

Databases for Analytics

Kroenke / Auer
Chapter 5

Learning Objectives

- **Skills:** You should know how to ...
 - Draw and interpret entity relationship diagram
- **Theory:** You should be able to explain ...
 - The purpose and process of data modeling
 - Entities, attributes, relationships, identifiers, and cardinalities, ID dependencies, etc.
 - The graphical syntax and structure of an ERD
 - The relationship between ERDs and table designs

Data Models

For when you are starting with a clean sheet of paper

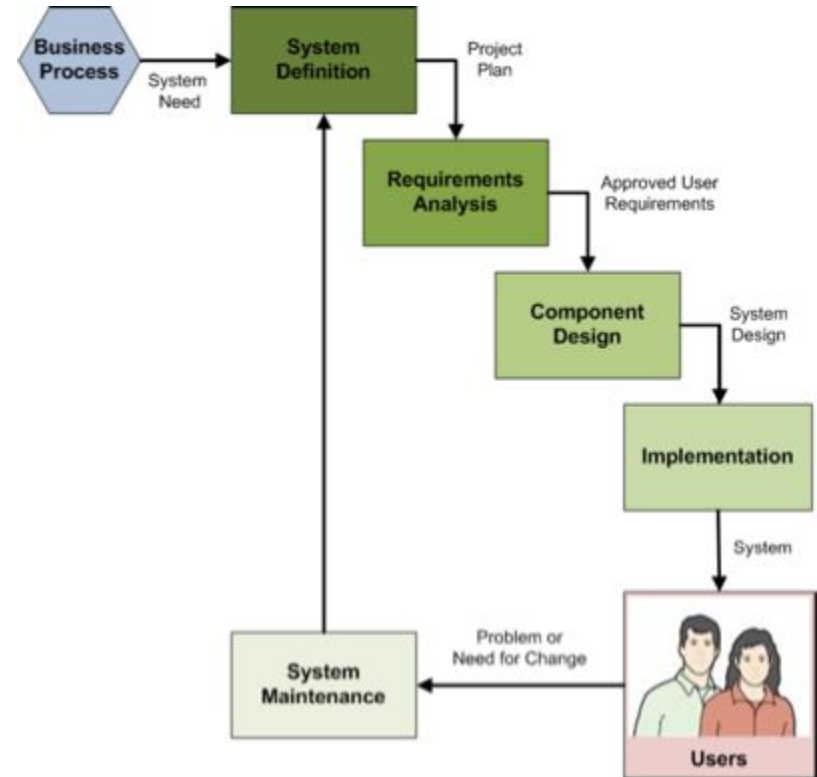
Data Modeling

- 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 than it is to change a database design, so it is the appropriate place to work through conceptual database problems.

Data Modeling in the SDLC

Data modeling **primarily** resides in the Requirements Analysis phase.

However, the data model will continue to change throughout the SDLC as the overall system design evolves.



Data Modeling as Conceptual Design

- 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 **conceptual** design.
- Table designs (and SQL DDL) are logical design.
- The live database in the DBMS is physical design.

ER Models

Entities, Attributes, and Relationships

Entity-Relationship Model

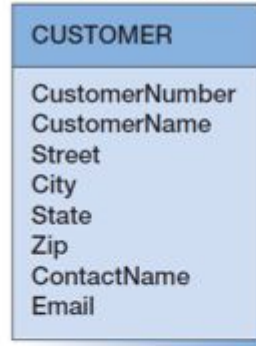
- Based on the idea that in order to capture data about *things*, you need to model the *things*
- Lots of ERM standards:
 - **Original E-R model** by [Peter Chen](#) (1976)
 - **Extended E-R model**: extensions to the Chen model
 - **Information Engineering (IE)** by James Martin (1990); uses “crow’s foot” notation, is easier to understand, and **we will use it**
 - **IDEF1X**: a national standard developed by the National Institute of Standards and Technology [see Appendix C]
 - **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; the class defines a schema/blueprint/structure for all entities of the given type
 - **Entity instance**: the occurrence of a particular entity; an instance has actual data
- There are usually many instances of an entity in an entity class.

Entity Class vs Entity Instance

CUSTOMER Entity



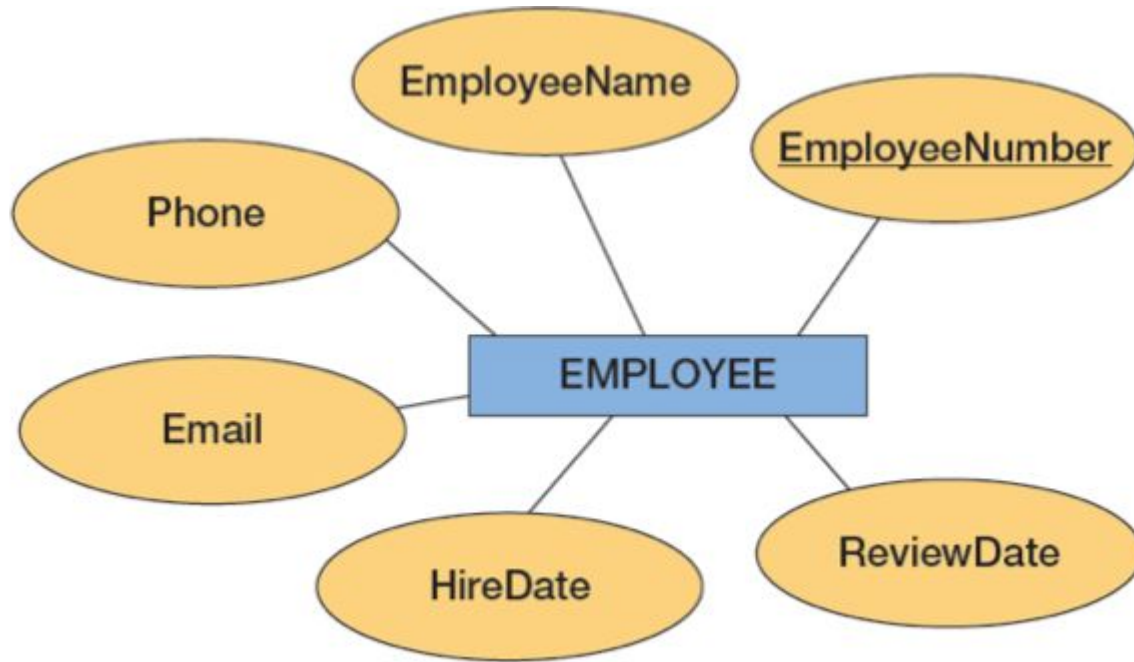
Two CUSTOMER Instances



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 **elliptical shapes**.
- Data modeling products today commonly show attributes in **rectangular form**.

Original Chen-style Attributes



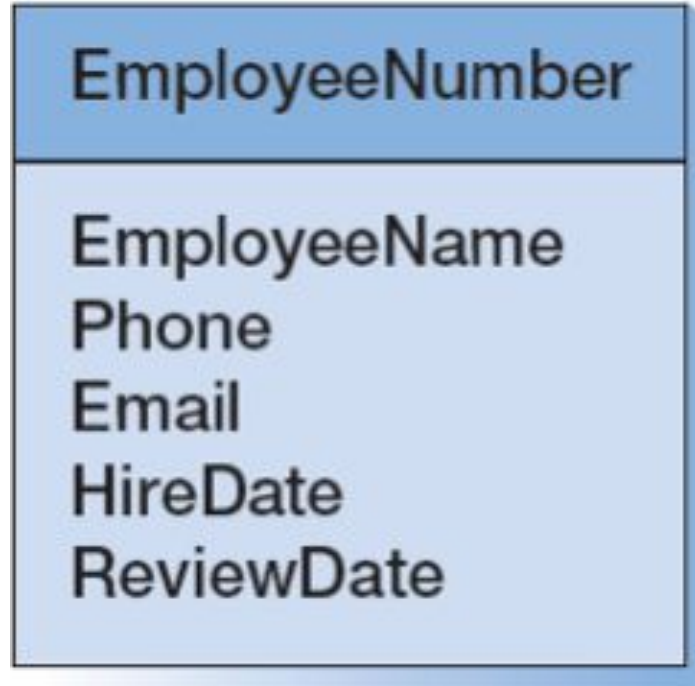
Modern Style Attributes (in Textbook)

The class name is above the rectangle.

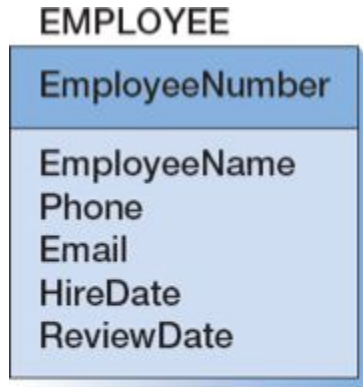
The identifier attribute (PK) is at the top of the box.

The other attributes are in the bottom of the box.

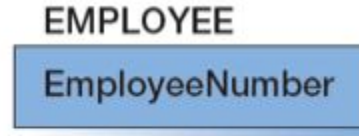
EMPLOYEE



Varying Levels of Detail ...



Entity with All
Attributes



Entity with Identifier
Attribute Only



Entity with No
Attributes

Entities (in most software packages)

Entity	
<u>Identifier-Field</u>	
Field	
Field	

Entity	
Key	Field
Key	Field
Key	Field

Entity		
Key	Field	Type
Key	Field	Type
Key	Field	Type

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.

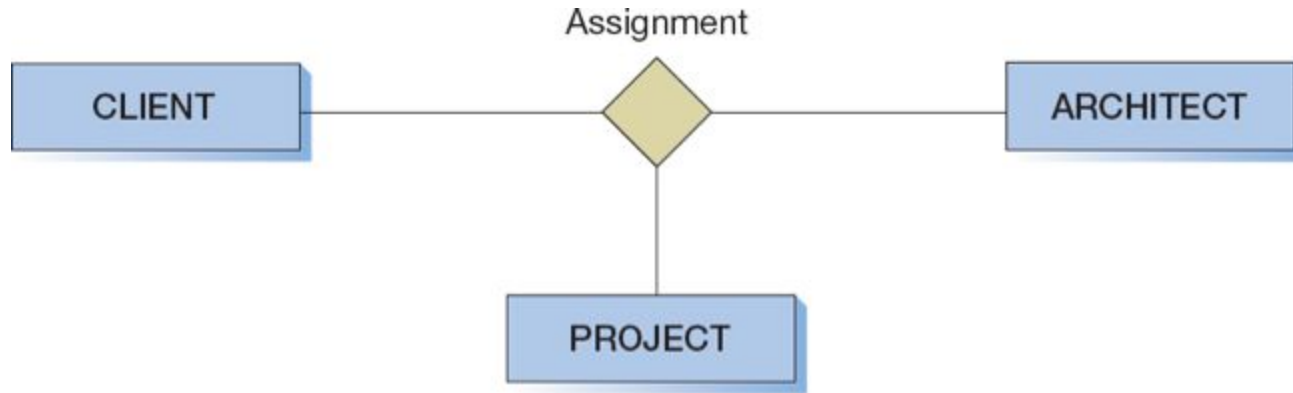
Degrees of Relationship

- 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.
- In practice, the relational model supports only binary relationships. Tertiary and higher-order relationships must be converted to multiple binary relationships (PK/FK pairs).

Binary Relationship



Tertiary Relationship



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.

Relationship Cardinalities

It's all in the counting

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.

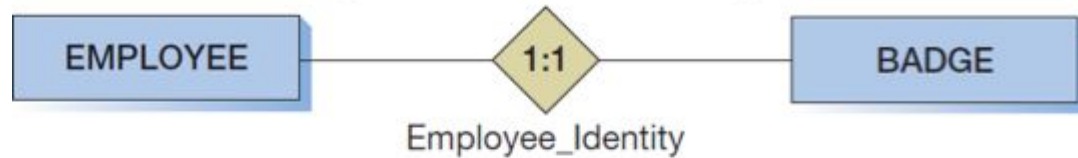
Maximum Cardinalities

There are three types of maximum cardinality:

- One-to-One [1:1]
- One-to-Many [1:N]
- Many-to-Many [N:M]

Examples

(a) One-to-One Relationship



(b) One-to-Many Relationship



(c) Many-to-Many Relationship



Parent and Child Entities

- 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**.
- In the figure below, EMPLOYEE is the parent and COMPUTER is the child:



HAS-A 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.

Minimum Cardinalities

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.

Indicating Minimum Cardinalities

- oval → entity is optional, with a minimum cardinality of 0
- vertical hash mark → entity is mandatory (required), with a minimum of 1

(a) Mandatory-to-Mandatory (M-M) Relationship



(b) Optional-to-Optional (O-O) Relationship




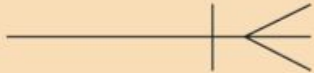


(c) Optional-to-Mandatory (O-M) Relationship



Crow's Foot Notation

Shows maximum and minimum cardinalities at each end of a relationship.

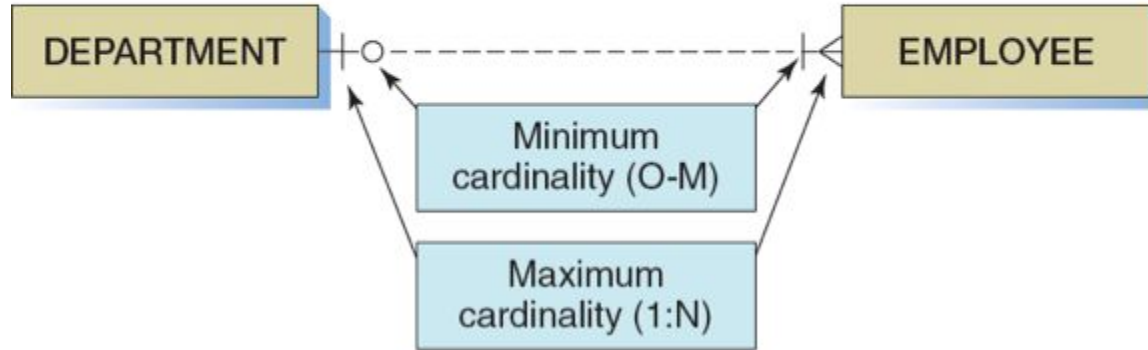
A crow's foot means maximum cardinality > 1 .

Symbol	Meaning	Numeric Meaning
	Mandatory—One	Exactly one
	Mandatory—Many	One or more
	Optional—One	Zero or one
	Optional—Many	Zero or more

Original Chen vs Crow's Foot



(a) Original E-R Model Version



(b) Crow's Foot Version

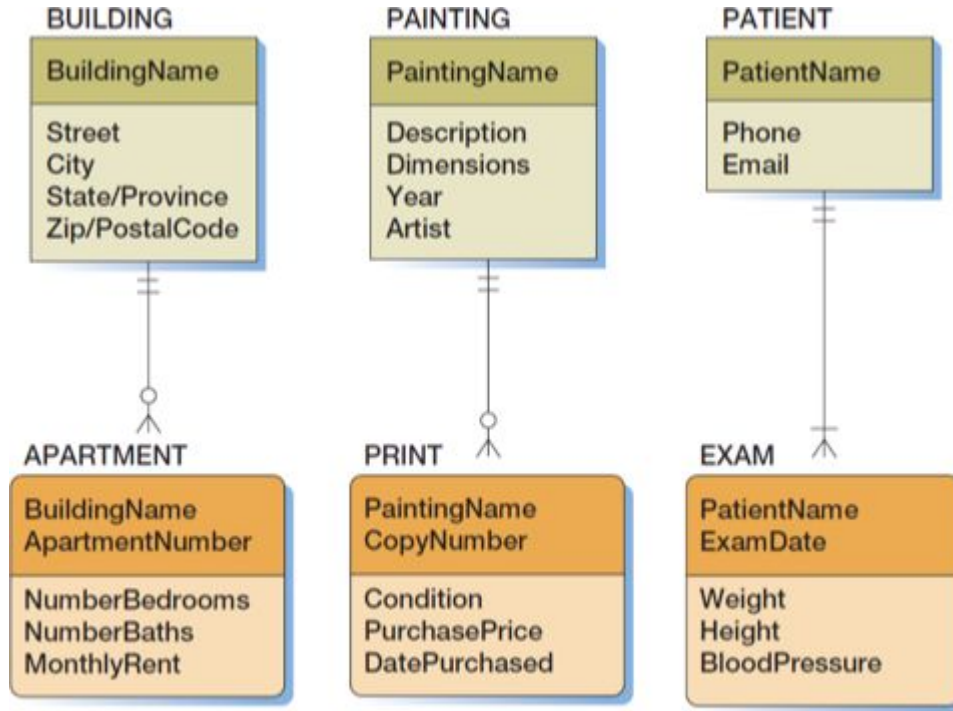
Entities and Identifiers

Which comes first, the identifier or the entity?

ID-Dependent Entities

- An **ID-dependent entity** is an entity (child) whose identifier includes the identifier of another entity (parent).
- The ID-dependent entity is a **logical extension** or subunit of the parent:
 - BUILDING ← APARTMENT
 - PAINTING ← PRINT
- The **minimum cardinality** from the ID-dependent entity to the parent **is always one**.

ID-Dependency Examples



(a) APARTMENT is
ID-Dependent on
BUILDING

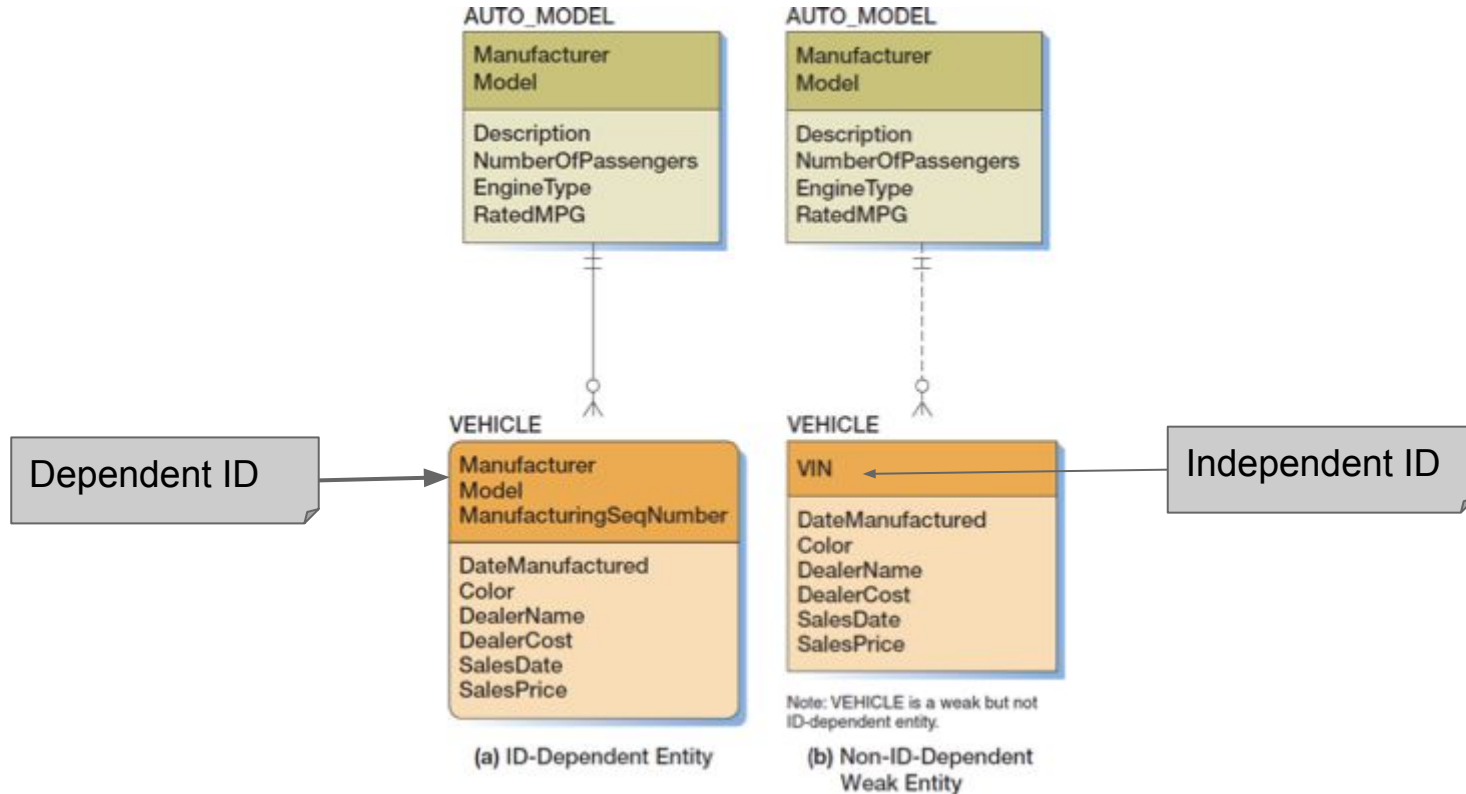
(b) PRINT is
ID-Dependent
on PAINTING

(c) EXAM is
ID-Dependent
on PATIENT

Weak Entities

- A **weak entity** is an entity whose existence depends upon another entity.
- All **ID-Dependent** entities are considered weak.
- There are also **non-ID-dependent** weak entities.
 - The identifier of the parent does **not** appear in the identifier of the weak child entity.
 -

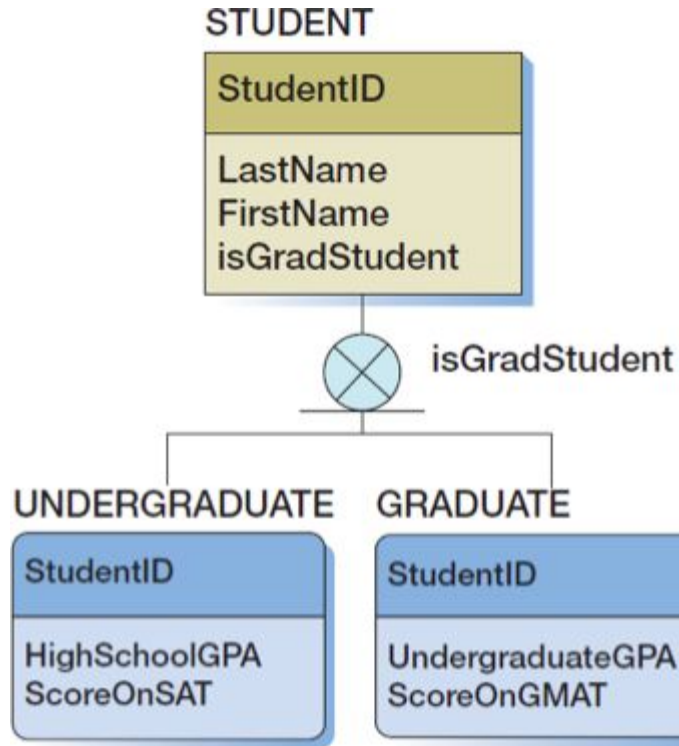
Weak Entity Examples



Subtype Entities (IS-A Relationships)

- A **subtype entity** is a special case of a supertype entity:
 - UNDERGRADATE and GRADUATE are subtypes of STUDENT
 - An UNDERGRADUATE **is a** STUDENT
- The supertype contains all common attributes, while the subtypes contain specific attributes.
- The supertype may have a discriminator attribute which indicates the subtype.

Subtypes with Discriminators

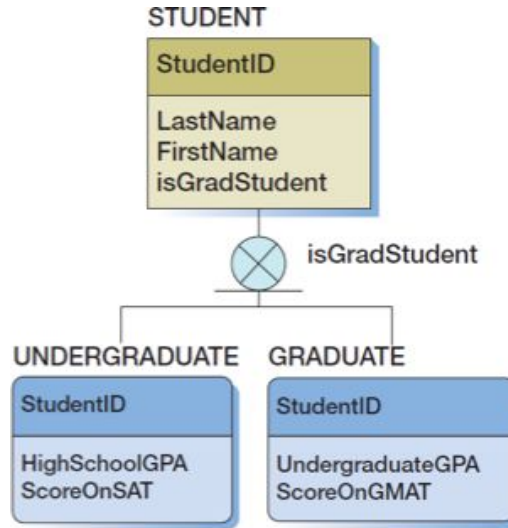


Discriminators are rarely shown these days. It's more common to use UML class notation instead.

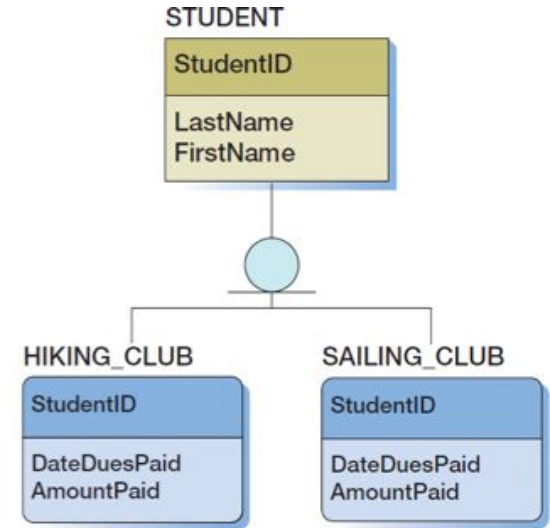
Inclusive vs Exclusive Subtypes

Can subtypes overlap?

- **Exclusive:** one supertype relates to at most one subtype
- **Inclusive:** one supertype can relate to multiple subtypes.



(a) Exclusive Subtypes with Discriminator



(b) Inclusive Subtypes

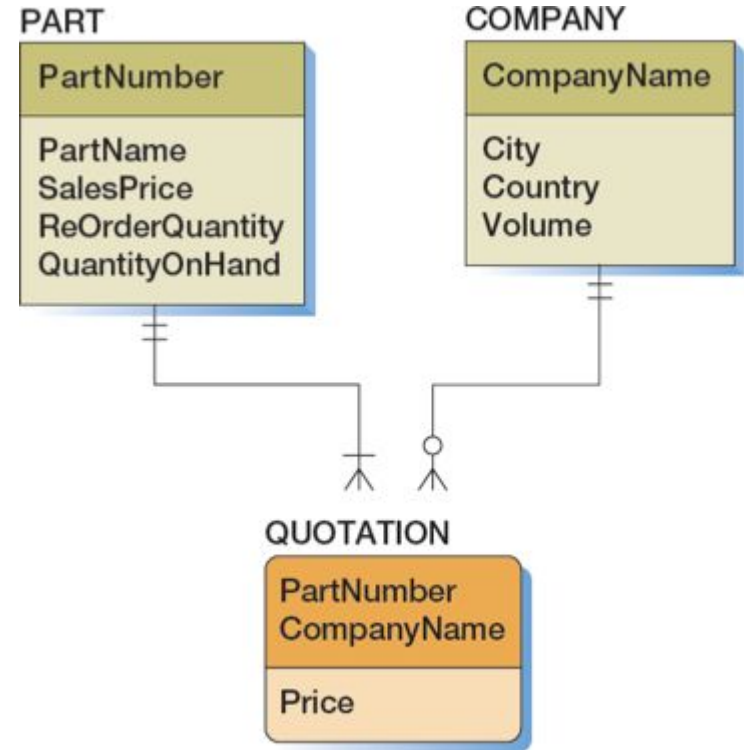
Do we need Subtypes?

- Relationships connecting supertypes and subtypes are called **IS-A relationships**, because a subtype is a supertype.
- The identifier of the supertype and all of its subtypes must be **identical**; i.e., the identifier of the supertype becomes the identifier of the related subtype(s).
- Subtypes are used to avoid **value-inappropriate nulls**.

Associative Entities

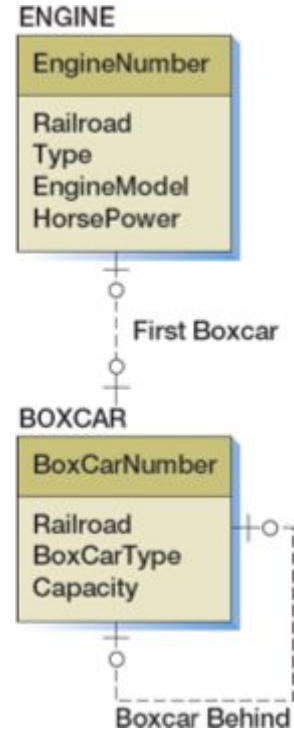
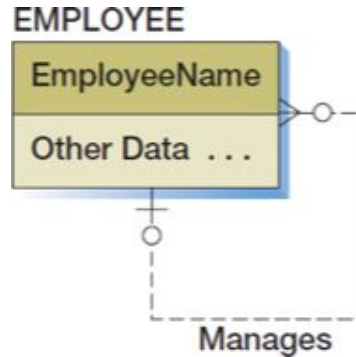
An entity that holds one or more additional attributes beyond the identifiers of the parent entities is called an **associative entity** (or association entity).

In this case Price is the additional attribute.



Recursive Relationships

A **recursive relationship** occurs when an entity class has a relationship to itself.



Design Process

Highline University example from the book

Highline University case

- Suppose the administration at a hypothetical university named **Highline University** wants to create a database to track colleges, departments, faculty, and students.
- To do this, a data modeling team has collected a series of reports as part of its requirements determination.

The College Report

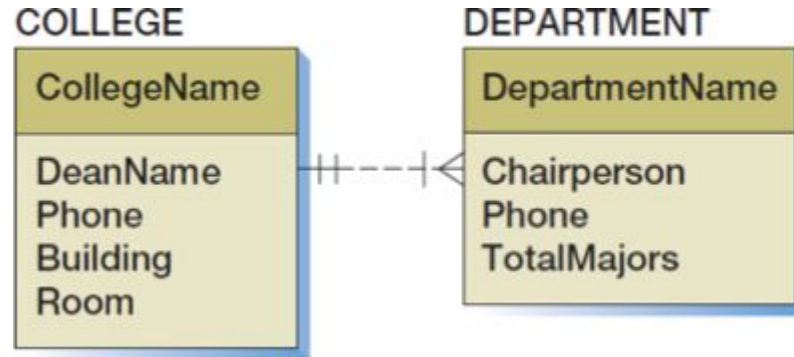
College of Business
Mary B. Jefferson, Dean

Phone: 232-1187

Campus Address:
Business Building, Room 100

<u>Department</u>	<u>Chairperson</u>	<u>Phone</u>	<u>Total Majors</u>
Accounting	Jackson, Seymour P.	232-1841	318
Finance	HeuTeng, Susan	232-1414	211
Info Systems	Brammer, Nathaniel D.	236-0011	247
Management	Tuttle, Christine A.	236-9988	184
Production	Barnes, Jack T.	236-1184	212

First Data Model



The Department Report

Information Systems Department College of Business

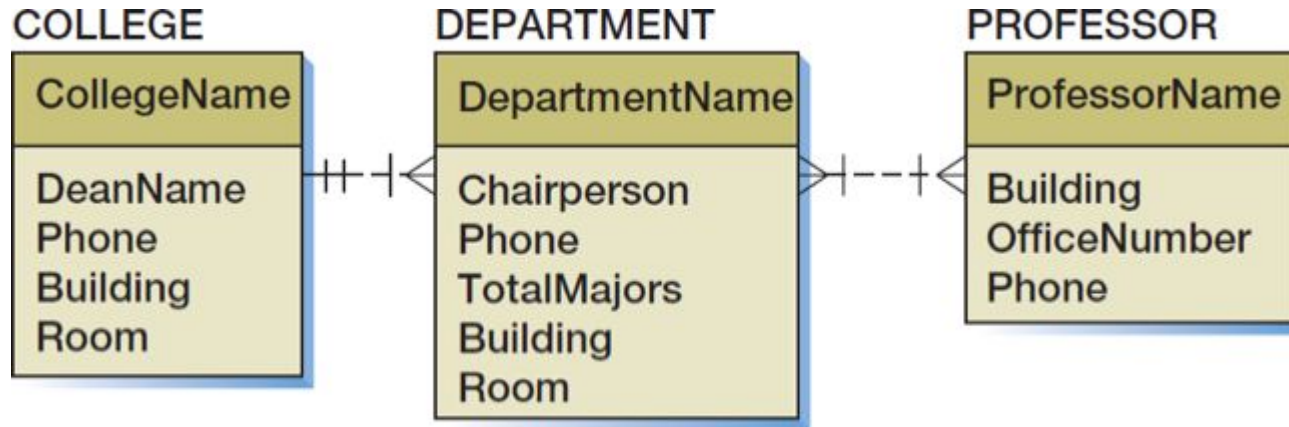
Chairperson: Brammer, Nathaniel D

Phone: 236-0011

Campus Address: Social Science Building, Room 213

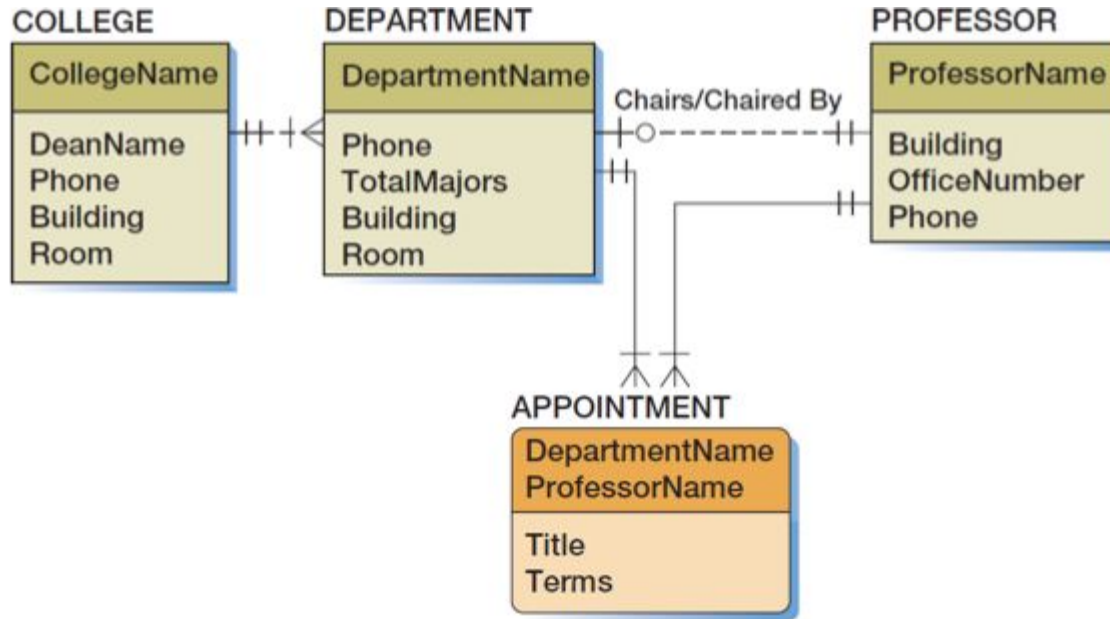
<u>Professor</u>	<u>Office</u>	<u>Phone</u>
Jones, Paul D.	Social Science, 219	232-7713
Parks, Mary B	Social Science, 308	232-5791
Wu, Elizabeth	Social Science, 207	232-9112

Second Data Model



(a) Data Model Using an N:M Relationship

Third Data Model



(d) Data Model Using an Association Pattern and 1:1 Relationship

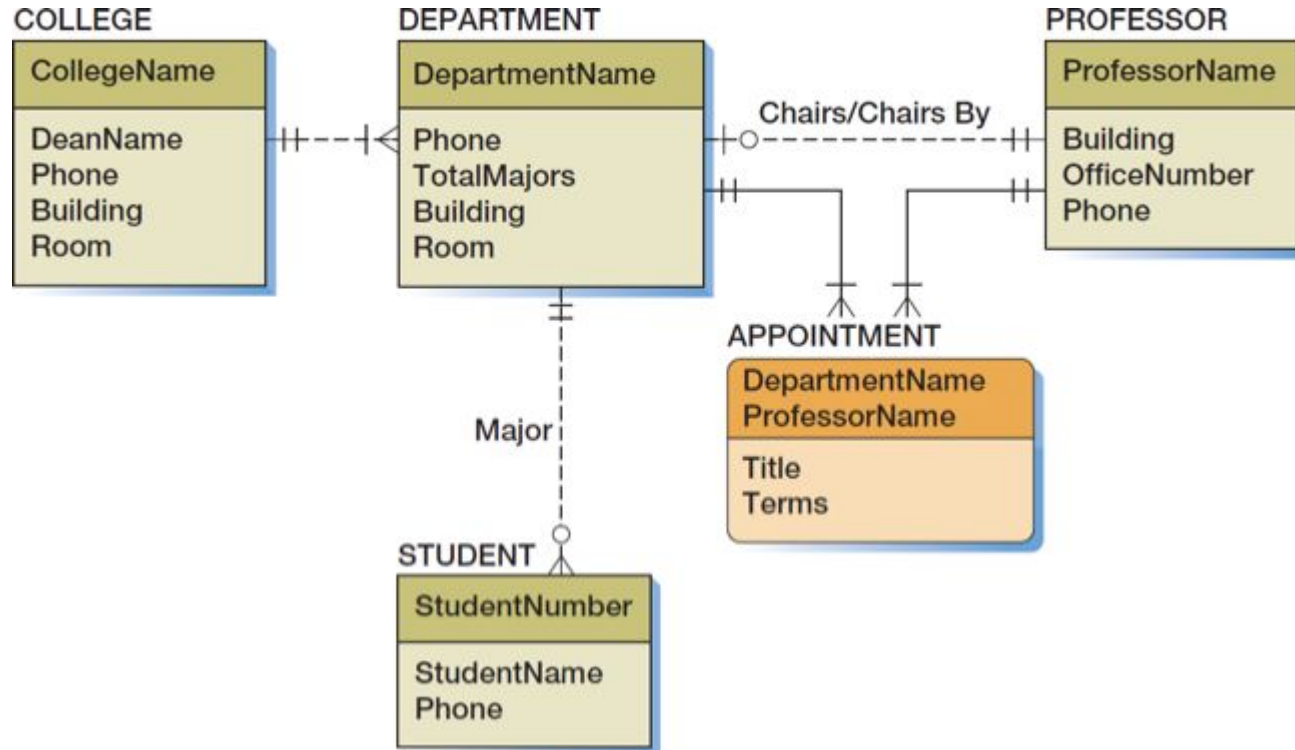
The Department Student Report

Student Major List Information Systems Department

Chairperson: Brammer, Nathaniel D Phone: 236-0011

<u>Major's Name</u>	<u>Student Number</u>	<u>Phone</u>
Jackson, Robin R.	12345	237-8713
Lincoln, Fred J.	48127	237-8924
Madison, Janice A.	37512	237-9035

Fourth Data Model



Acceptance Letter

Mr. Fred Parks
123 Elm Street
Los Angeles, CA 98002

Dear Mr. Parks:

You have been admitted as a major in the **Accounting** Department at Highline University, starting in the Fall Semester, 2015. The office of the Accounting Department is located in the **Business Building, Room 210**.

Your adviser is professor **Elizabeth Johnson**, whose telephone number is 232-8740 and whose office is located in the **Business Building, Room 227**. Please schedule an appointment with your adviser as soon as you arrive on campus.

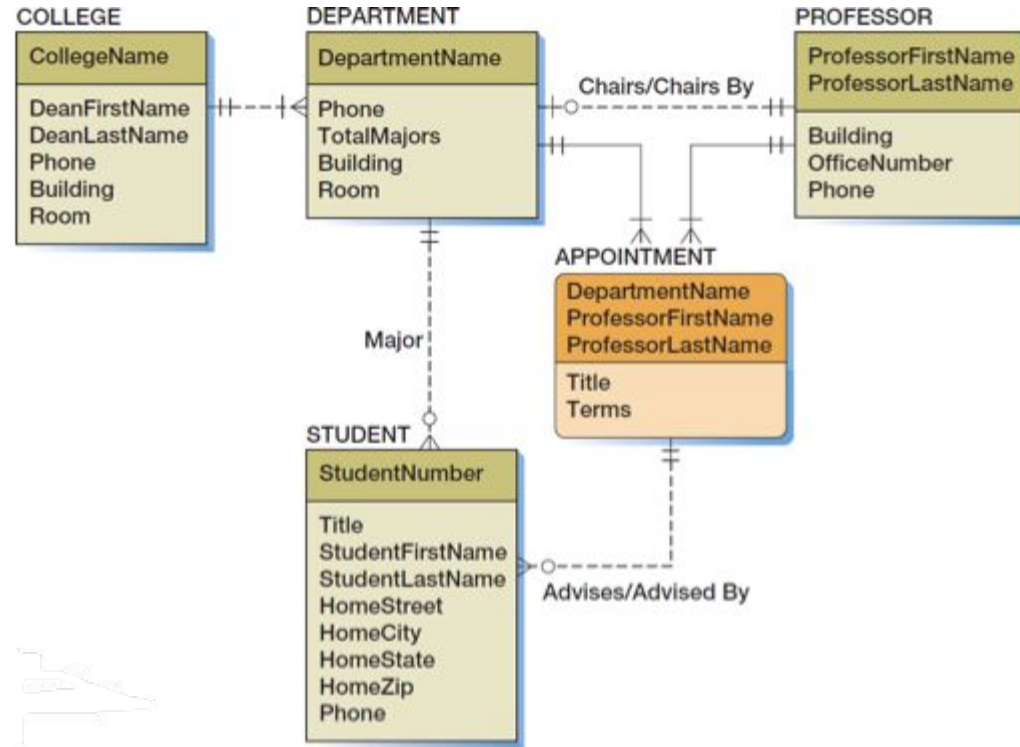
Congratulations and welcome to Highline University!

Sincerely,

Jan P. Smathers
President

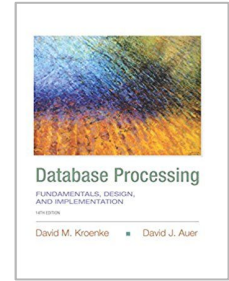
JPS/rkp

Final Data Model



More About Design Process

What the book left out



Databases for Analytics

Kroenke / Auer
Chapter 5