Information Systems Analysis & Modeling

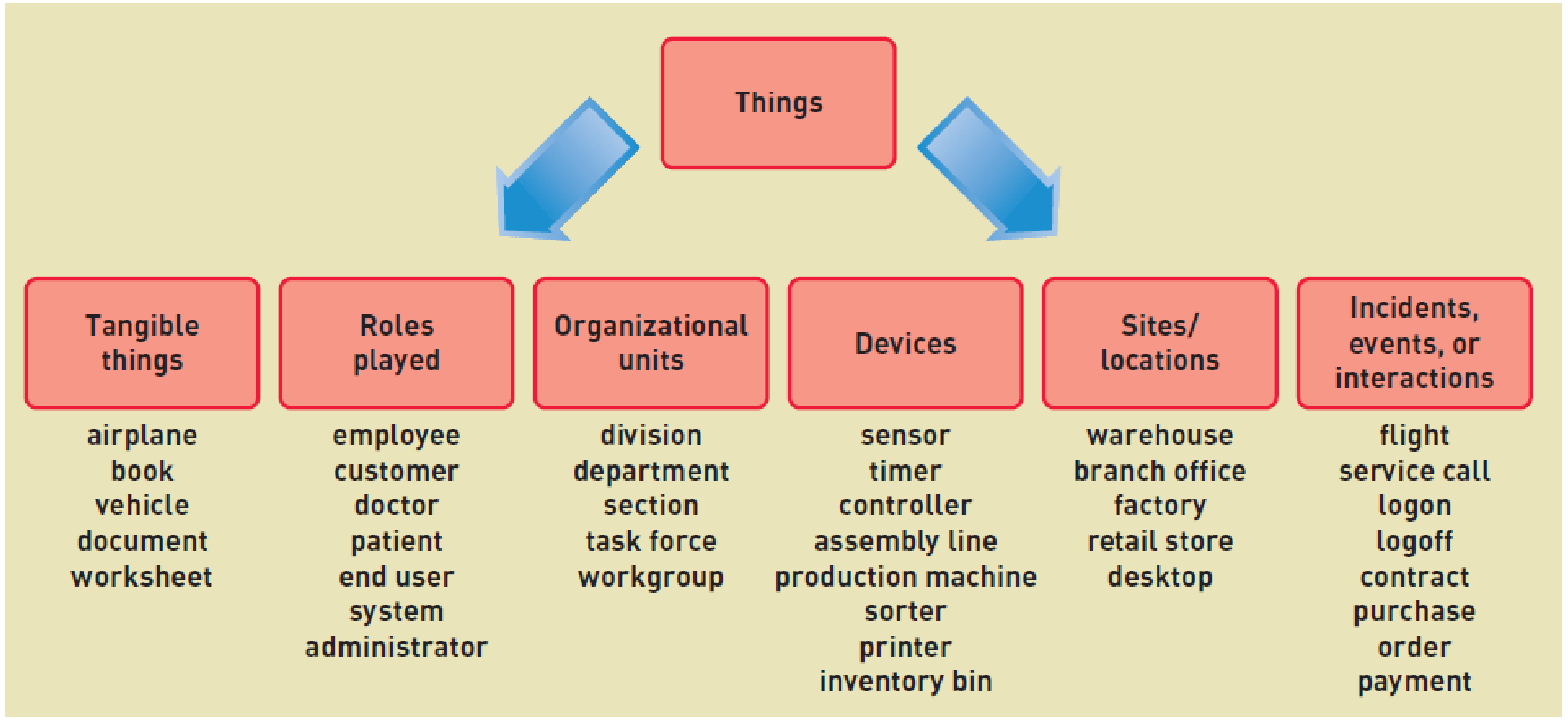
Lecture 8: ERD diagram

Mona Taghavi

LaSalle College
Montréal

Data Entities

- Domain classes or data entities are what end users deal with when they do their work—for example, products, sales, shippers, shipments, and customers. These are often referred to as “things” in the context of a system’s problem domain.



Two techniques of identifying “Things”

Brainstorming technique:

1. Identify a user and a set of use cases or user stories.
2. Brainstorm with the user to identify things involved when carrying out the use case—that is, things about which information should be captured by the system.
3. Use the types of things (categories) to systematically ask questions about potential things, such as the following: Are there any tangible things you store information about? Are there any locations involved? Are there roles played by people that you need to remember?
4. Continue to work with all types of users and stakeholders to expand the brainstorming list.
5. Merge the results, eliminate any duplicates, and compile an initial list.

Two techniques of identifying “Things”

Noun technique:

- Recall that a noun is a person, place, or thing. Therefore, identifying nouns might help you identify what needs to be stored by the system. Begin by listing all the nouns that users mention when talking about the system. Nouns used to describe events, use cases, user stories, and the actors are potential things.

1. Using the use cases, actors, and other information about the system—including inputs and outputs—identify all nouns.
   - For the RMO CSMS, the nouns might include the following: customer, sale, confirmation, transaction, shipping, bank, change request, summary report, back order log, back-order, return, return confirmation, etc.
2. Using other information from existing systems, current procedures, and current reports or forms, add items or categories of information needed.
   - For the RMO CSMS, these might include more detailed information, such as price, size, color, style, season, inventory quantity, payment method, and shipping address. Some of these items might be additional things, and some might be specific information (called attributes) about things you have already identified.

Noun technique

3. As this list of nouns builds, you will need to refine it. Ask these questions about each noun to help you decide whether you should include it:
   - ❑ Is it a unique thing the system needs to know about?
   - ❑ Is it inside the scope of the system I am working on?
   - ❑ Does the system need to remember more than one of these items?

   - Ask these questions about each noun to decide whether you should exclude it:
   - ❑ Is it really a synonym for some other thing I have identified?
   - ❑ Is it really just an output of the system produced from other information I have identified?
   - ❑ Is it really just an input that results in recording some other information I have identified?

   - Ask these questions about each noun to decide whether you should research it:
     - ❑ Is it likely to be a specific piece of information (attribute) about some other thing I have identified?
     - ❑ Is it something I might need if assumptions change?

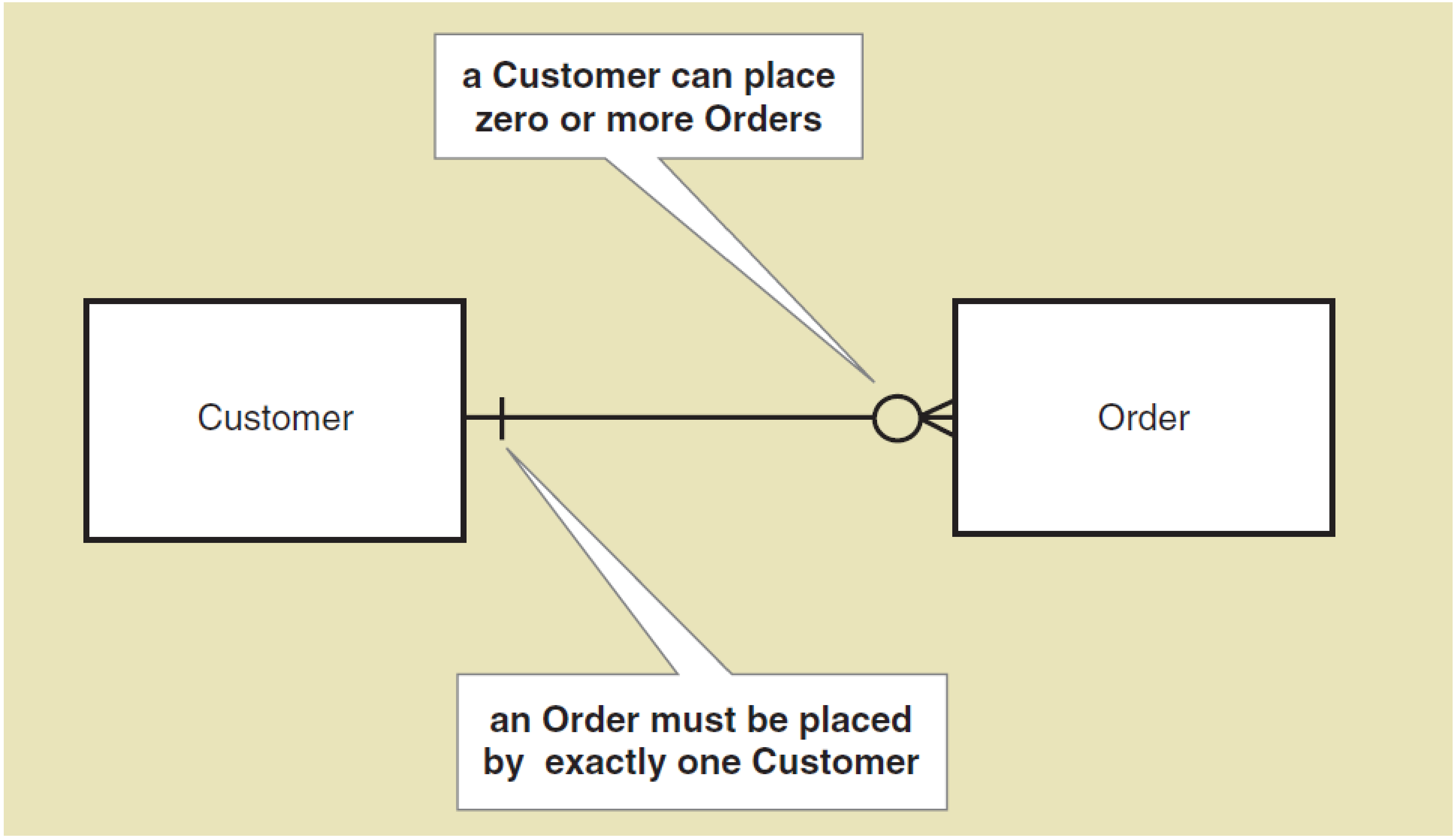
4. Review the list with users, stakeholders, and team members and then refine the list of things in the problem domain.

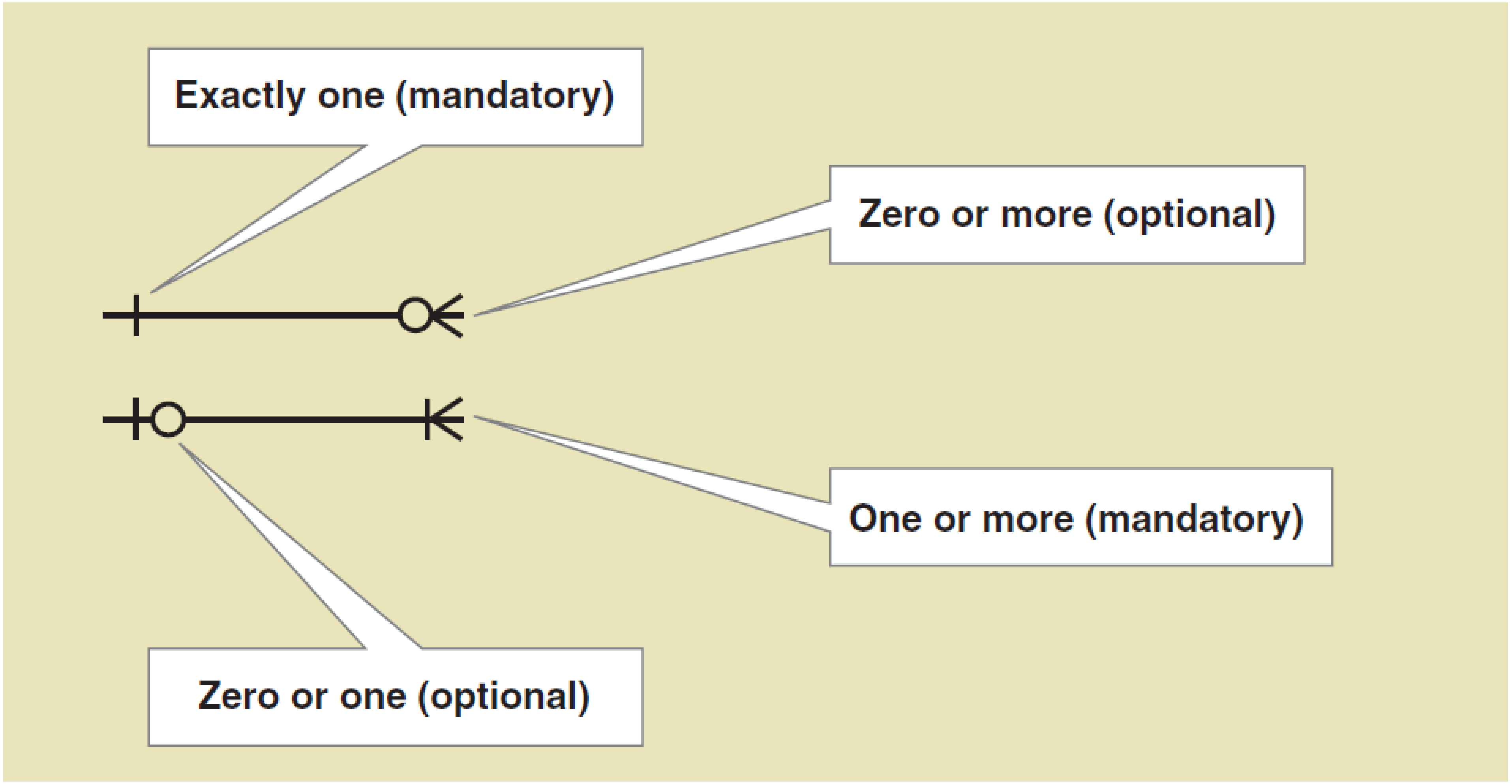
Attributes

- The noun technique involves listing all the nouns that come up in discussions or documents about the requirements. The list can become quite long because many of these nouns are actually attributes.
    - For example, a customer has a name, a phone number, a credit limit, and so on. Each of these details is an attribute. So as you refine your list of nouns, you may redefine many nouns as attributes rather than fundamental "things."
- The attribute that uniquely identifies the thing is called a **key**.
- A system may need to remember many similar attributes. For example, a customer has several names: a first name, a middle name, a last name, and possibly a nickname. A **compound attribute** is an attribute that contains a collection of related attributes, so an analyst may choose one compound attribute to represent all these names, perhaps naming it *Customer full name*.

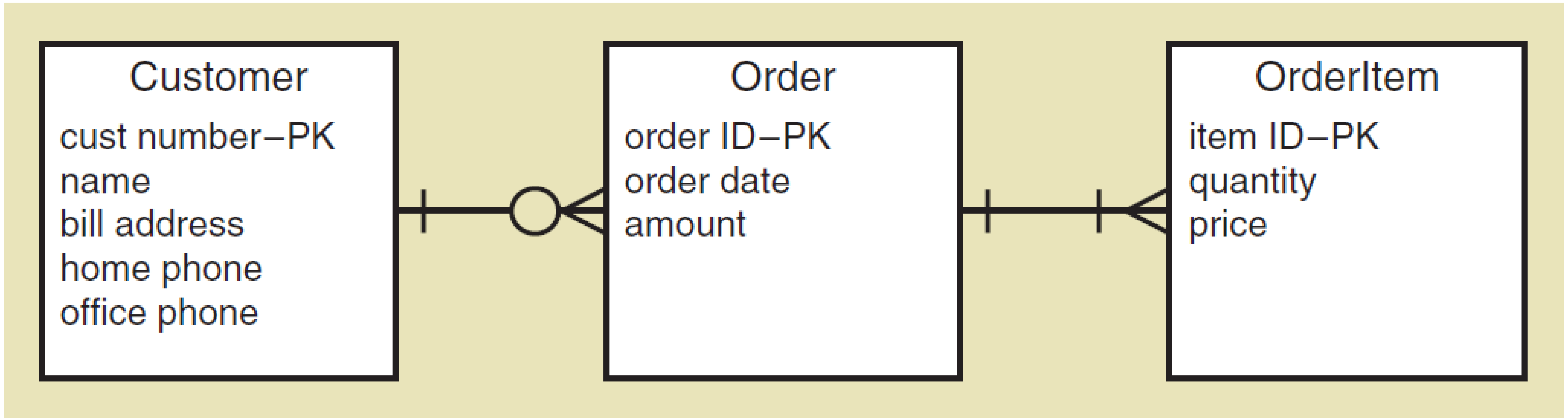
Entity-relationship diagram (ERD)

- Traditional approaches to system development place a great deal of emphasis on data requirements for a new system and use the term **data entities** for the things about which the system needs to store information. Data requirements include the data entities, their attributes, and the relationships (called *associations* in UML) among the data entities.
- **Entity-relationship diagram (ERD)** is a diagram consisting of data entities, their attributes, and their relationships.
- The ERD is not a UML diagram, but it is frequently used and is quite similar to the UML domain model class diagram.

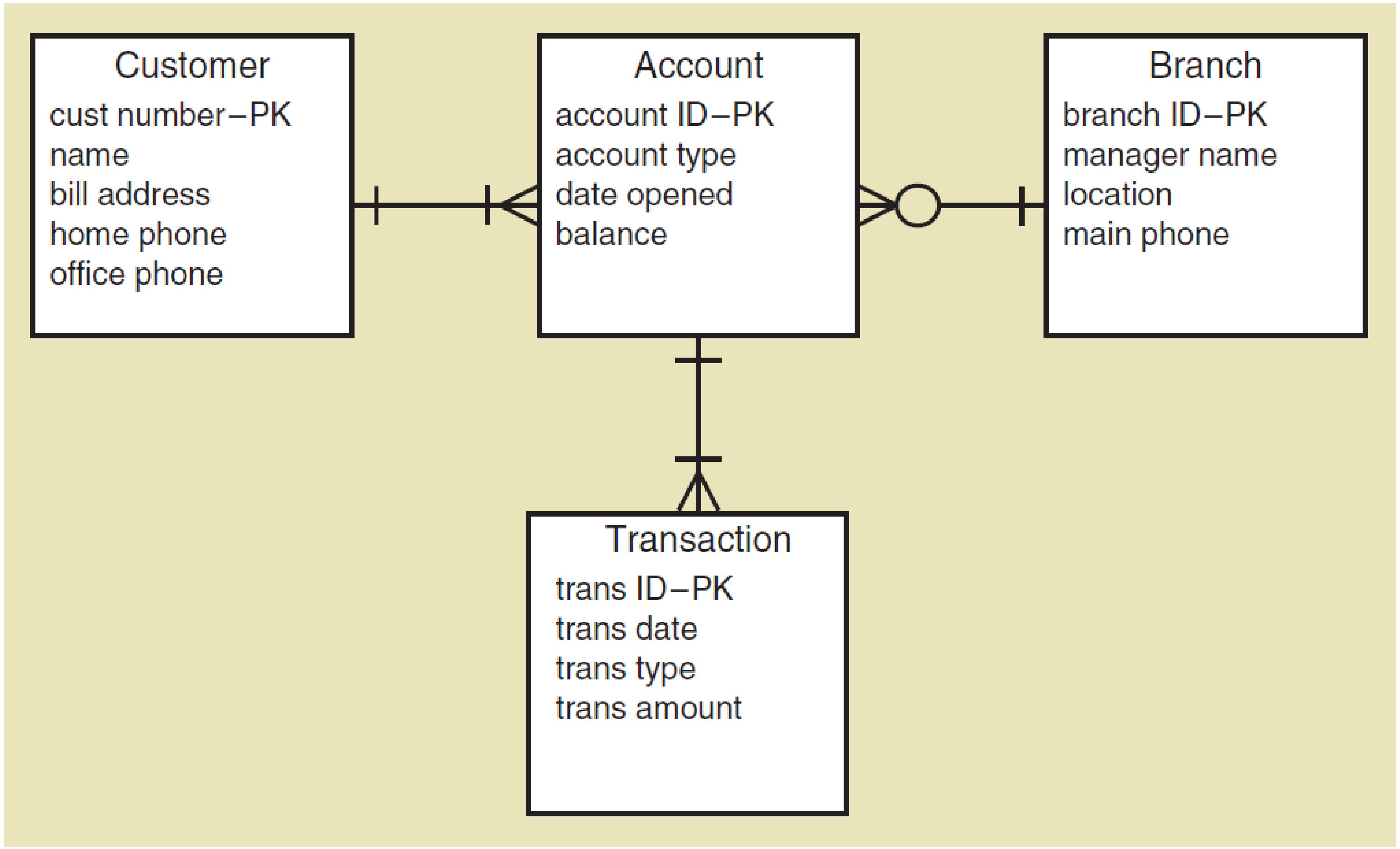

a Customer can place zero or more Orders
Customer
Order
an Order must be placed by exactly one Customer


Exactly one (mandatory)
Zero or more (optional)
One or more (mandatory)
Zero or one (optional)

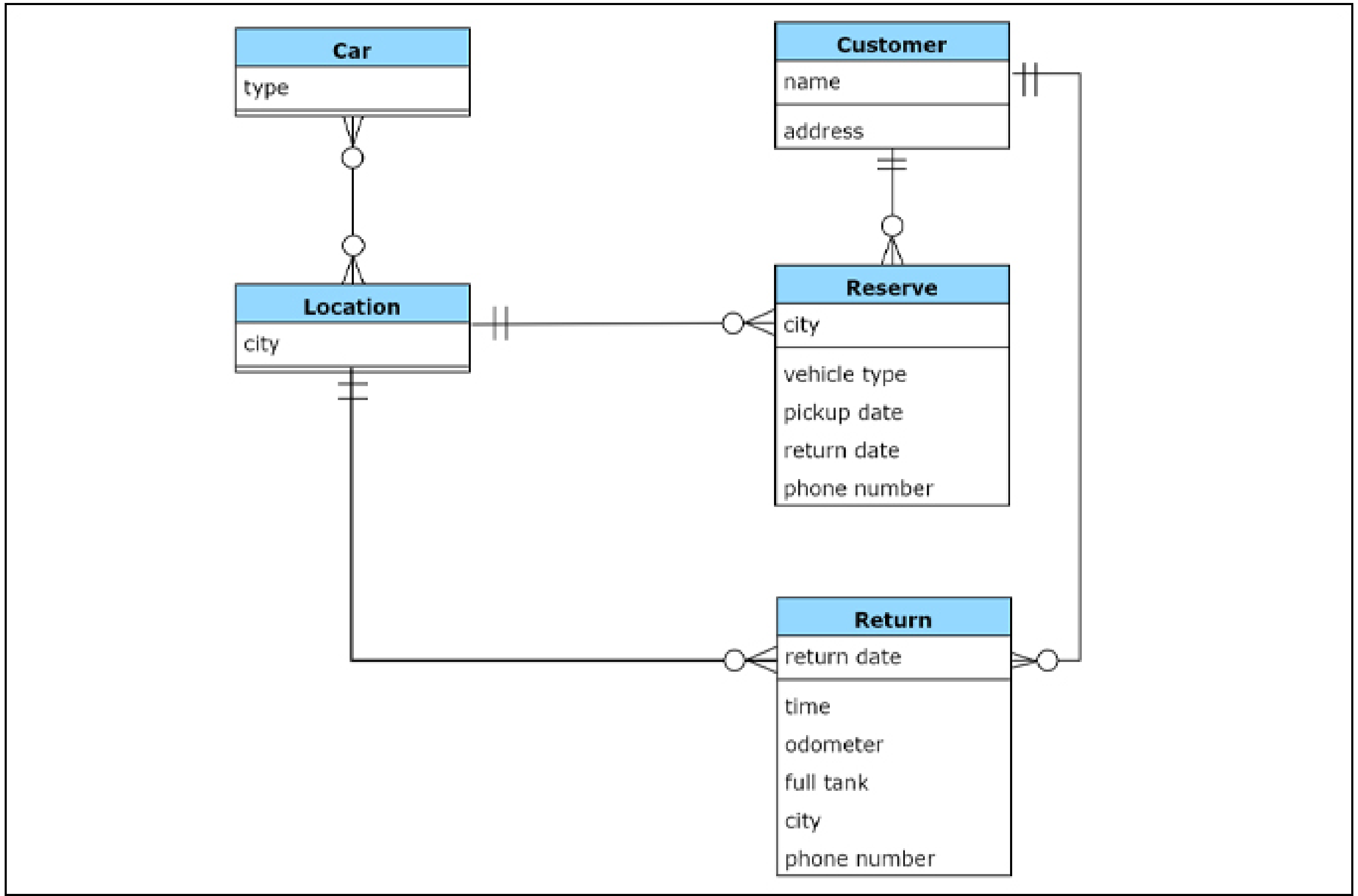
An expanded ERD with attributes shown

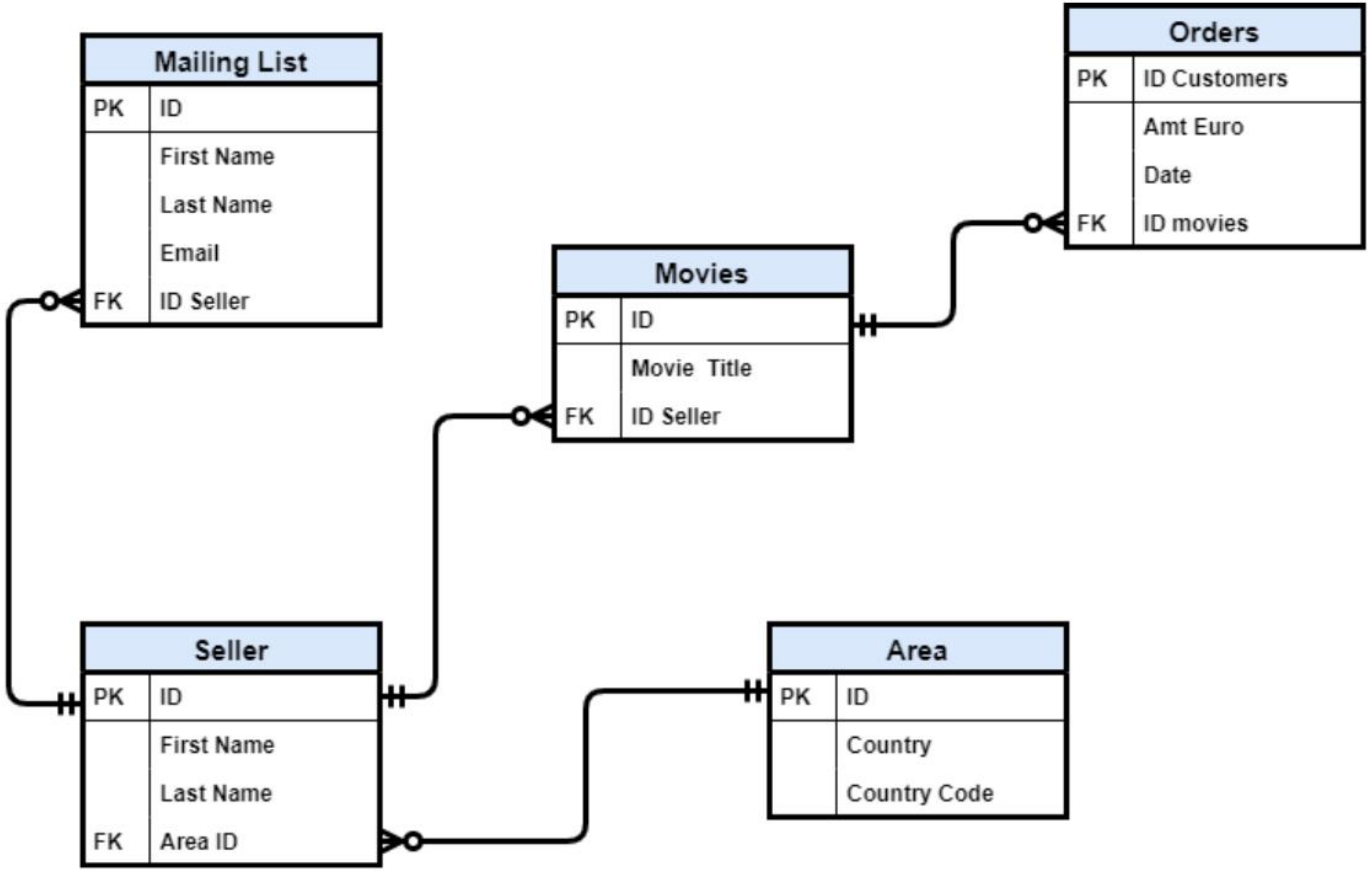


An ERD for a bank with many branches



Car rental systems





Mailing List
PK ID
First Name
Last Name
Email
FK ID Seller
Orders
PK ID Customers
Amt Euro
Date
FK ID movies
Movies
PK ID
Movie Title
FK ID Seller
Seller
PK ID
First Name
Last Name
FK Area ID
Area
PK ID
Country
Country Code