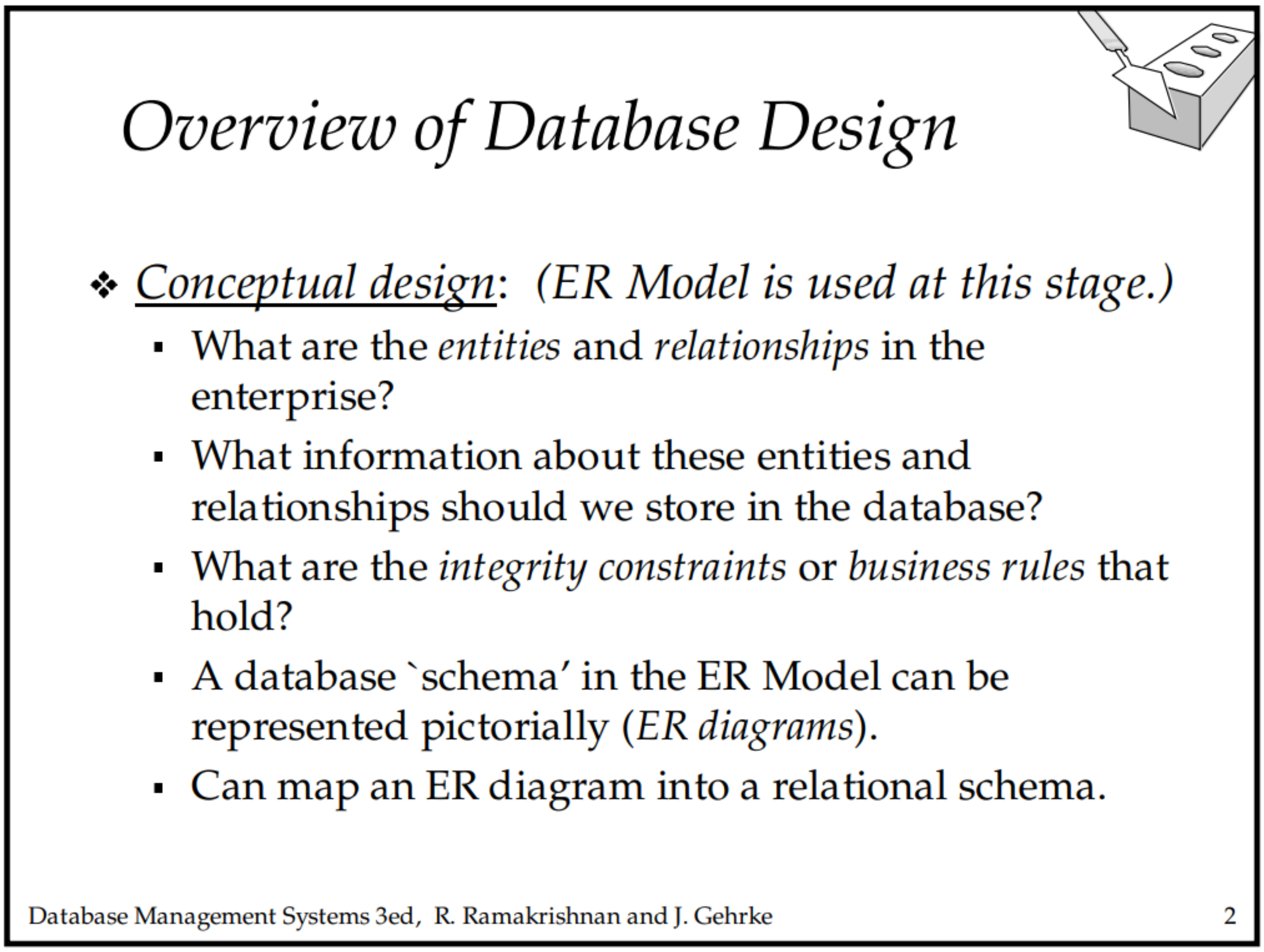
CH2: The Entity-Relationship Model

- CH2: The Entity-Relationship Model
  - What are the steps in designing a database?
  - Why is the ER model used to create an initial design?
  - What are the main concepts in the ER model?
  - What are guidelines for using the ER model effectively?
  - How does database design fit within the overall design framework for complex software within large enterprises?
  - What is UML and how is it related to the ER model?
  - CASE STUDY: THE INTERNET SHOP
    - Requirements Analysis
    - Conceptual Design

What are the steps in designing a database?

The database design process can be divided into six steps. The ER model is most relevant to the first three steps.

1. ***Requirements Analysis:*** The very first step in designing a database application is to understand what data is to be stored in the database, what applications must be built on top of it, and what operations are most frequent and subject to performance requirements.
2. ***Conceptual Database Design:*** The information gathered in the requirements analysis step is used to develop a high-level description of the data to be stored in the database, along with the constraints known to hold over this data.



Overview of Database Design

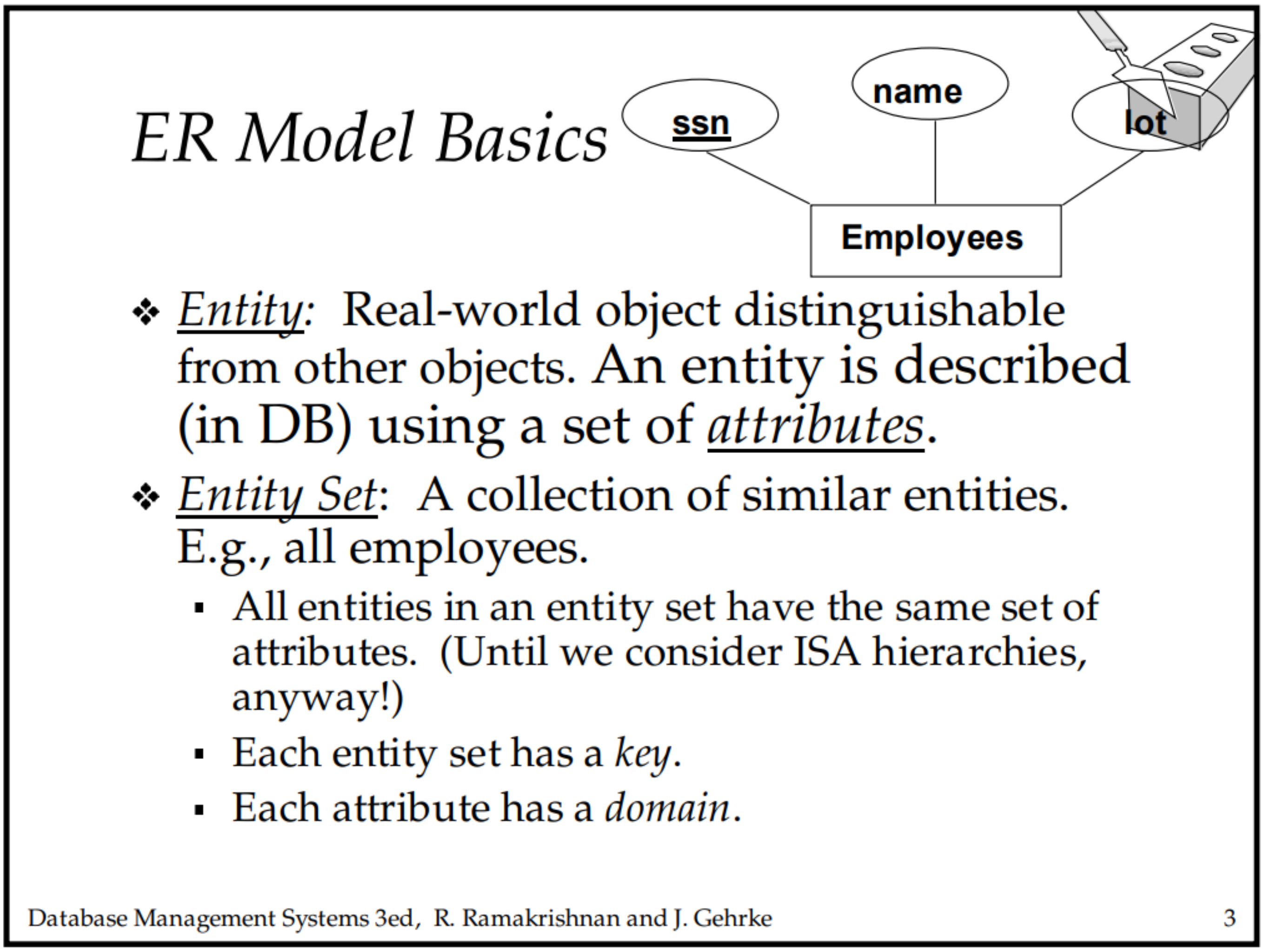
- Conceptual design: (ER Model is used at this stage.)
  - What are the *entities* and *relationships* in the enterprise?
  - What information about these entities and relationships should we store in the database?
  - What are the *integrity constraints* or *business rules* that hold?
  - A database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

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 2

3. ***Logical Database Design:*** We must choose a DBMS to implement our database design, and convert the conceptual database design into a database schema in the data model of the chosen DBMS. We will consider only relational DBMSs, and therefore, the task in the logical design step is to convert an ER schema into a relational database schema.
4. **Schema Refinement:** The fourth step in database design is to analyze the collection of relations in our relational database schema to identify potential problems, and to refine it.
5. **Physical Database Design:** In this step, we consider typical expected workloads that our database must support and further refine the database design to ensure that it meets desired performance criteria.
6. **Application and Security Design:** Any software project that involves a DBMS must consider aspects of the application that go beyond the database itself.

Why is the ER model used to create an initial design?

What are the main concepts in the ER model?



ER Model Basics

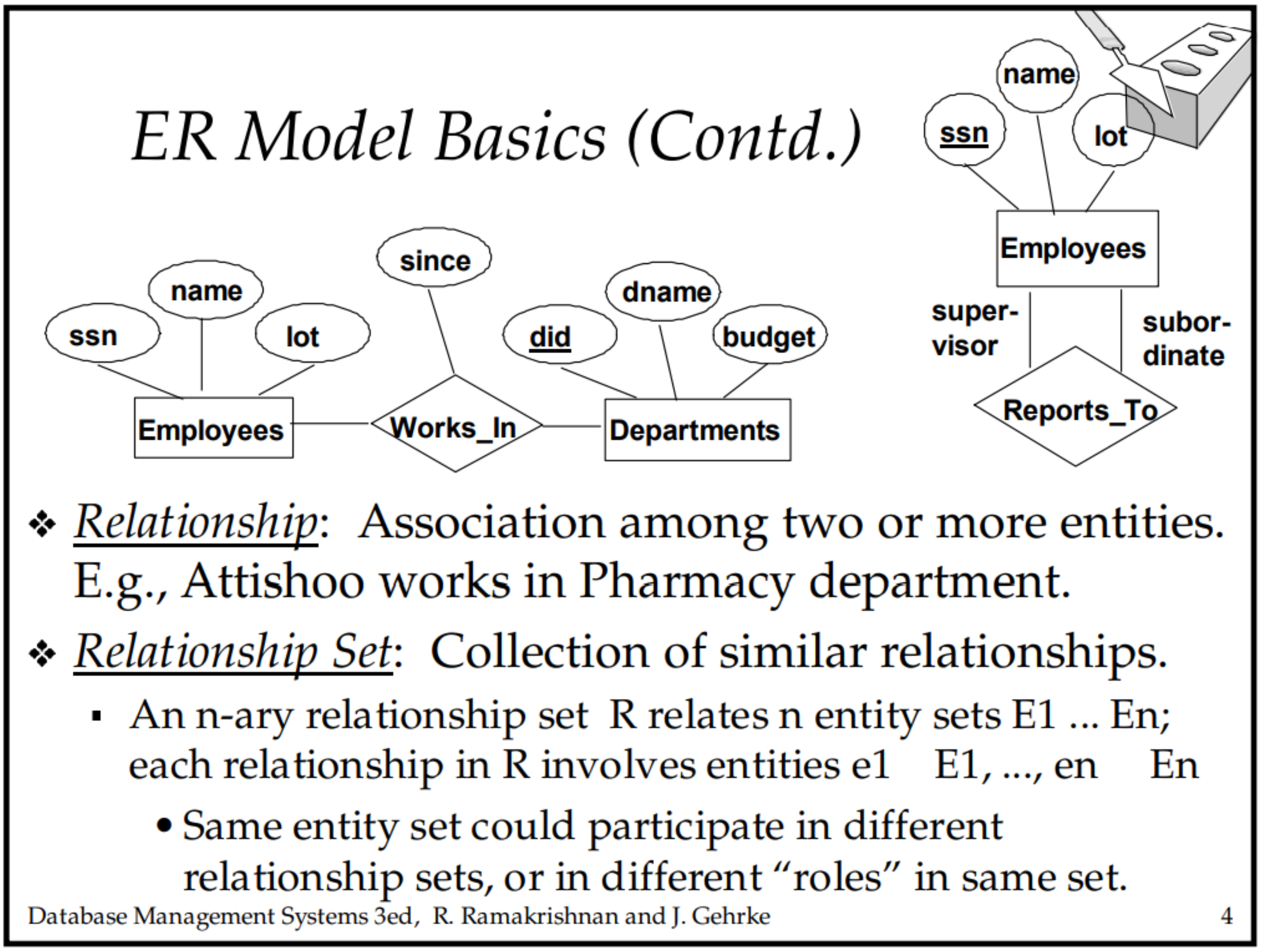
ssn
name
lot
Employees

- *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*.

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 3

key
: a minimal set of attributes whose values uniquely identify an entity in the set

There could be more than one **candidate key**; if so, we designate one of them as the **primary key**.


name
ssn
lot
Employees
super-
visor
subor-
dinate
Reports_To
since
name
ssn
lot
dname
did
budget
Employees
Works_In
Departments

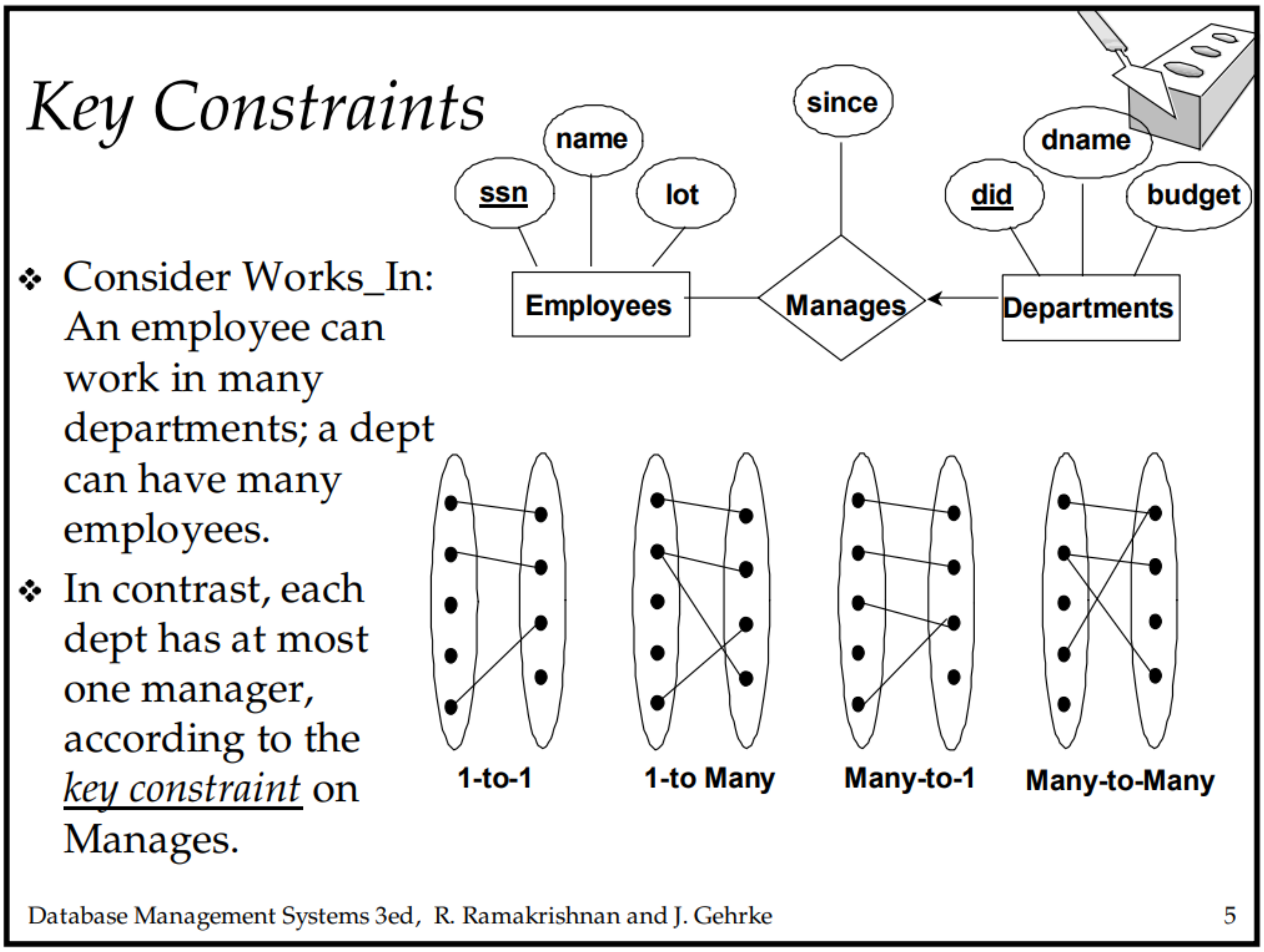
ER Model Basics (Contd.)

- *Relationship*: Association among two 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 E1, ..., en En
    - Same entity set could participate in different relationship sets, or in different "roles" in same set.

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 4

descriptive attributes: used to record information about the relationship, rather than about any one of the participating entities

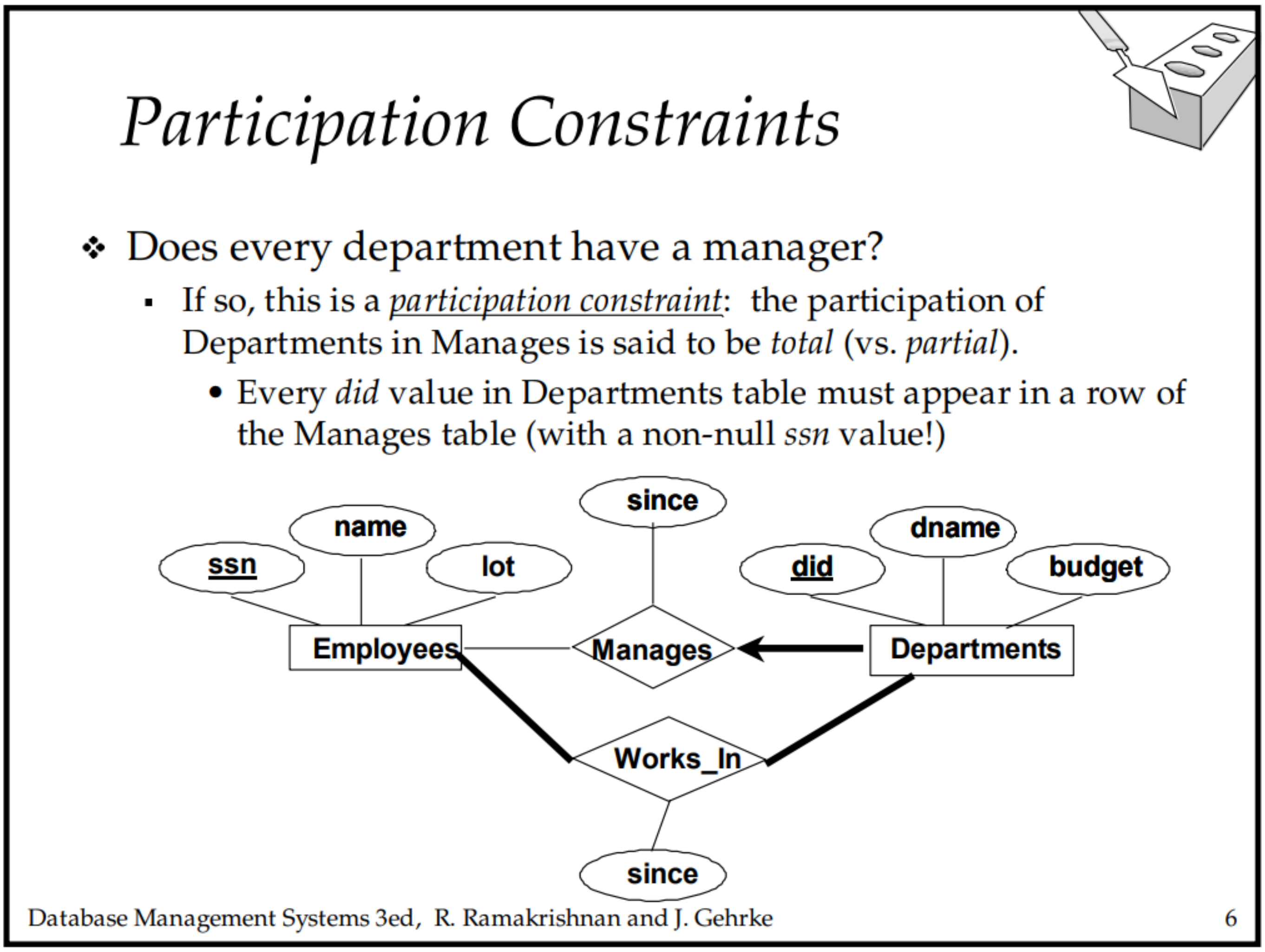
What are guidelines for using the ER model effectively?



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.

since
name
dname
ssn
lot
did
budget
Employees
Manages
Departments
1-to-1
1-to Many
Many-to-1
Many-to-Many

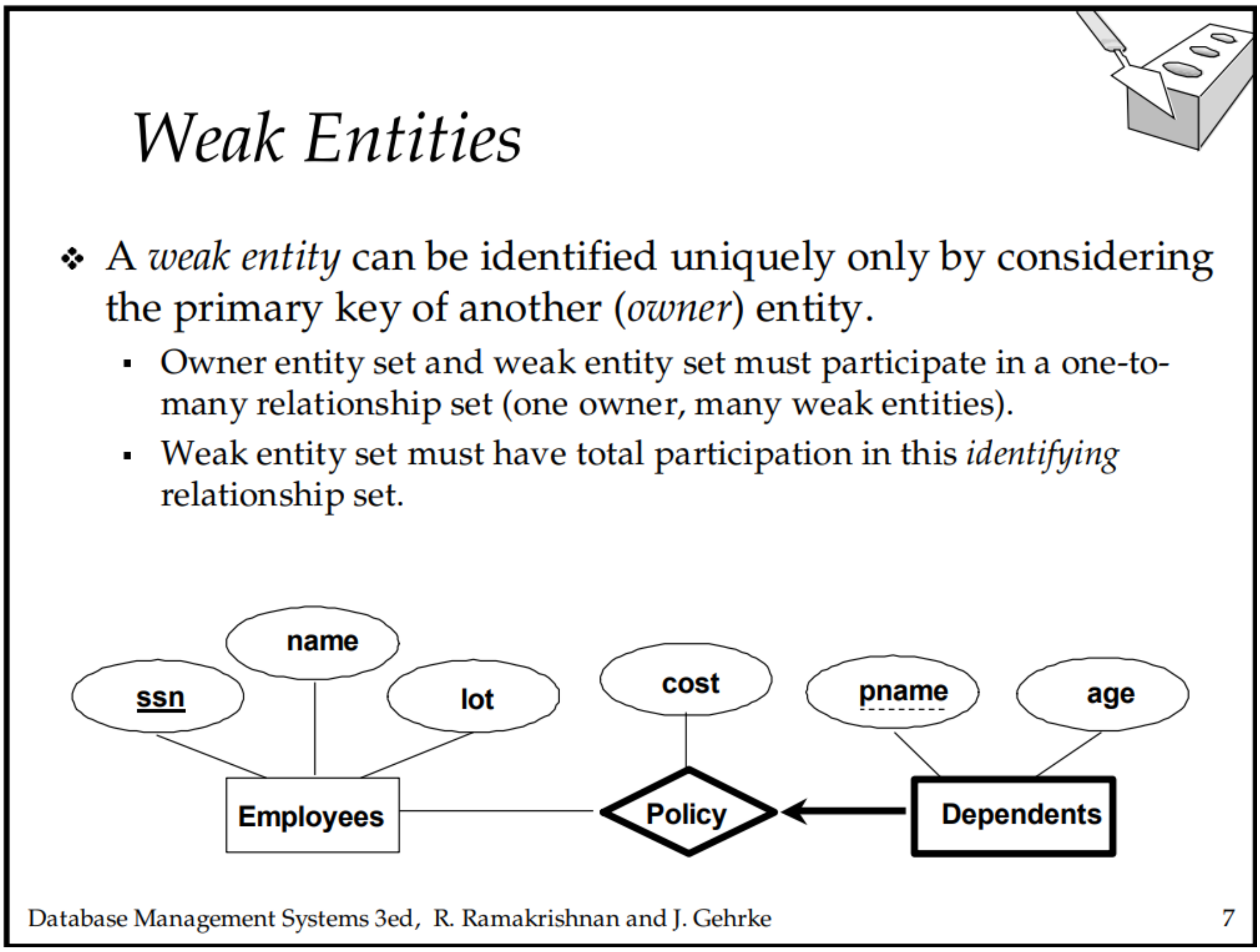


Participation Constraints

- 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 *did* value in Departments table must appear in a row of the Manages table (with a non-null *ssn* value!)

since
name
dname
ssn
lot
did
budget
Employees
Manages
Departments
Works_In
since

If the participation of an entity set in a relationship set is total, the two are connected by a thick line; independently, the presence of an arrow indicates a key constraint.



Weak Entities

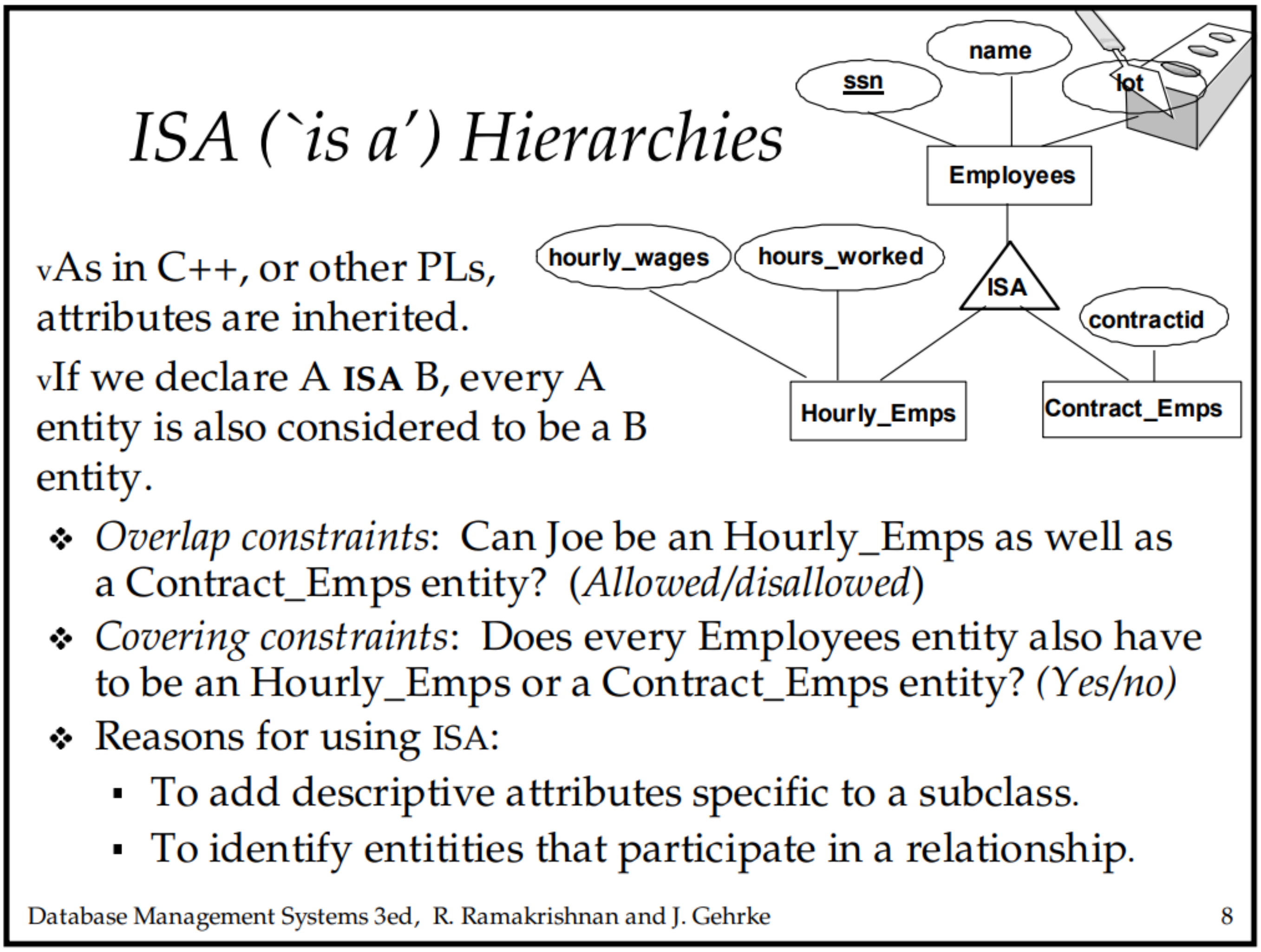
- A *weak entity* can be identified uniquely only by considering the primary key of another (*owner*) entity.
  - Owner entity set and weak entity set must participate in a one-to-many relationship set (one owner, many weak entities).
  - Weak entity set must have total participation in this *identifying*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

ssn
name
lot
cost
pname
age
Employees
Policy
Dependents

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 7

- **identifying relationship set:** The owner entity set and the weak entity set must participate in a one-to-many relationship set.
- The weak entity set must have total participation in the identifying relationship set.

How does database design fit within the overall design framework for complex software within large enterprises?


name
ssn
lot
Employees
hourly_wages
hours_worked
ISA
contractid
Hourly_Emps
Contract_Emps

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.

- *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 entitities that participate in a relationship.

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 8

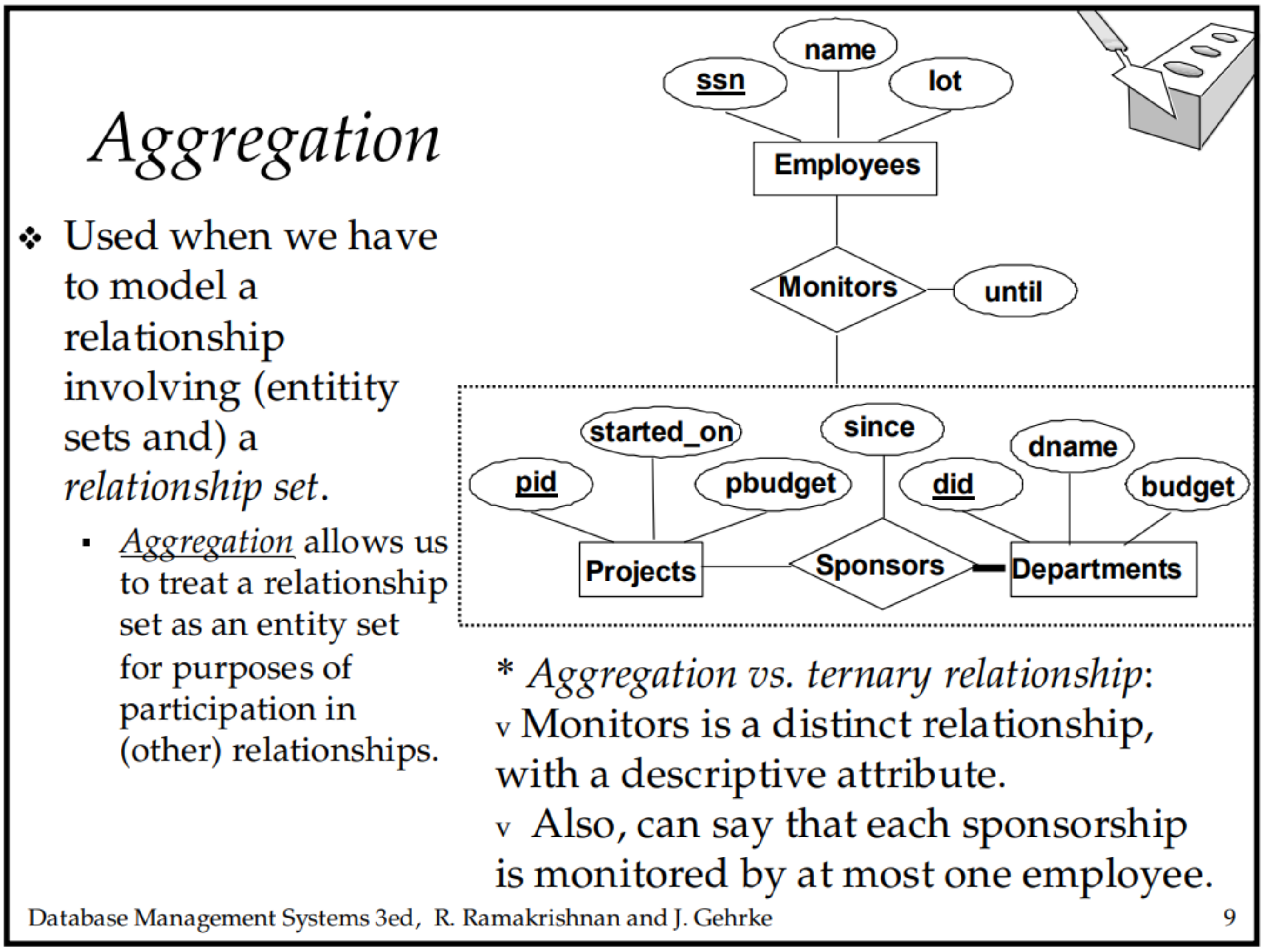
We say that the attributes for the entity set Employees are inherited by the entity set Hourly Emps and that Hourly Emps ISA (read is a) Employees.

- Employees is **specialized** into subclasses. Specialization is the process of identifying subsets of an entity set (the **superclass**) that share some distinguishing characteristic.
- Hourly Emps and Contract Emps are **generalized** by Employees.
- **Overlap constraints** determine whether two subclasses are allowed to contain the same entity. For example, can Attishoo be both an Hourly Emps entity and a Contract Emps entity? Intuitively, **no**. Can he be both a Contract Emps entity and a Senior Emps entity? Intuitively, **yes**. We denote this by writing '**Contract Emps OVERLAPS Senior Emps.**'
- **Covering constraints** determine whether the entities in the subclasses collectively include all entities in the superclass.For example, does every Employees entity have to belong to one of its subclasses? Intuitively, no. Does every Motor Vehicles entity have to be either a Motorboats entity or a Cars entity? Intuitively, yes. A characteristic property of generalization hierarchies is that every instance of a superclass is an instance of a subclass. We denote this by writing '**Motorboats AND Cars COVER Motor Vehicles.**'

There are two basic reasons for identifying subclasses (by specialization or generalization):

1. We might want to add descriptive attributes that make sense only for the entities in a subclass. For example, hourly wages does not make sense for a Contract Emps entity, whose pay is determined by

an individual contract.
2. We might want to identify the set of entities that participate in some relationship. For example, we might wish to define the Manages relationship so that the participating entity sets are Senior Emps and Departments, to ensure that only senior employees can be managers. As another example, Motorboats and Cars may have different descriptive attributes (say, tonnage and number of doors), but as Motor Vehicles entities, they must be licensed. The licensing information can be captured by a Licensed To relationship between Motor Vehicles and an entity set called Owners.



Aggregation

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

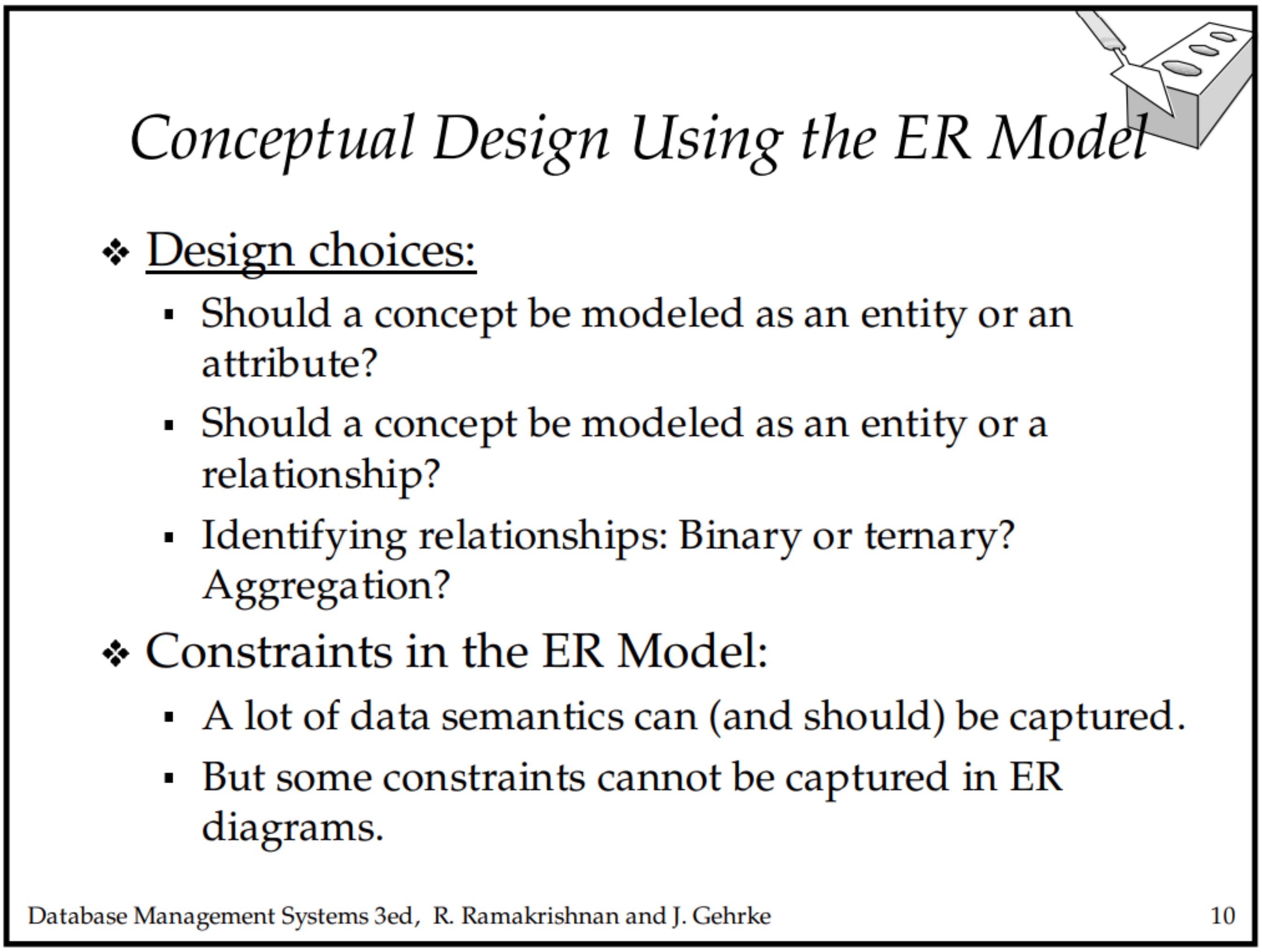
* *Aggregation vs. ternary relationship*:
- Monitors is a distinct relationship, with a descriptive attribute.
- Also, can say that each sponsorship is monitored by at most one employee.

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 9

Suppose that we have an entity set called Projects and that each Projects entity is sponsored by one or more departments. The Sponsors relationship set captures this information. A department that sponsors a project might assign employees to monitor the sponsorship. Intuitively, Monitors should be a relationship set that associates a Sponsors relationship (rather than a Projects or Departments entity) with an Employees entity. However, we have defined relationships to associate two or more entities.

To define a relationship set such as Monitors, we introduce a new feature of the ER model, called aggregation.

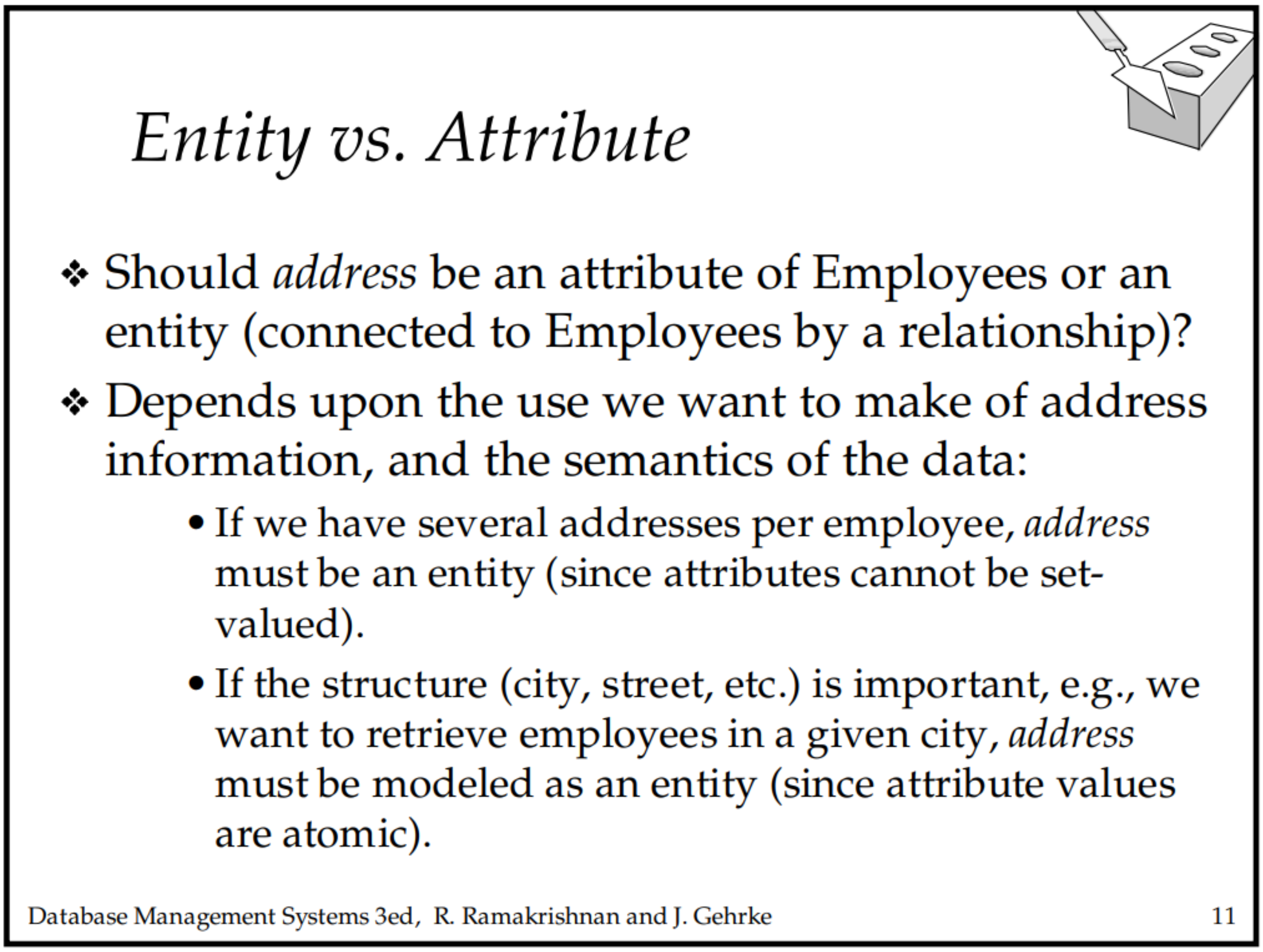
Aggregation
: allows us to indicate that a relationship set (identified through a dashed box) participates in another relationship set.



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.

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 10



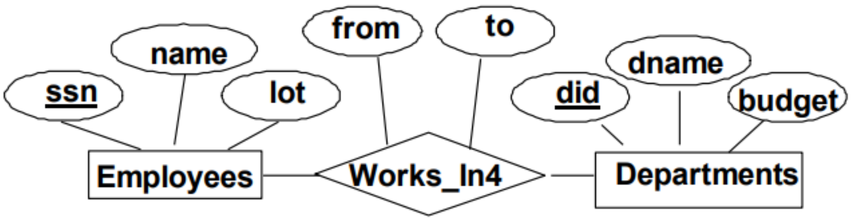
Entity vs. Attribute

- 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).

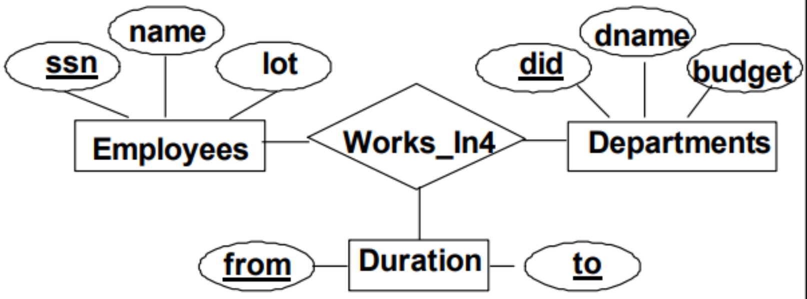
Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 11

Entity vs. Attribute (Contd.)

- Works_In4 does not allow an employee to work in a department for two or more periods.

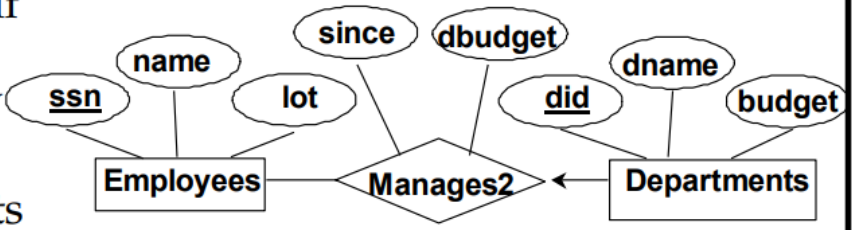


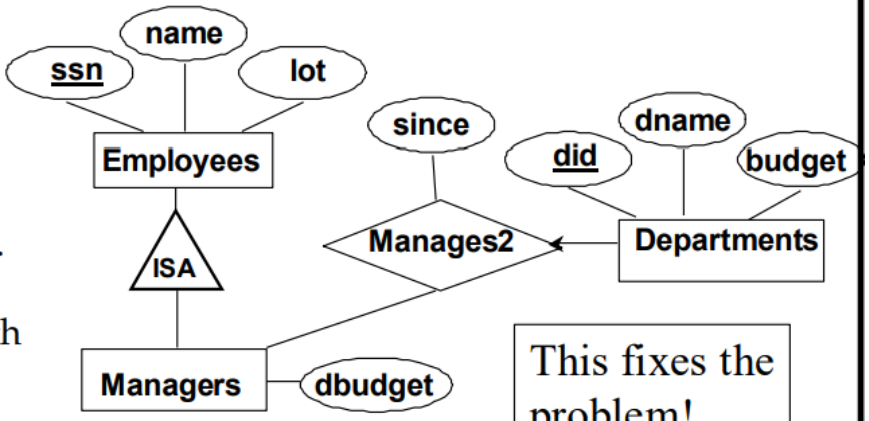
- Similar to the problem of wanting to record several addresses for an employee: We want to record *several values of the descriptive attributes for each instance of this relationship.* Accomplished by introducing new entity set, Duration.



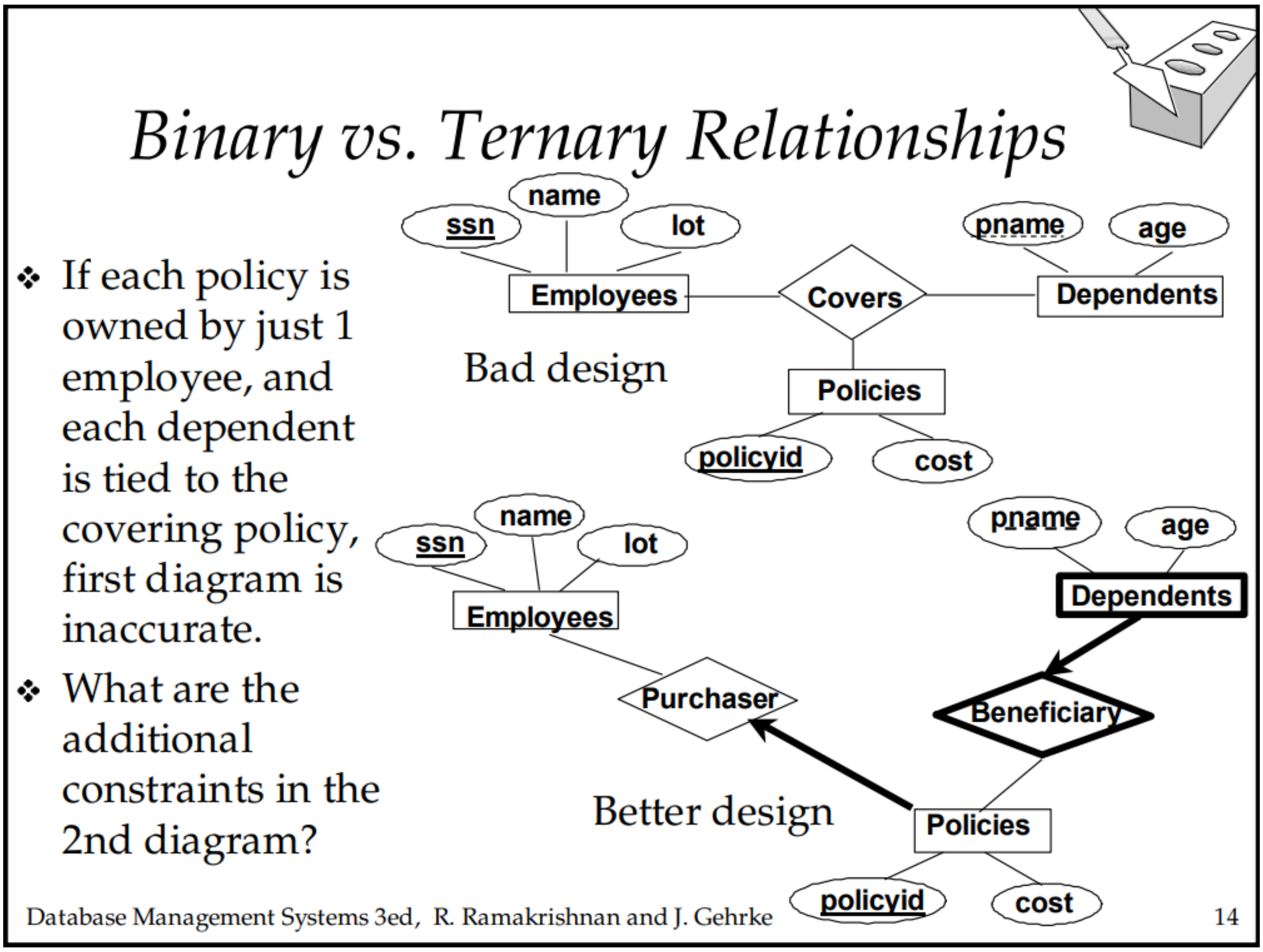
Entity vs. Relationship

- 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.





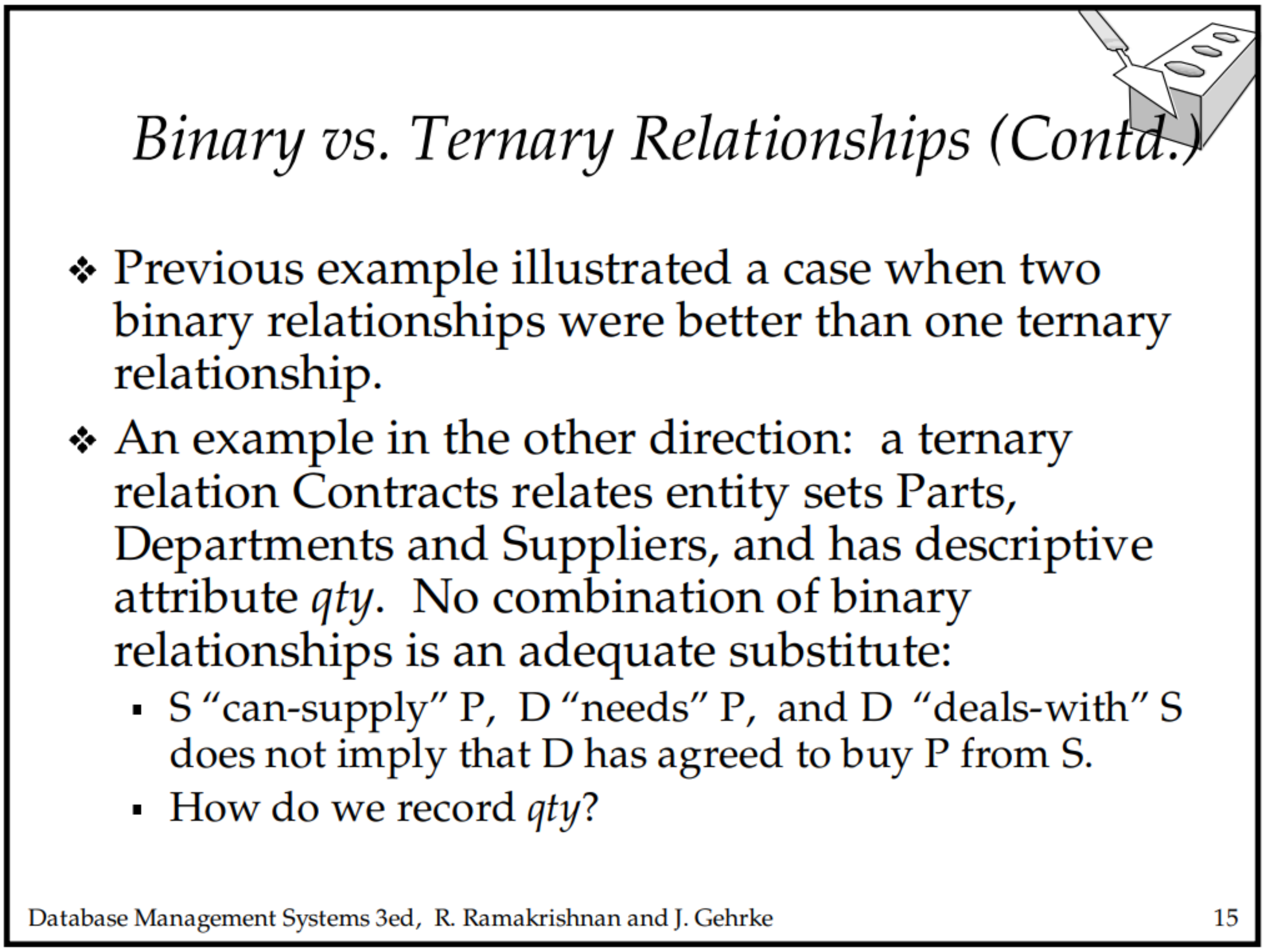
This fixes the problem!



Binary vs. Ternary Relationships

- If each policy is owned by just 1 employee, and each dependent is tied to the covering policy, first diagram is inaccurate.
- What are the additional constraints in the 2nd diagram?

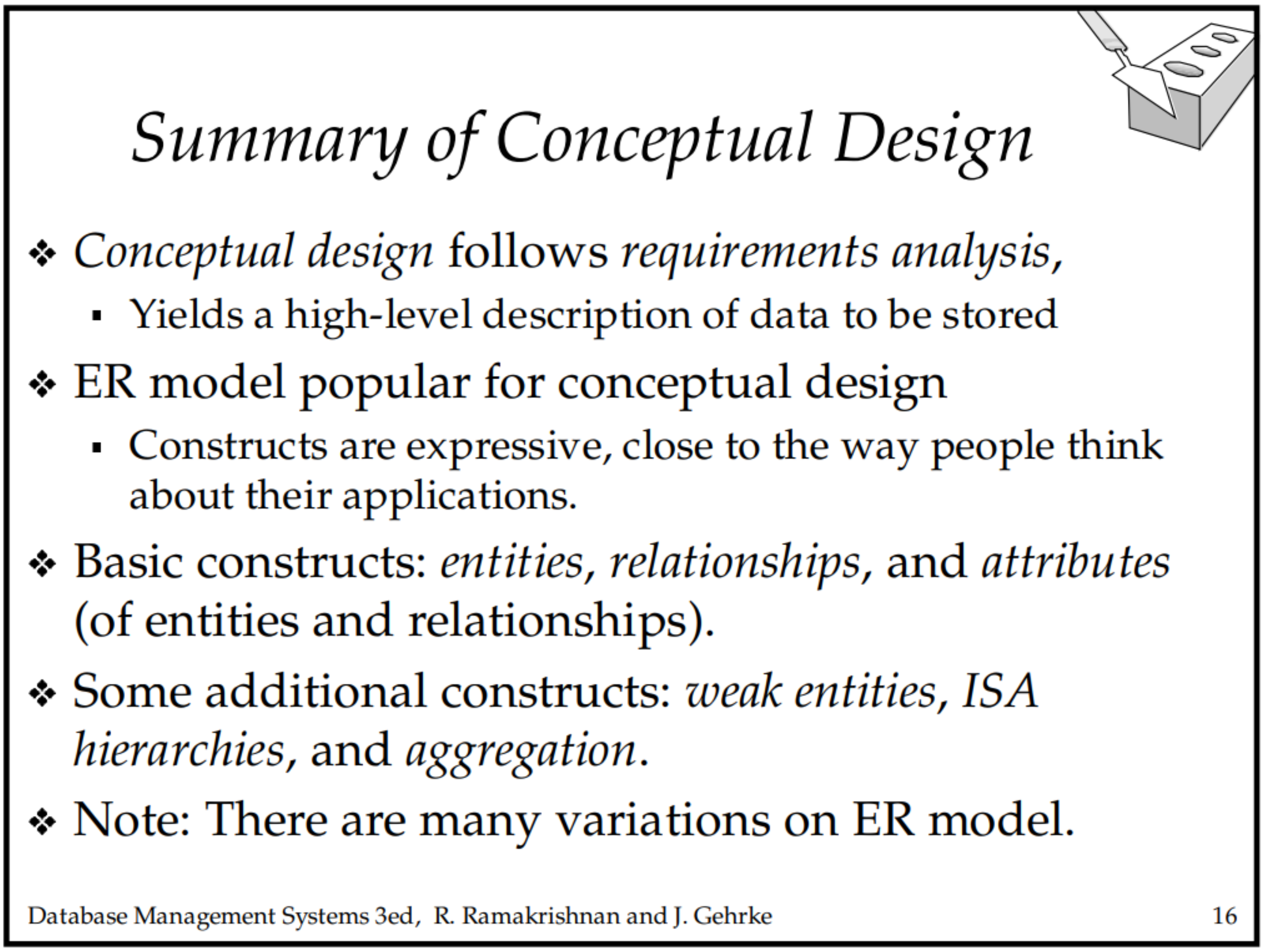
name
ssn
lot
Employees
Covers
pname
age
Dependents
Bad design
Policies
policyid
cost
name
ssn
lot
Employees
pname
age
Dependents
Purchaser
Beneficiary
Better design
Policies
policyid
cost



Binary vs. Ternary Relationships (Contd.)

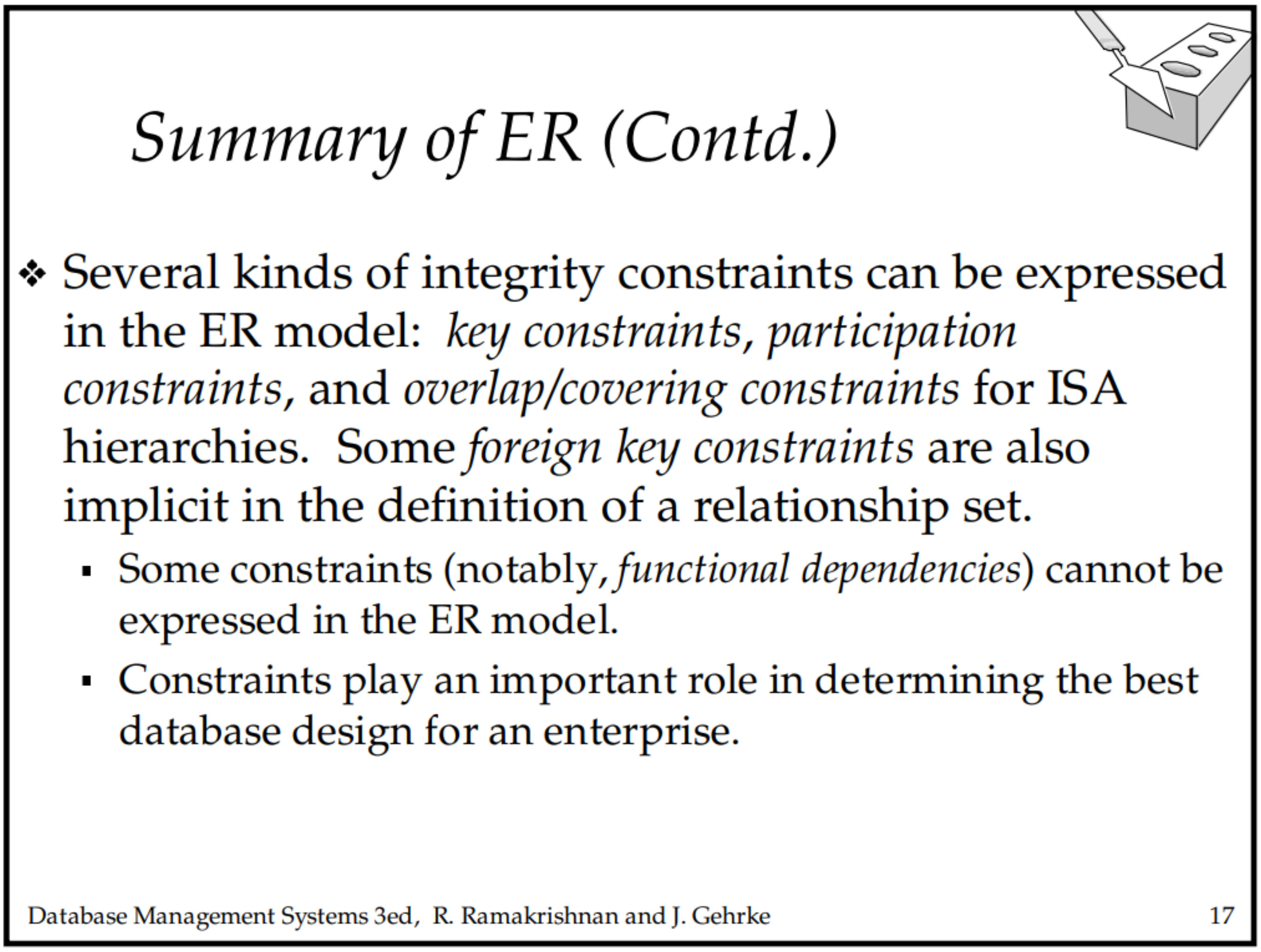
- Previous example illustrated a case when two binary relationships were better than one ternary relationship.
- An example in the other direction: a ternary relation Contracts relates entity sets Parts, Departments and Suppliers, and has descriptive attribute *qty*. No combination of binary relationships 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*?

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 15



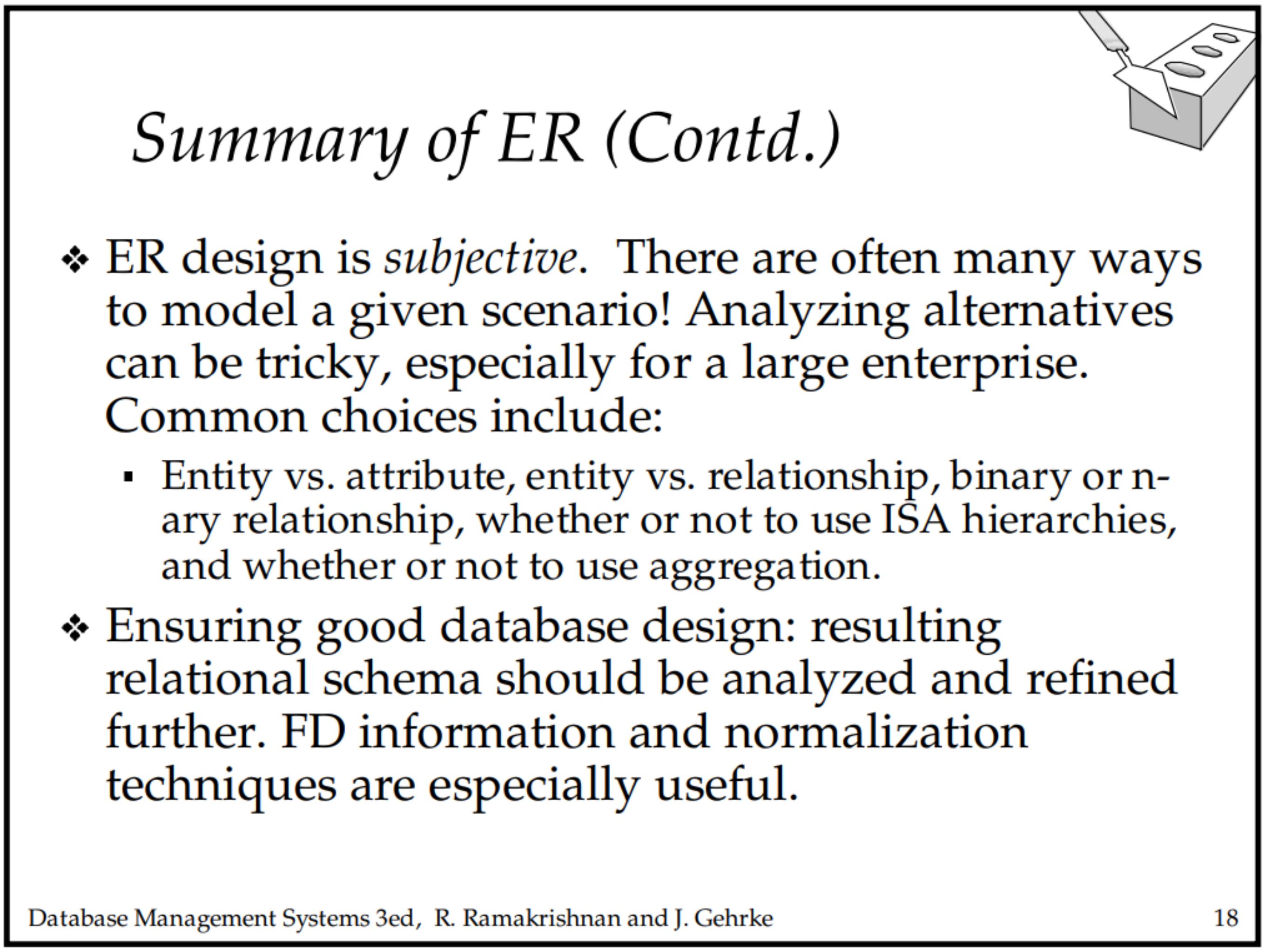
Summary of Conceptual Design

- *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.



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.

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 18

What is UML and how is it related to the ER model?

unified modeling language (UML) approach

> like the ER model, has the attractive feature that its constructs can be drawn as diagrams. It encompasses a broader spectrum of the software design process than the ER model

CASE STUDY: THE INTERNET SHOP

B&N is a large bookstore specializing in books on horse racing, and it has decided to go online. DBDudes first verifies that B&N is willing and able to pay its steep fees and then schedules a lunch meeting—billed to B&N, naturally—to do requirements analysis.

Requirements Analysis

“I would like my customers to be able to browse my catalog of books and place orders over the Internet. Currently, I take orders over the phone. I have mostly corporate customers who call me and give me the ISBN number of a book and a quantity; they often pay by credit card. I then prepare a shipment that contains the books they ordered. If I don’t have enough copies in stock, I order additional copies and delay the shipment until the new copies arrive; I want to ship a customer’s entire order together. My catalog includes all the books I sell. For each book, the catalog contains its ISBN number, title, author, purchase price, sales price, and the year the book was published. Most of my customers are regulars, and I have records with their names and addresses.

New customers have to call me first and establish an account before they can use my website.

On my new website, customers should first identify themselves by their unique customer identification number. Then they should be able to browse my catalog and to place orders online."

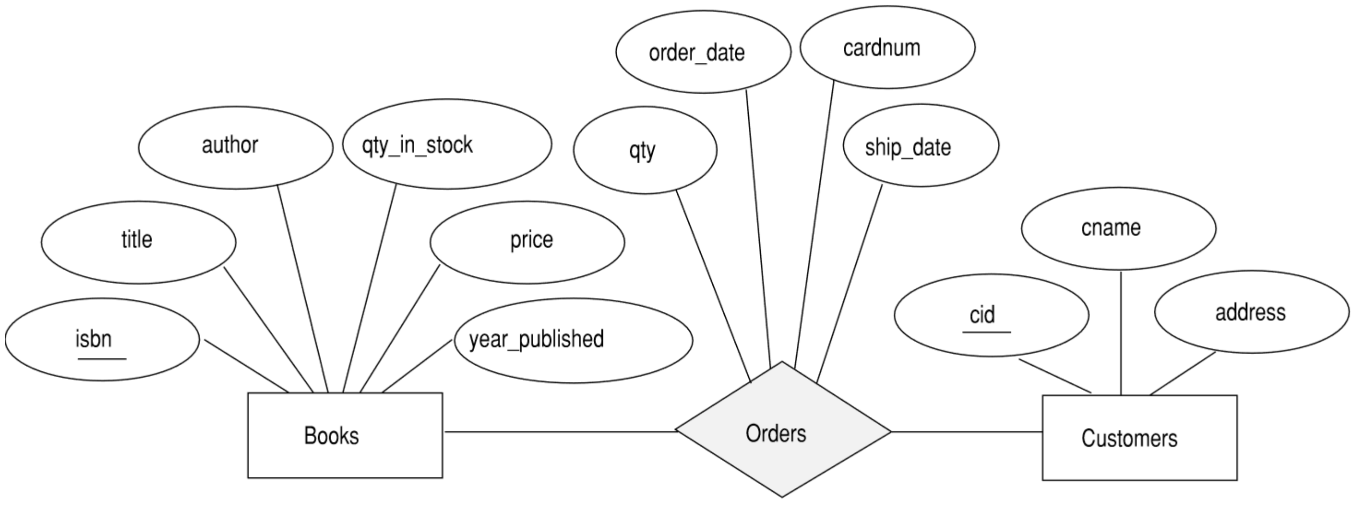


Figure 2.20 ER Diagram of the Initial Design

Conceptual Design

Books and customers are modeled as entities and related through orders that customers place. Orders is a relationship set connecting the Books and Customers entity sets. For each order, the following attributes are stored: quantity, order date, and ship date. As soon as an order is shipped, the ship date is set; until then the ship date is set to null, indicating that this order has not been shipped yet.