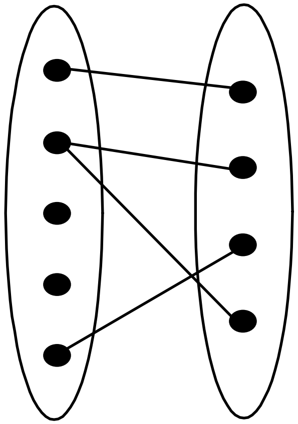
# Key Constraints: Types

***Key constraints*** determine the number of objects taking part in the relationship set (how many from each side)

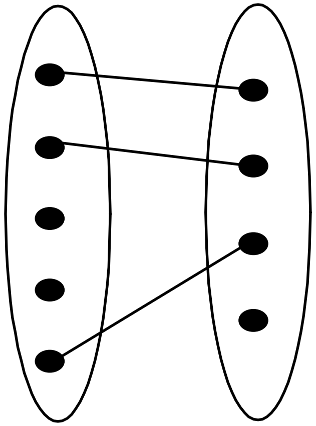
**Types of key constraints:**



**Many-to-Many**



**One-to-Many**



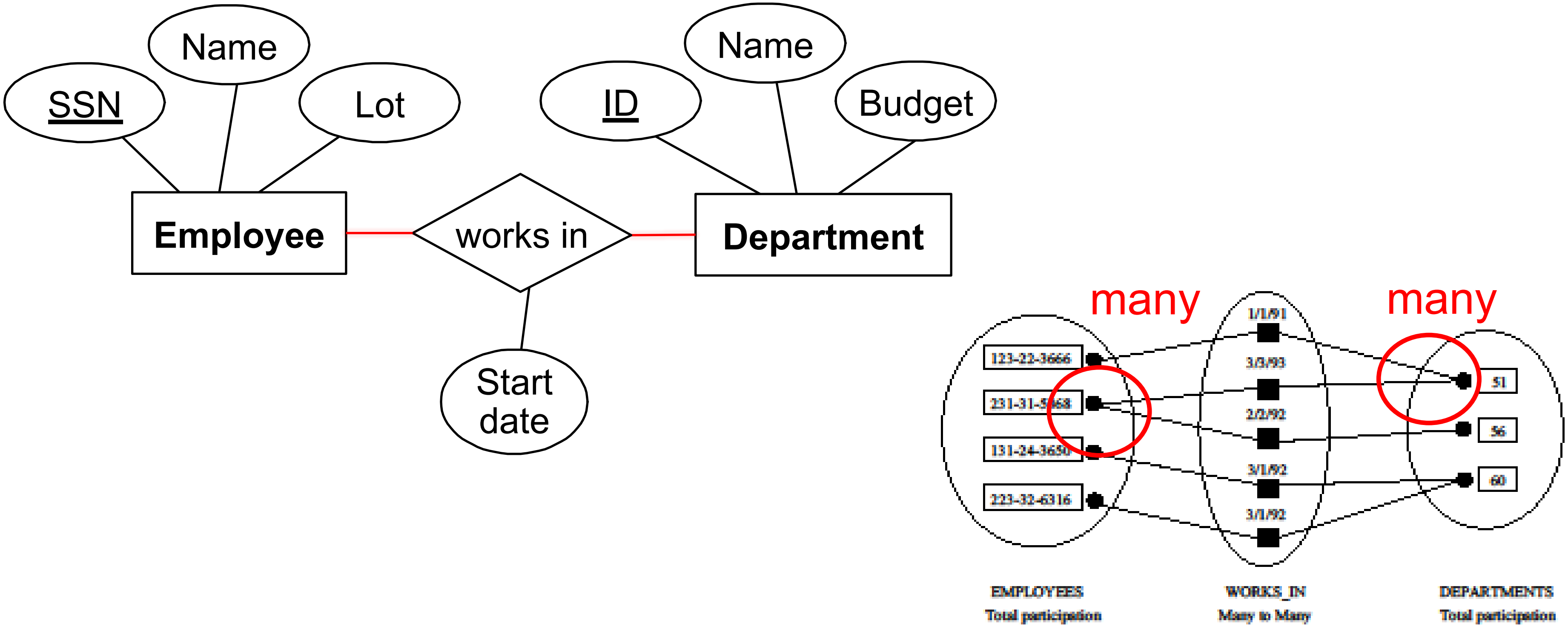
**One-to-One**

# Key Constraints: Many-to-Many

**Example**:

*An employee can work in many departments; a department can have many employees.*

Many is represented by a *line*.



Employee works in many departments

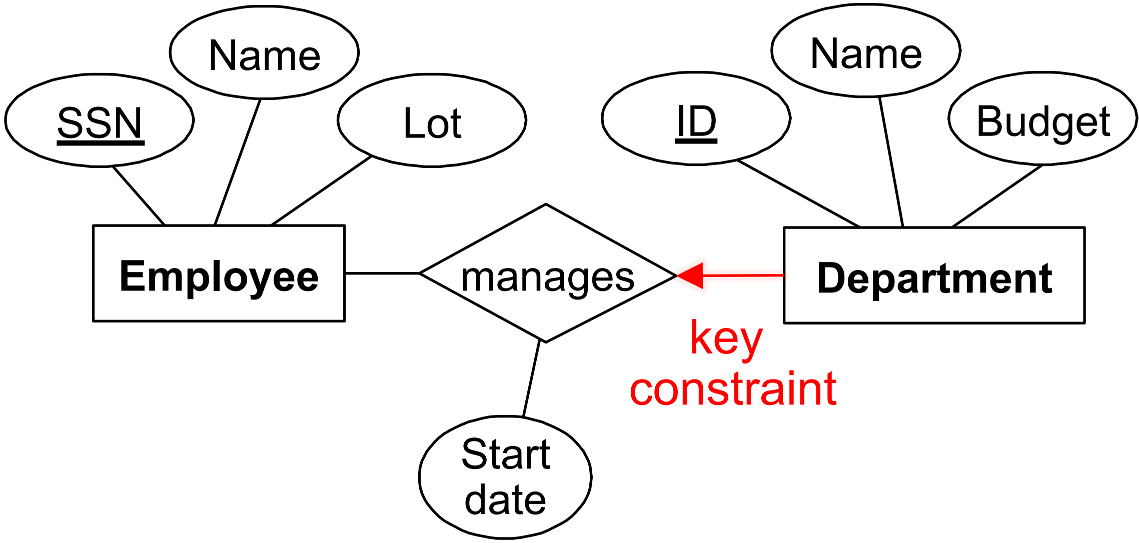
# Key Constraints: One-to-Many

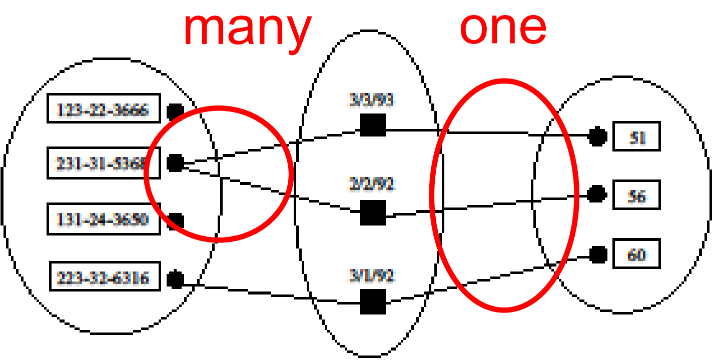
***One-to-many*** constrains one entity set to have a *single* entity per a relationship. An entity of that set can never participate in two relationships of the same relationship set. This is called a **key constraint** and is represented by an *arrow*.

**Example**:

*Each department has at most one manager.*

This is the key constraint on Manages.



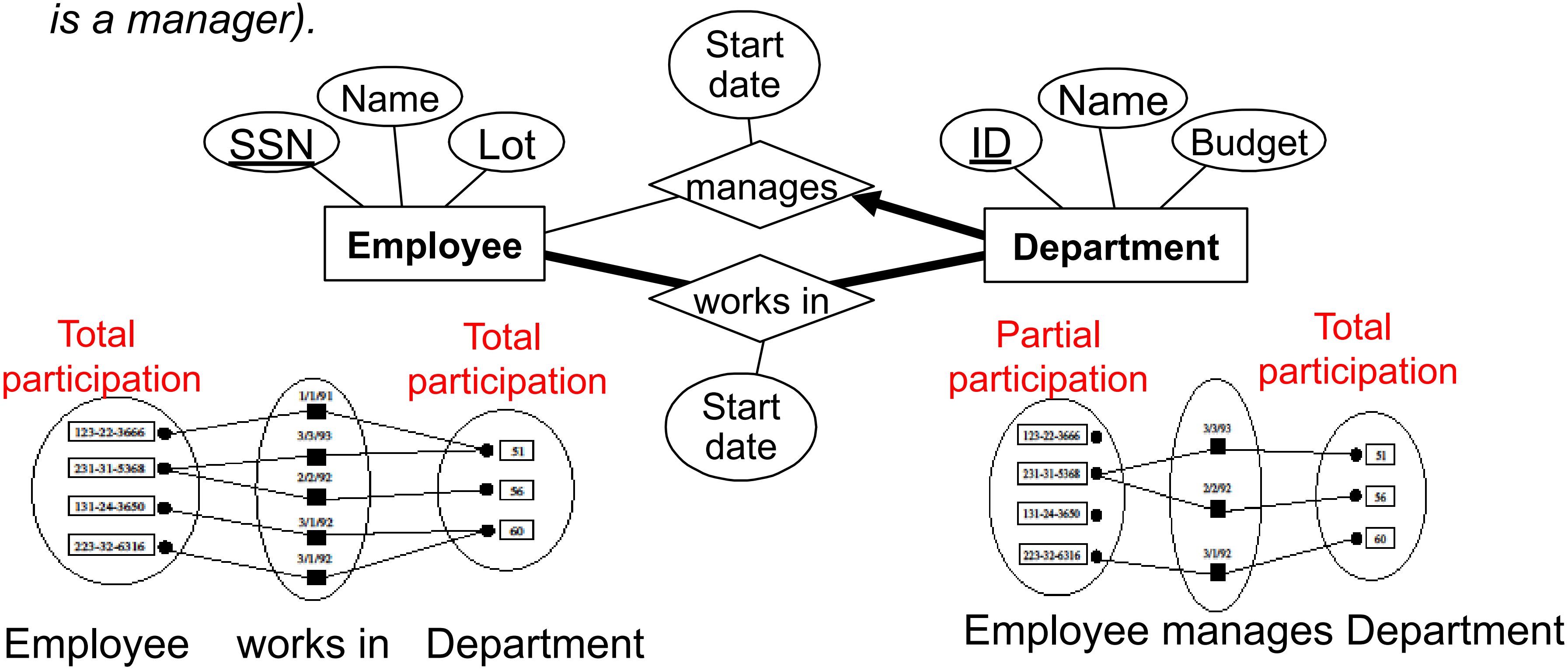


Department has at most one manager

# Participation Constraints

***Participation constraint*** explores whether all entities of one entity set take part in a relationship. If yes this is a **total** participation, otherwise it is **partial**. Total participation says that each entity takes part in "**at least one**" relationship, and is represented by a bold line.

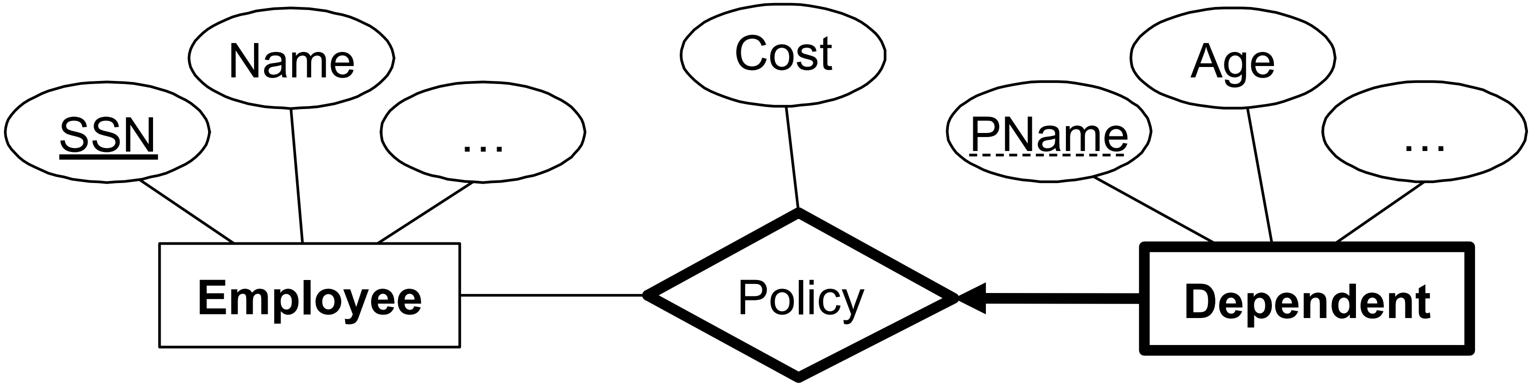
**Example**: *Every employee must work in a department. Each department has at least one employee. Each department has to have a manager (but not everyone is a manager).*



# Weak Entities

A ***weak entity*** can be identified uniquely only by considering (the primary key of) another (*owner*) entity. They are represented as a “bold” rectangle.

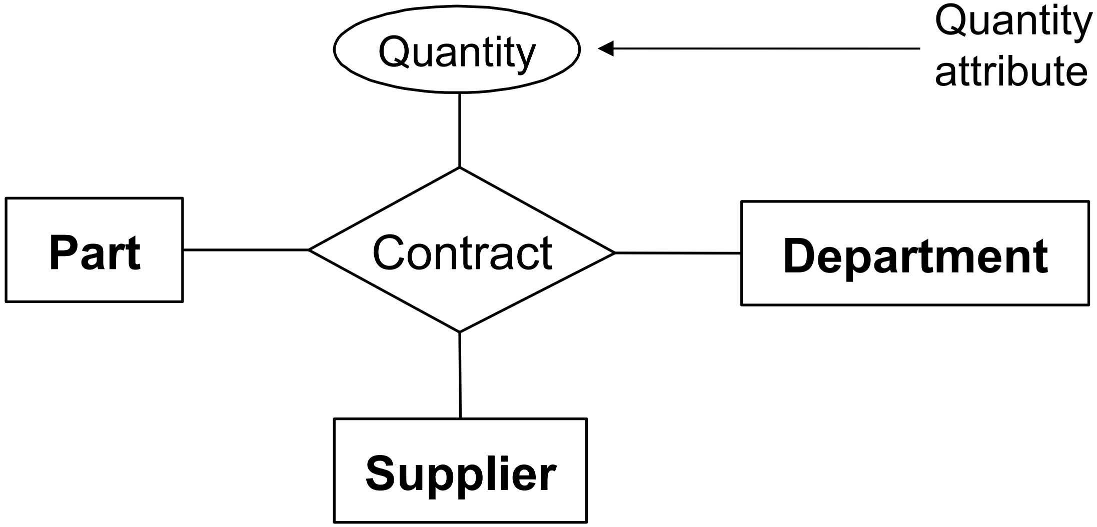
- Owner entity set and weak entity set must participate in a *one-to-many* relationship set (one owner, many weak entities)
- Weak entity set must have total participation in this relationship set. Such relationship is called *identifying* and is represented as “bold”.



Weak entities have only a “partial key” (dashed underline) and they are identified uniquely only when considering the primary key of the owner entity

# Ternary Relationships

In general, we can have **n**-ary relationships, and relationships can have attributes



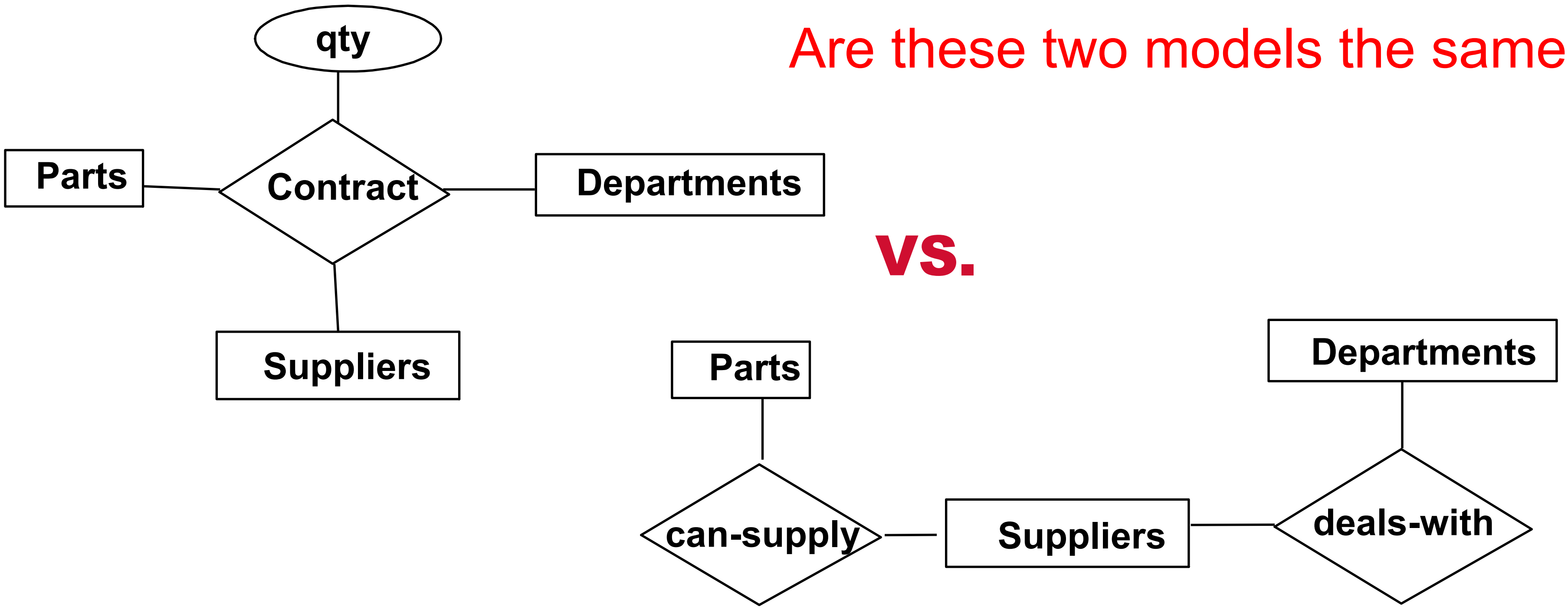
Quantity is an attribute of Contract

This is a ternary relationship with one relationship attribute

*“Departments will use one or more suppliers to obtain one or more parts for producing the widget”*

# Ternary vs. Binary Relationships

Are these two models the same?



## Second model:

- S “can-supply” P, D “needs” P, and D “deals-with” S does not imply that D has agreed to buy P from S. Not the same!
- How do we record *qty*?

# The Entity-Relationship Model

- Basic ER modeling concepts
- Constraints
- Conceptual Design

*Readings: Chapter 2, Ramakrishnan & Gehrke, Database Systems*

# Conceptual Design Using the ER Model

- **Design choices:**
  - Should a concept be modelled as an entity or an attribute?
  - Should a concept be modelled as an entity or a relationship?
  - Should we model relationships as binary, ternary, n-ary?

- **Constraints in the ER Model:**
  - A lot of data semantics can (and should) be captured

# Entity vs. Attribute

**Example**:

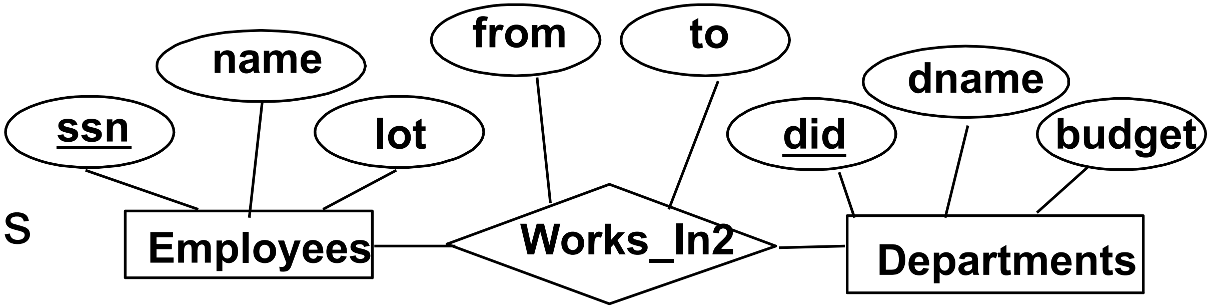
Should "*address*" be an attribute of Employees or an entity (related to Employees)?

**Answer**:

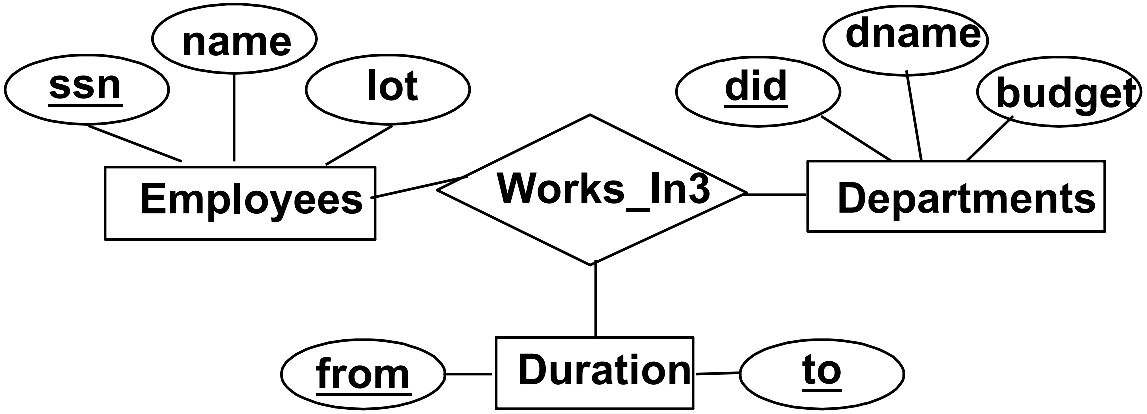
- *Depends* upon how we want to use address information, and the semantics of the data:
  - If we have several addresses per employee, *address* must be an entity
  - If the structure (city, street, etc.) is important, *address* should be modeled as an entity

# Entity vs. Attribute (Cont.)

- Works_In2 does not allow an employee to work in a department for two or more periods



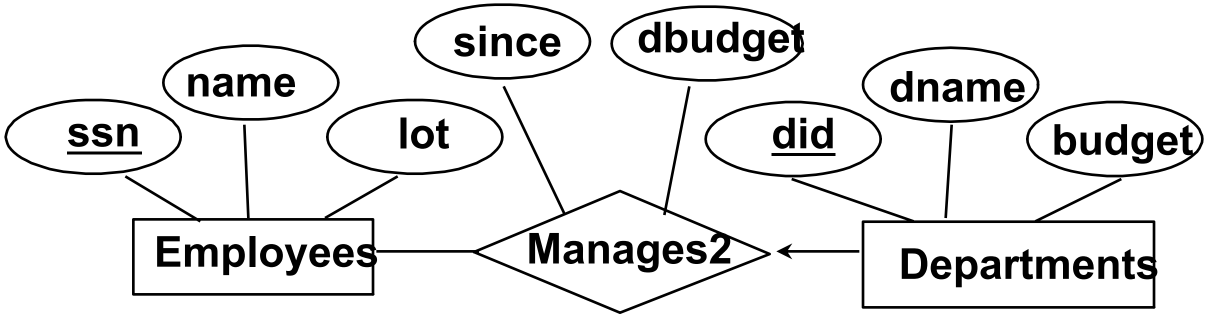
- Similar to the problem of wanting to record several addresses for an employee: we want to record *several values of the descriptive attributes for each instance of this relationship*

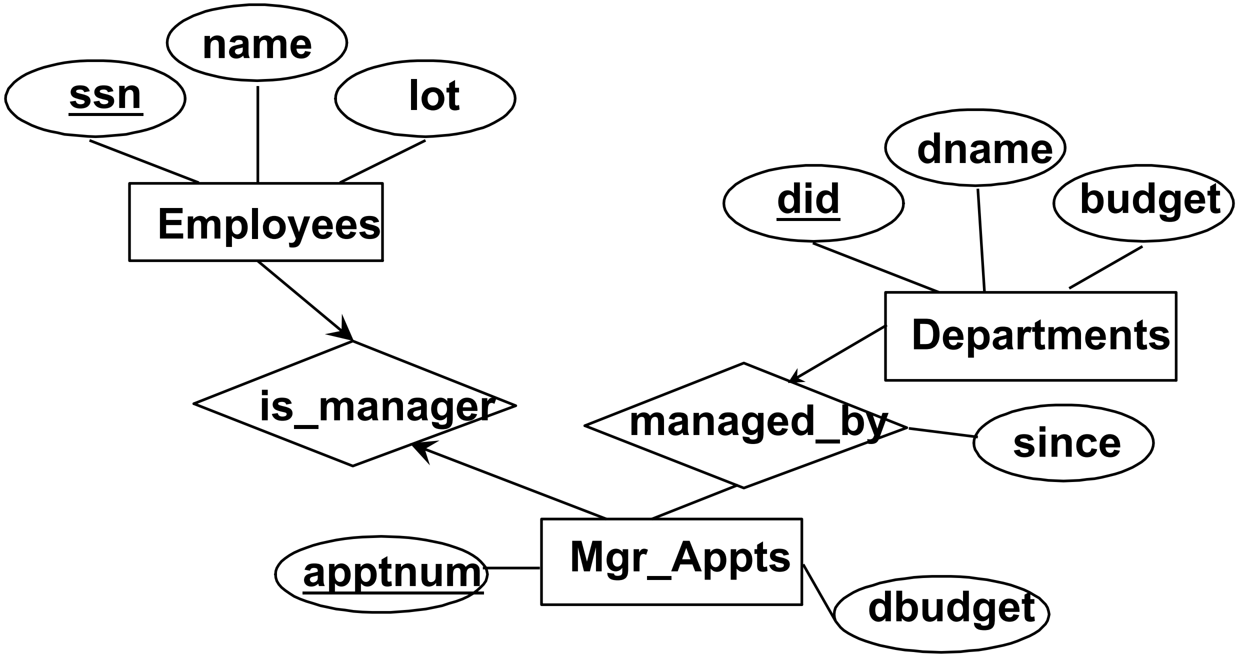


# Entity vs. Relationship

- OK as long as a manager gets a separate discretionary budget (*dbudget*) for each department

- What if manage's *dbudget* covers *all* managed departments? (can repeat value, but such redundancy is problematic)





# Notes on the ER design

- 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.

- There is no standard notation (we will cover two notations, today we learned **Chen's** notation)

# 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
  - Originally proposed by Peter Chen, 1976

Note: there are many variations on ER model

- Basic constructs: *entities*, *relationships*, and *attributes* (of entities and relationships)
- Some additional constructs: *weak entities*

# HOME WORK Chen Conceptual Model

**University database schema:**

- *Entities:* Subject, Professors
- Each subject has id, title, time

- Make up suitable attributes for professors

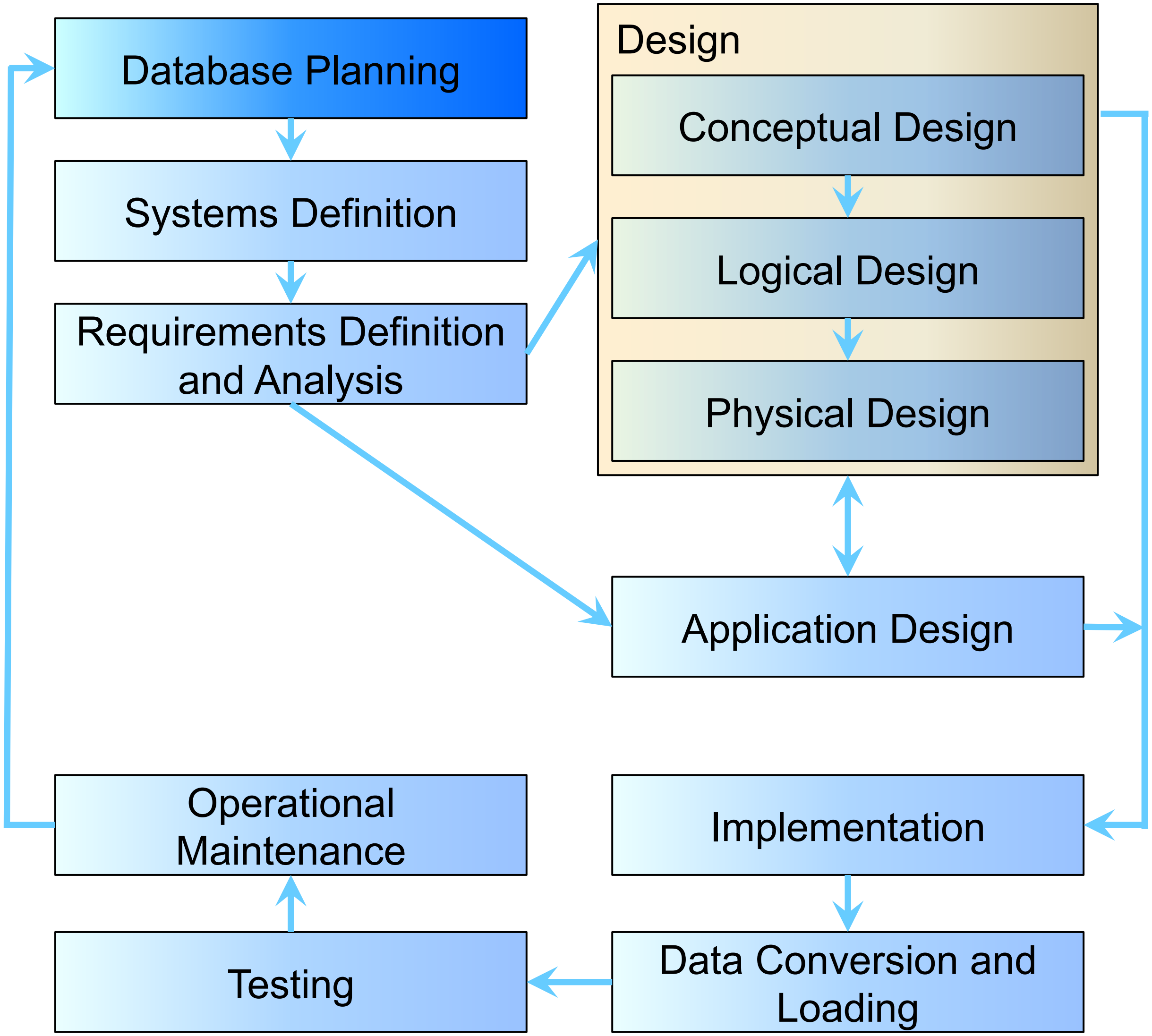
# Your turn…

1. Every professor must teach some subject.

2. Every professor teaches exactly one subject (no more, no less).

3. Every professor teaches exactly one subject (no more, no less), and every course must be taught by some professor.

# Database Development Lifecycle: Review



Database Planning

Systems Definition

Requirements Definition and Analysis

Design

Conceptual Design

Logical Design

Physical Design

Application Design

Implementation

Data Conversion and Loading

Testing

Operational Maintenance

Today

Next time

# What's Examinable?*

- Need to be able to draw conceptual diagrams on your own
  - Given a problem, *determine entities, attributes, relationships*
  - What is key constraint and participation constraint, weak entity?
  - Determine constraints for the given entities & their relationships
  - You must use CHEN notation for conceptual models

* All material is examinable – these are the suggested key skills you would need to demonstrate in an exam scenario

# Next Lecture

- Logical and Physical Modelling
  - From conceptual through to physical
  - Introducing the **relational model**

# Next 4 weeks

- More detailed understanding of database design
  - Conceptual design
  - Logical design
  - Physical design
- SQL
  - Overview
  - DML JOINS FUNCTIONS
  - GROUP BY HAVING
- Relational Algebra