DATA MODELING WITH THE ENTITY-RELATIONSHIP MODEL

revised by 김태연

OBJECTIVES

- To understand the two-phase data modeling/database design process
- To understand the purpose of the data modeling process
- To understand entity-relationship (E-R) diagrams
- To be able to determine entities, attributes, and relationships
- To be able to create entity identifiers
- To be able to determine minimum and maximum cardinalities
- To understand and be able to use ID-dependent and other weak entities
- To understand and be able to use supertype/subtype entities

THE DATA MODEL

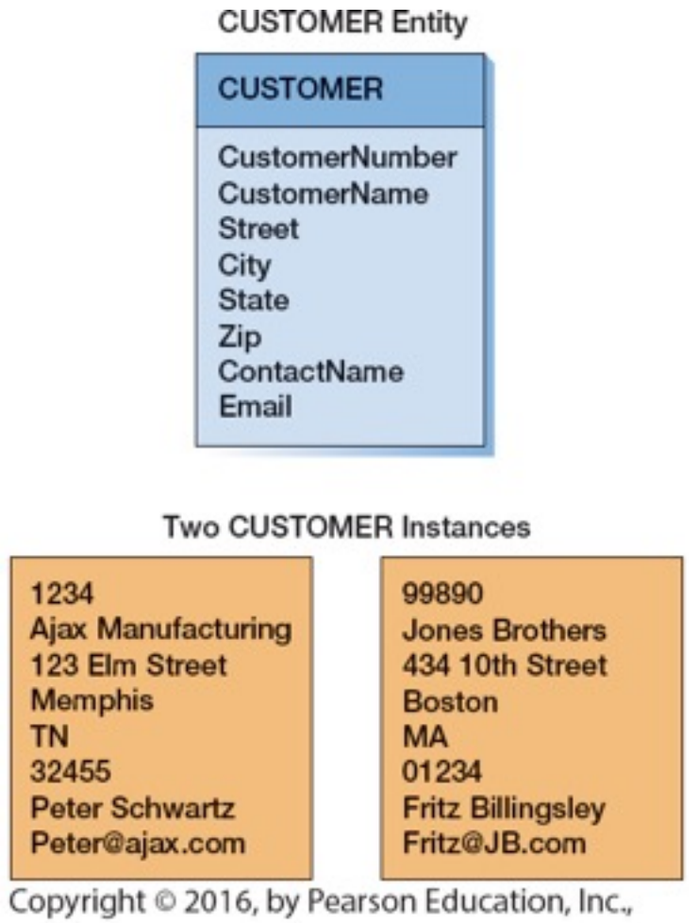
- A data model is a plan or blueprint for a database design.
- A data model is more generalized and abstract than a database design.
- It is easier to change a data model then it is to change a database design, so it is the appropriate place to work through conceptual database problems.
- Books on systems analysis and design often identify three design stages:
  - Conceptual design (conceptual schema)
  - Logical design (logical schema)
  - Physical design (physical schema)
- The data model we are discussing is equivalent to the conceptual design as defined in these books.

E-R MODEL

- Entity-Relationship model is a set of concepts and graphical symbols that can be used to create conceptual schemas.
- Versions:
  - Original E-R model—by Peter Chen (1976)
  - Extended E-R model—extensions to the Chen model (**+Subtype**)
  - Information Engineering (IE)—by James Martin (1990); uses **"crow's foot" notation**, is easier to understand, and we will use it
  - IDEF1X—a national standard developed by the National Institute of Standards and Technology
  - Unified Modeling Language (UML)—by the Object Management Group; it supports object-oriented methodology

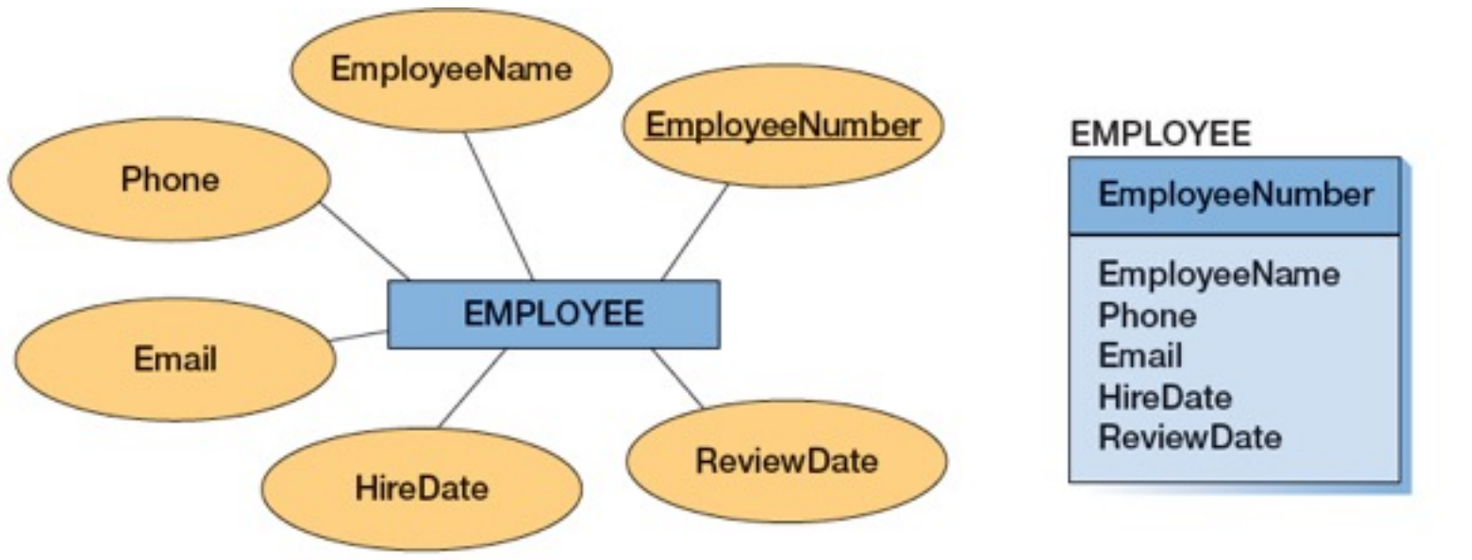
ENTITIES

- Something that can be identified and the users want to track:
  - Entity class—a collection of entities of a given type
  - Entity instance—the occurrence of a particular entity
- There are usually many instances of an entity in an entity class.



ATTRIBUTES

- Attributes describe an entity's characteristics.
- All entity instances of a given entity class have the same attributes, but vary in the values of those attributes.
- Originally shown in data models as ellipses.
- Data modeling products today commonly show attributes in rectangular form.



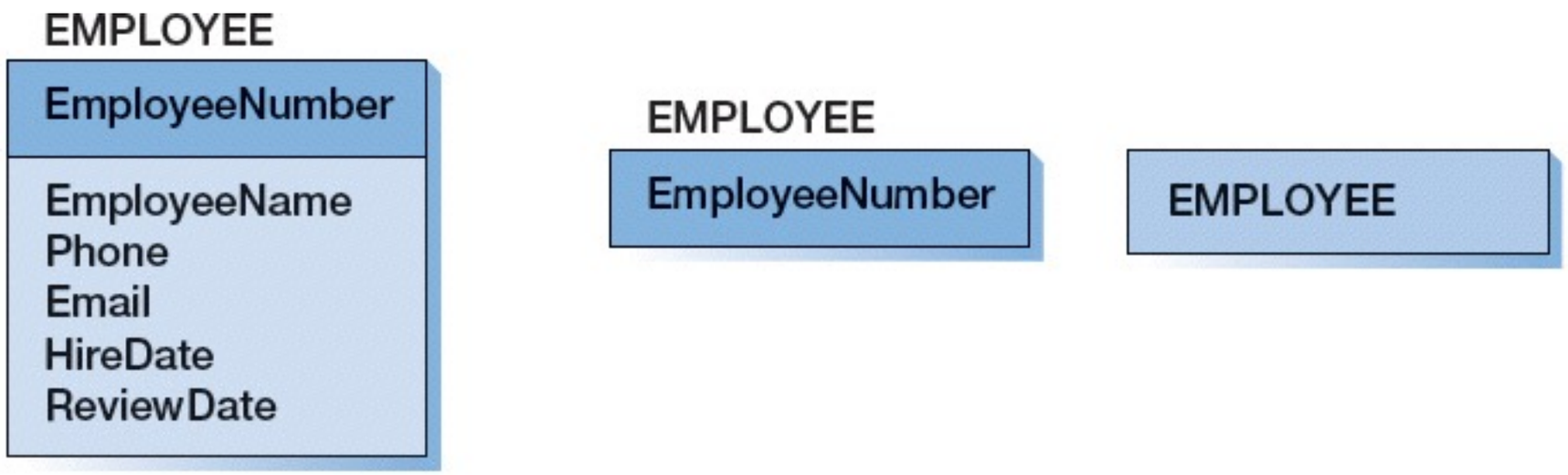
(a) Attributes in Ellipses

(b) Attributes in Rectangle

IDENTIFIERS

- Identifiers are attributes that name, or identify, entity instances.
- The identifier of an entity instance consists of one or more of the entity's attributes.
- Composite identifiers are identifiers that consist of two or more attributes.
- Identifiers in data models become keys in database designs.
  - Entities have identifiers.
  - Tables (or relations) have keys.

ENTITY ATTRIBUTE DISPLAY IN DATA MODELS



(a) Entity with All Attributes

(b) Entity with Identifier Attribute Only

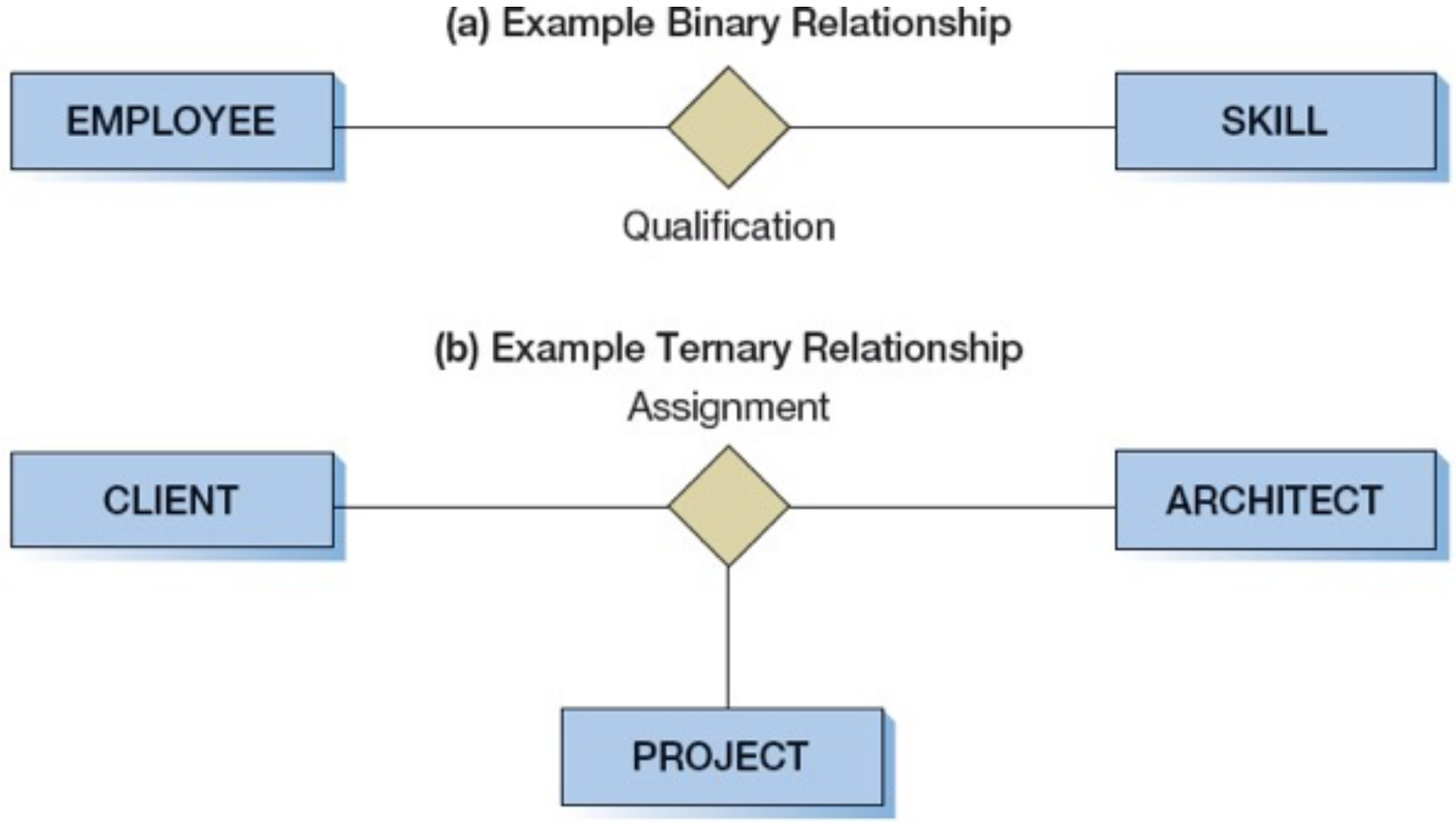
(c) Entity with No Attributes

RELATIONSHIPS

- Entities can be associated with one another in relationships:
  - Relationship classes: associations among entity classes
  - Relationship instances: associations among entity instances
- In the original E-R model, relationships could have attributes, but today this is no longer done.
- A relationship class can involve two or more entity classes.

BINARY AND TERNARY RELATIONSHIP

- The degree of the relationship is the number of entity classes in the relationship:
- Two entities have a binary relationship of degree two.
- Three entities have a ternary relationship of degree three.



ENTITIES AND TABLES

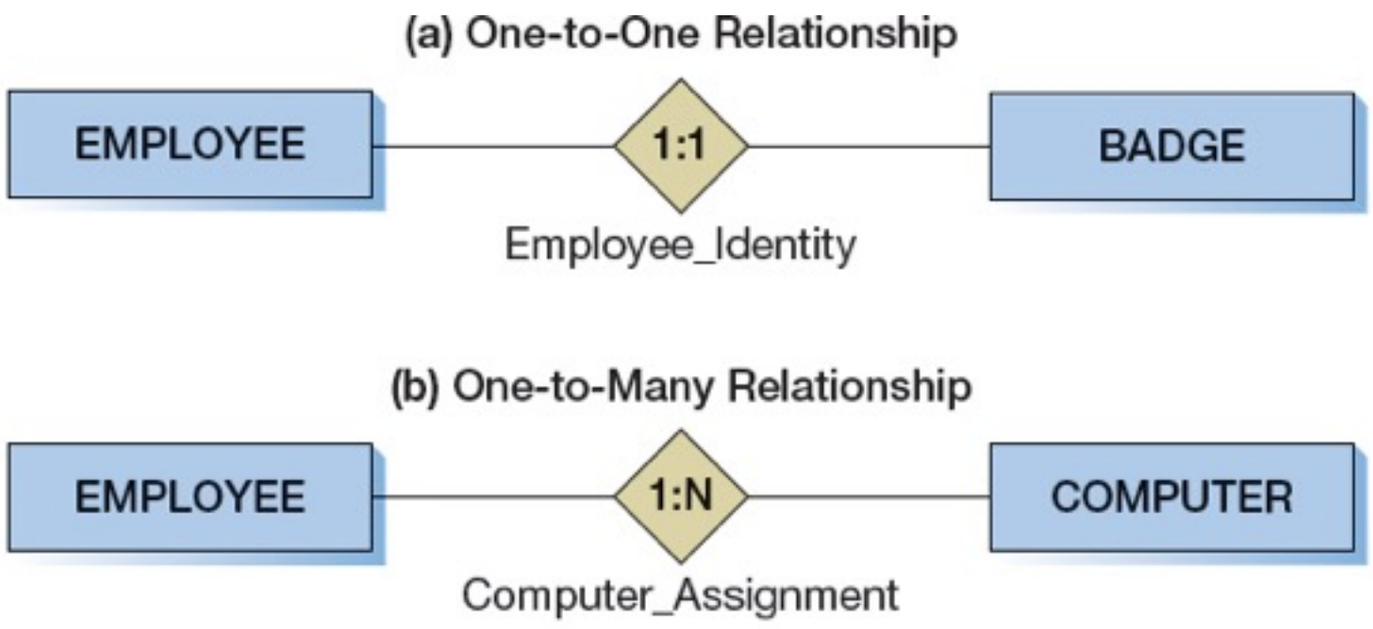
- The principle difference between an entity and a table (relation) is that you can express a relationship between entities without using foreign keys.
- This makes it easier to work with entities in the early design process where the very existence of entities and the relationships between them is uncertain.
- For example, you can say that a DEPARTMENT relates to many EMPLOYEEs before you know any of the attributes of either EMPLOYEE or DEPARTMENT.
- This characteristic enables you to work from the general to the specific.
- First, identify the entities, then think about relationships, and, finally, determine the attributes.

CARDINALITY

- Cardinality means “count,” and is expressed as a number.
- Maximum cardinality is the maximum number of entity instances that can participate in a relationship.
- Minimum cardinality is the minimum number of entity instances that must participate in a relationship.

MAXIMUM CARDINALITY

- Maximum cardinality is the maximum number of entity instances that can participate in a relationship.
- There are three types of maximum cardinality:

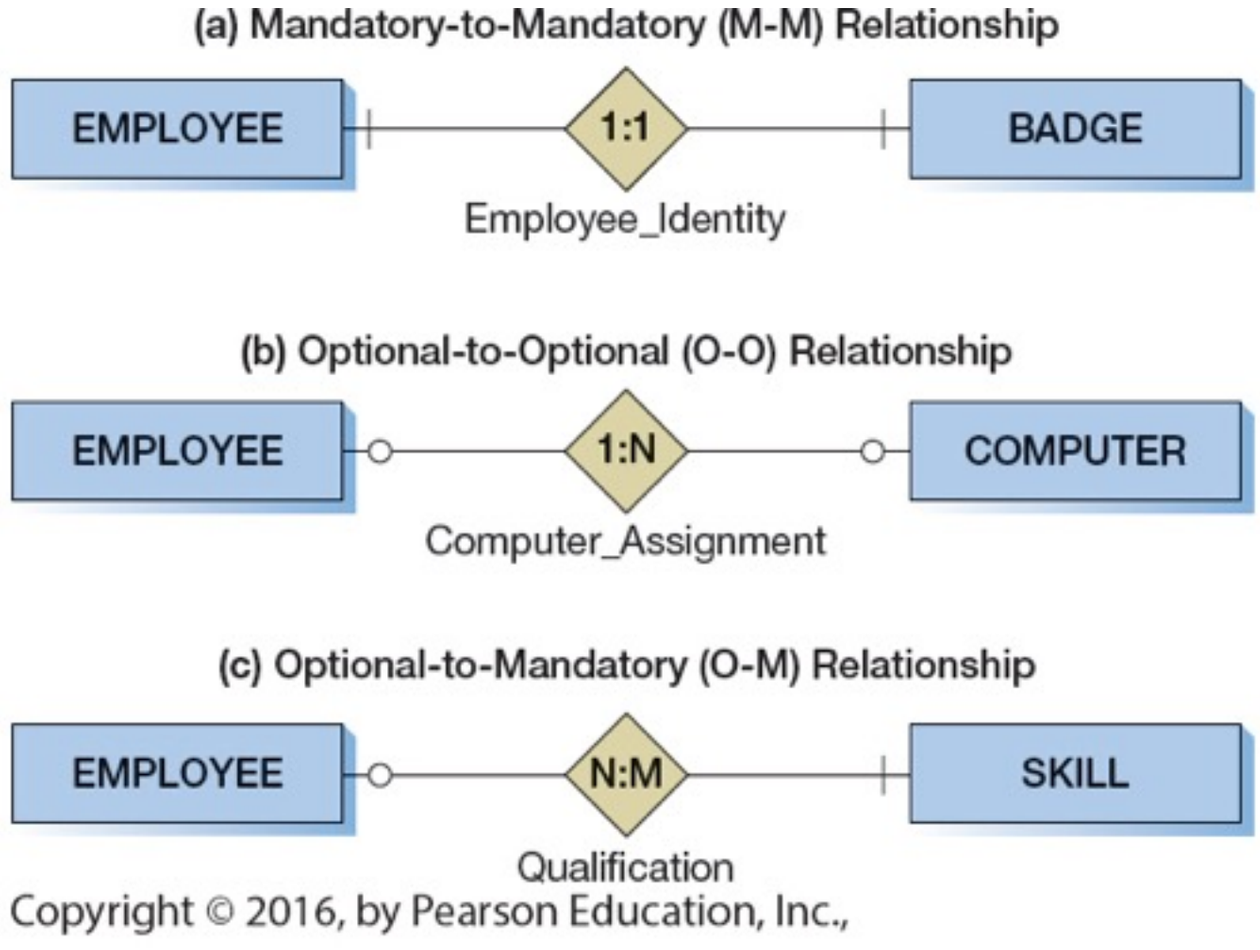


INTERPRETATION OF RELATIONSHIP

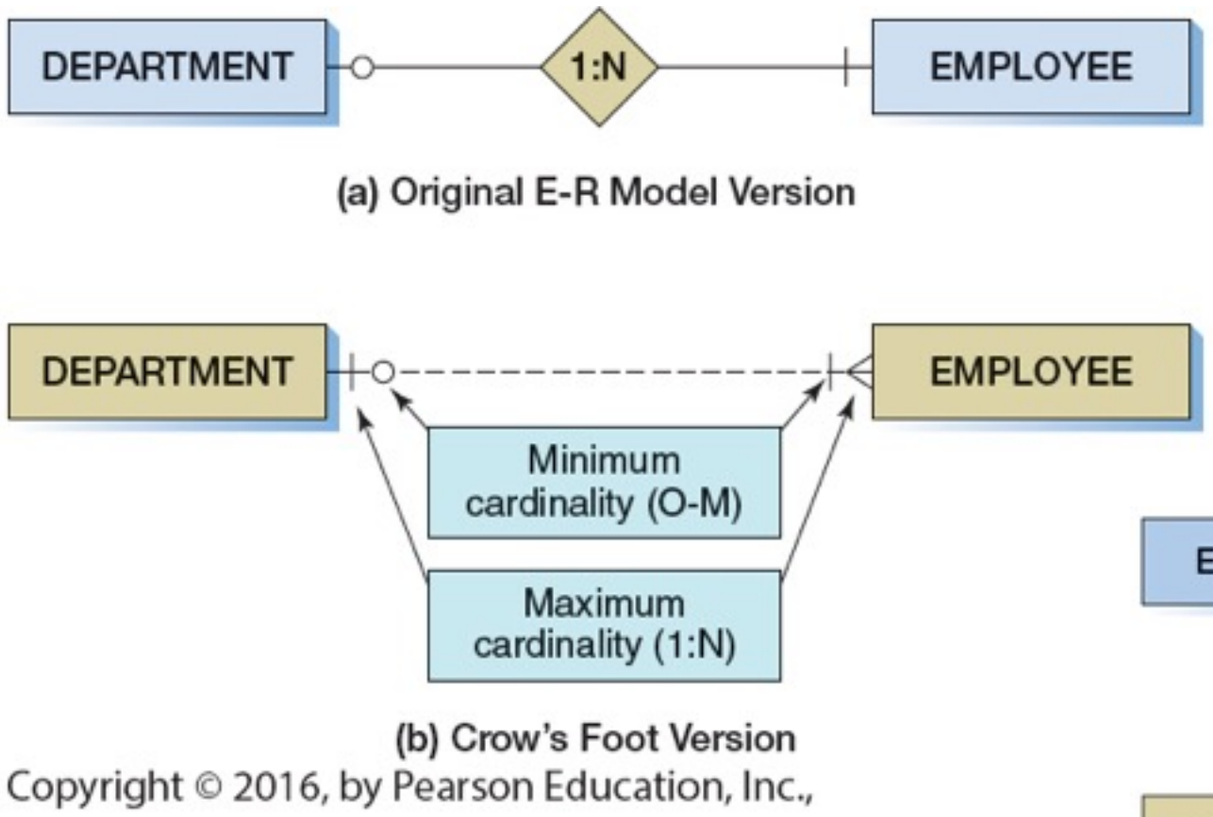
- The relationships we have been discussing are known as HAS-A relationships:
- Each entity instance has a relationship with another entity instance.
  - An EMPLOYEE has one or more COMPUTERs.
  - A COMPUTER has one assigned EMPLOYEE.
- In a one-to-many relationship:
  - The entity on the one side of the relationship is called the parent entity or just the parent.
  - The entity on the many side of the relationship is called the child entity or just the child.
  - An EMPLOYEE is the parent and a COMPUTER is the child:

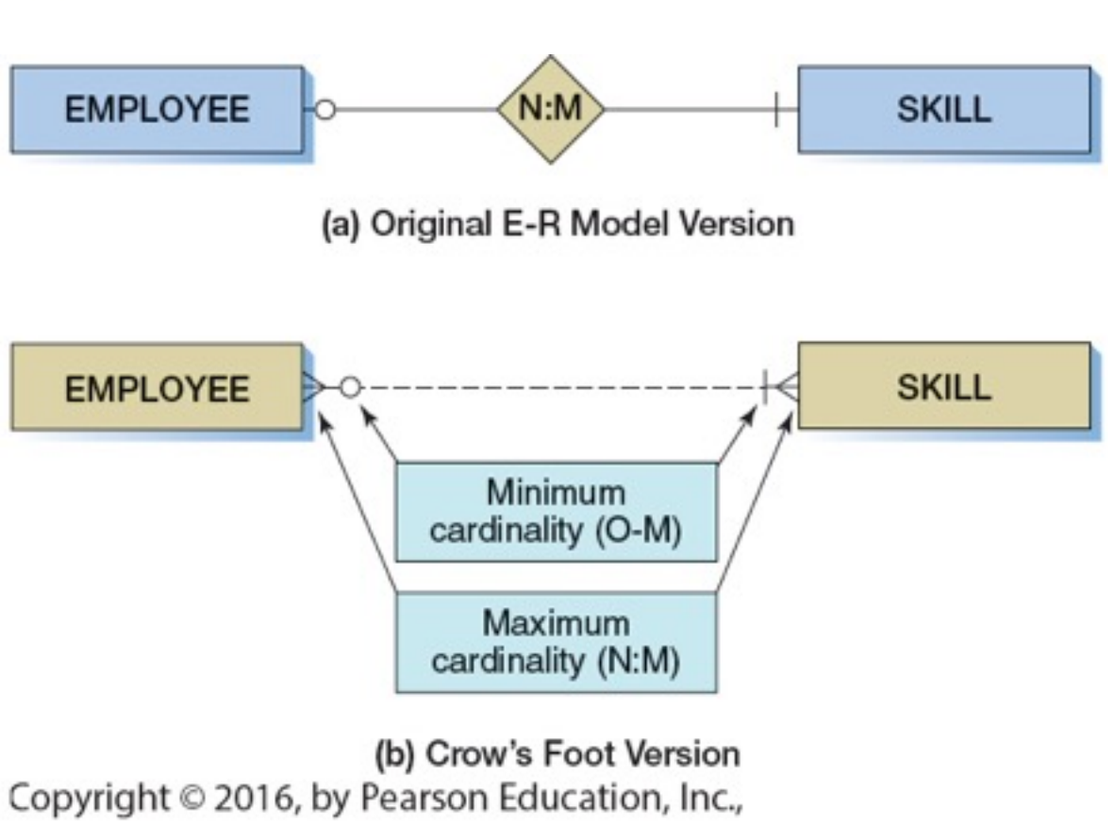
MINIMUM CARDINALITY

- Minimum cardinality is the minimum number of entity instances that must participate in a relationship.
- Minimums are generally stated as either zero or one:
  - IF zero [0] THEN participation in the relationship by the entity is **optional**, and no entity instance must participate in the relationship.
  - IF one [1] THEN participation in the relationship by the entity is **mandatory**, and at least one entity instance must participate in the relationship.



DATA MODELING NOTATION: IE CROW'S FOOT 1:N & N:M





- We will use the IE Crow's Foot model for E-R diagrams

STRONG & WEAK ENTITIES

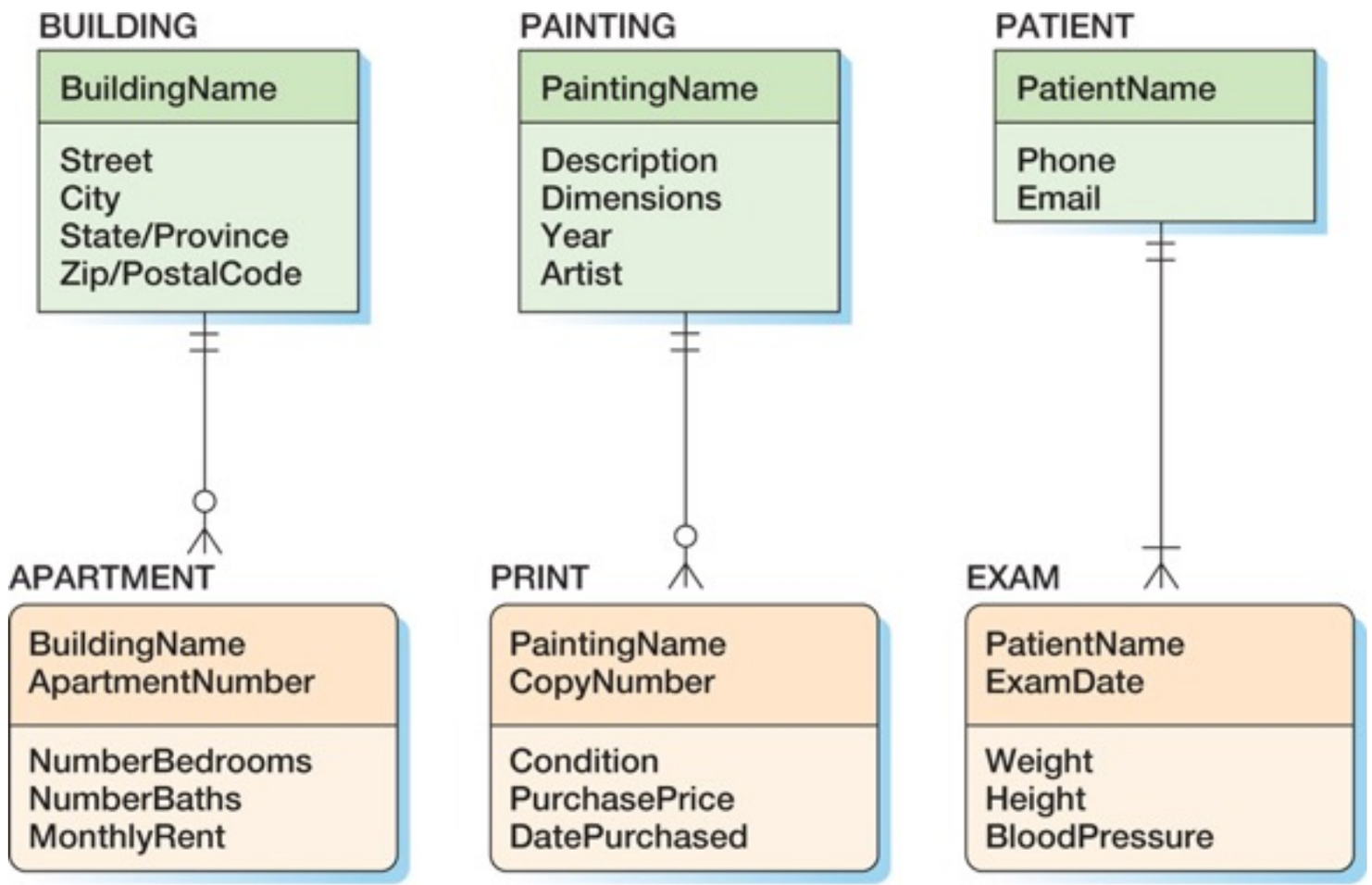
- A **strong entity** is an entity that represents something that can exist on its own.
  - For example, PERSON is a strong entity—we consider people to exist as individuals in their own right.
  - Similarly, AUTOMOBILE is a strong entity.
- A **weak entity** is defined as any entity whose existence depends on the presence of another entity.

ID-DEPENDENT ENTITIES

- An **ID-dependent entity** is an entity (child) whose identifier includes the identifier of another entity (parent).
- The minimum cardinality from the ID-dependent entity to **the parent is always one**.
- All ID-Dependent entities are considered as **a weak entity**.
- ID-dependent entities pose restrictions on the processing of the database that is constructed from them.
  - Namely, the row that represents the parent entity must be created before any ID-dependent child row can be created.
  - Further, when a parent row is deleted, all child rows must be deleted as well.
- We use an entity with **rounded corners** to represent the ID-dependent entity.
- We also use **a solid line** to represent the relationship between the ID-dependent entity and its parent, called an **identifying relationship**.

EXAMPLES OF ID-DEPENDENT ENTITIES

- The ID-dependent entity is a logical extension or subunit of the parent:
  - BUILDING : APARTMENT
  - PAINTING : PRINT
  - PATIENT : EXAM



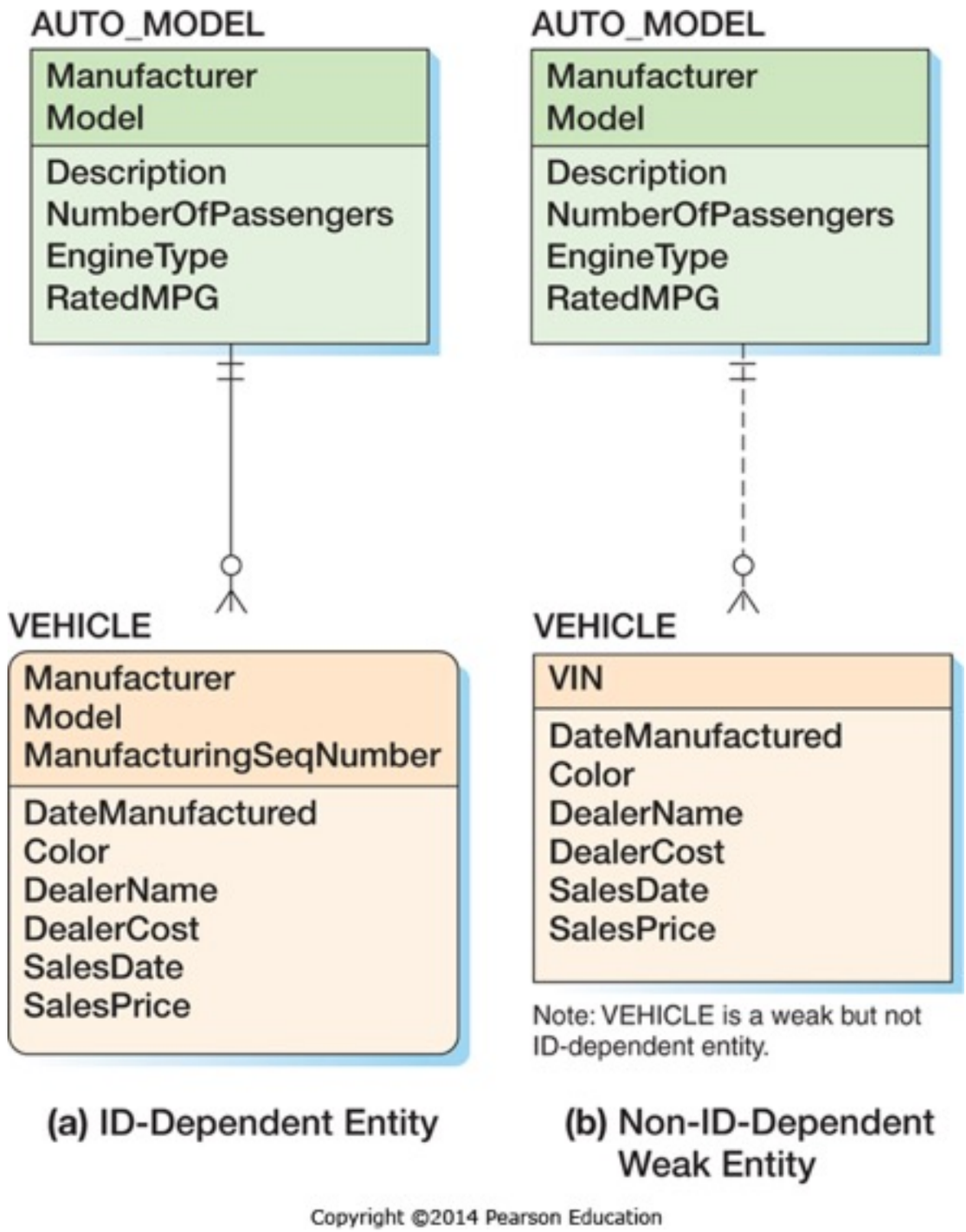
(a) APARTMENT Is ID-Dependent on BUILDING

(b) PRINT Is ID-Dependent on PAINTING

(c) EXAM Is ID-Dependent on PATIENT

NON-ID-DEPENDENT ENTITIES

- An non-ID-dependent weak entities is the identifier of the parent does not appear in the identifier of the weak child entity.
- We will use **a nonidentifying relationship** with a note added to the data model indicating that the entity is weak.

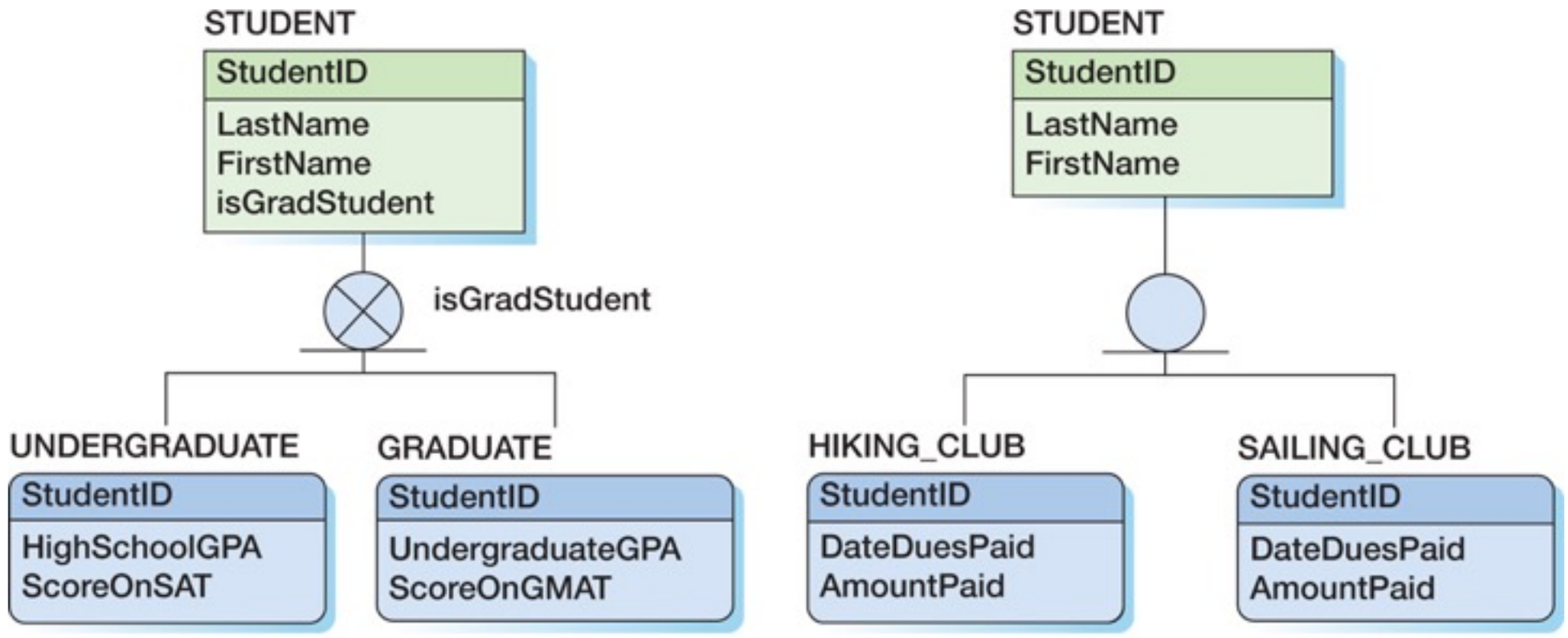


THE AMBIGUITY OF THE WEAK ENTITY

- The ambiguity is that in a strict sense, if a weak entity is defined as any entity whose presence in the database depends on another entity, then any entity that participates in a relationship having a minimum cardinality of one to a second entity is a weak entity.
- Thus, in an academic database, if a STUDENT must have an ADVISER, then STUDENT is a weak entity because a STUDENT entity cannot be stored without an ADVISER.
- A STUDENT is not physically dependent on an ADVISER (unlike an APARTMENT to a BUILDING), and a STUDENT is not logically dependent on an ADVISER.
- To avoid such situations, some people interpret the definition of weak entity more narrowly. They say that to be a weak entity an entity must logically depend on another entity.
- We agree with the latter approach.

SUBTYPE ENTITIES

- The extended E-R model introduced the concept of subtypes.
  - For example, Students may be classified as undergraduate or graduate students.
- A subtype entity is a special case of another entity called its supertype.
  - In this case, STUDENT is the supertype, and UNDERGRADUATE and GRADUATE are the subtypes.
- The most important (some would say the only) reason for creating subtypes in a data model is to avoid value-inappropriate nulls.



(a) Exclusive Subtypes with Discriminator

(b) Inclusive Subtypes