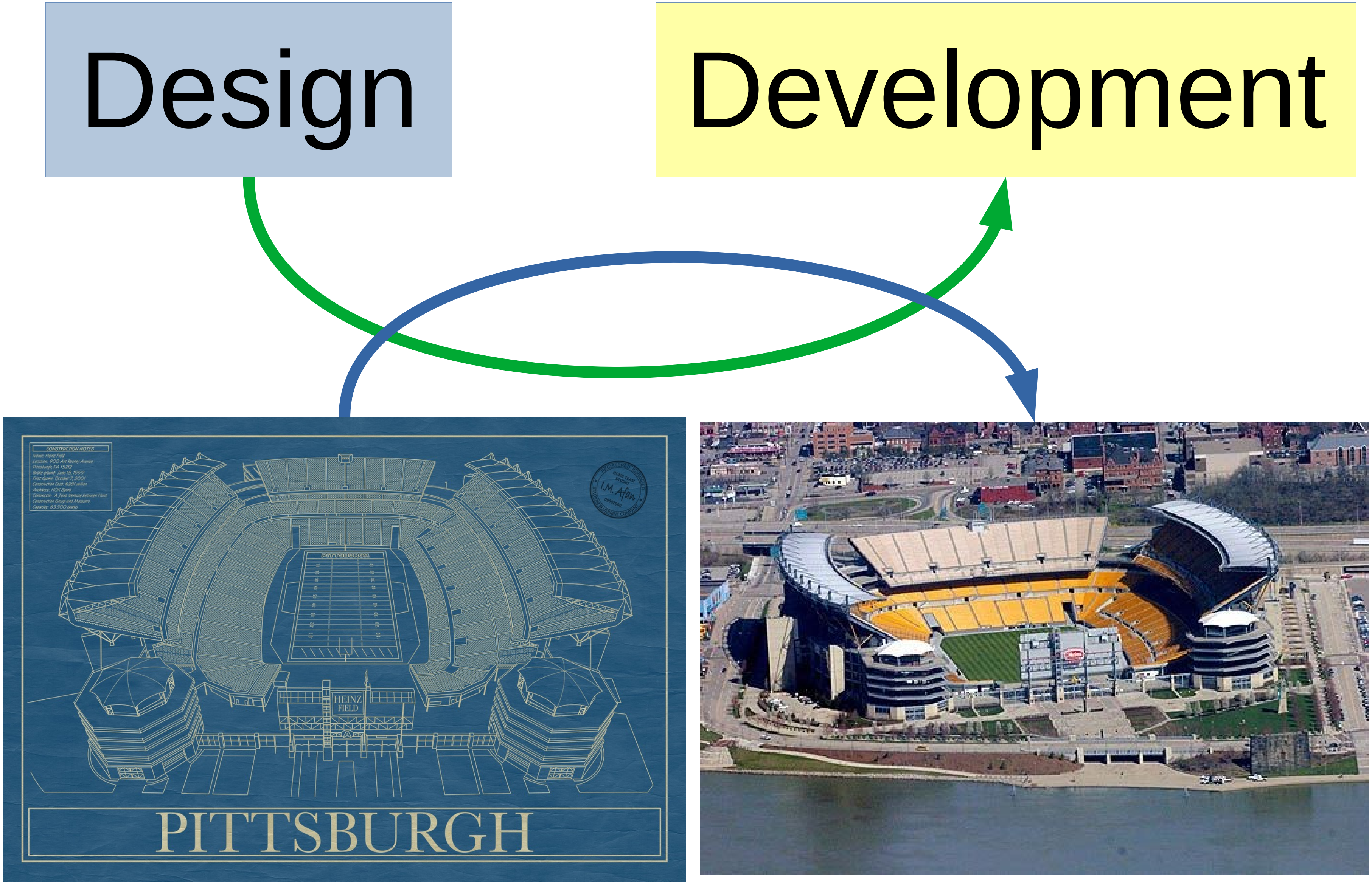
Principles of Database Systems
CS 312
Tables and Schemas

23 Sept 2020

Oliver BONHAM-CARTER

Database Development



Design

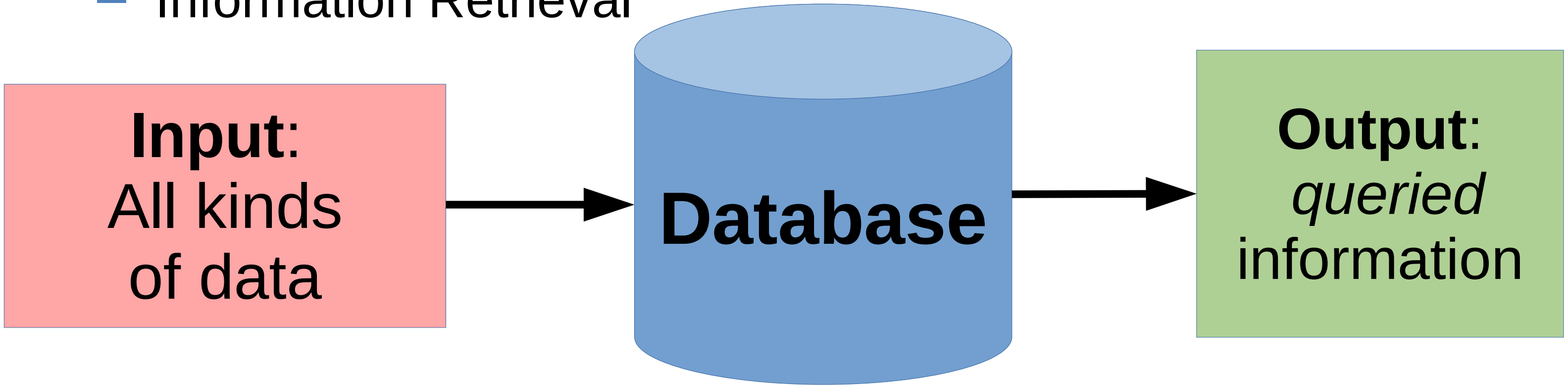
Development

Database Development

How do we *think* about designing a database?

- Steps for development:
  - Analysis of Usage
    - What is the **purpose** of the db?
  - Design of system
  - Construction
    - relationships
    - programming
  - Data Entry
  - Information Retrieval

With some help from Uncle Jeff Bezos (CEO of Amazon), book seller and computer guru extraordinaire...



Input: All kinds of data

Database

Output: *queried* information

Let's Imagine: Our Client

Fantastic Books

The Client Says...

I WANT EVERYTHING!!!

I need data about... Customers!
Wholesalers! Sales! Purchases!
Employees! Receipts!
And reports! Lots of reports!
And profits!!!

FYI: Clients **never** know exactly what they need!

The Challenge:

We are to change the current “system”…

The Grand Ambition!

Fantastic Books

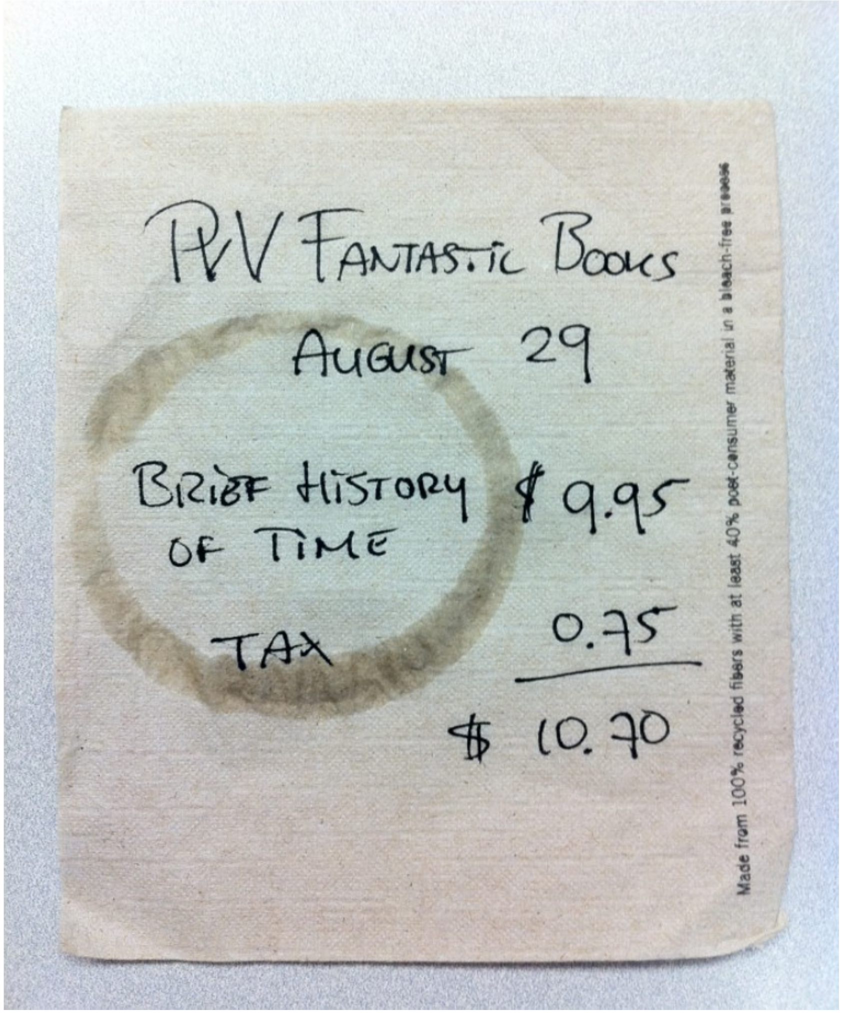
J. Smith
1234 PKI Road
Omaha, NE 68123

September 15, 2013

Zero History	1	$ 15.95
Anathem	1	8.90
The Hobbit	1	14.25
The Subtle Knife	1	7.30
		========
Total		$ 46.40
Tax		3.48
		========
Total Due		$ 49.88
Cash Payment		$ 49.88

Exchange only w/ receipt in 7 days.
Sign up for our newsletter
at pvv-fantasticbooks.com

The Sad Reality



PVV FANTASTIC BOOKS

AUGUST 29

BRIEF HISTORY OF TIME	$ 9.95
TAX	0.75
	$ 10.70

Made from 100% recycled fibers with at least 40% post-consumer material in a bleach-free process

What information do I have?

What data is available?
What is this database supposed to do?
What reports are important to make?
What kinds of queries?

Users?
Software?
Hardware?
How populated?
How queried?

Maintenance?
Centralized?
Networked?
Points of use?

Entities & Attributes *Version 1.0*

Customer: first name, last name, address, city, state, zip, phone number, e-mail address

Employee: first name, last name, address, city, state, zip, phone number, e-mail address

Book: ISBN, title, author(s), publisher, publication date, genre, number in stock, number on order

Sale: date, customer first name, last name, ISBN, title, quantity sold, total due, price paid, payment method, employee first name, last name

Purchase: date, wholesaler name, ISBN title, quantity bought, total due, amount paid,

Wholesaler: wholesaler name, address, city, state, zip, phone number, account number, website, e-mail address

How to differentiate rows, tables?

Entities & Attributes *Version 2.0*

Customer: customer-ID, first name, last name, address, city, state, zip, phone number, e-mail address

Employee: employee-ID, first name, last name, address, city, state, zip, phone number, e-mail address

Book: ISBN, title, author(s), publisher, publication date, genre, number in stock, number on order

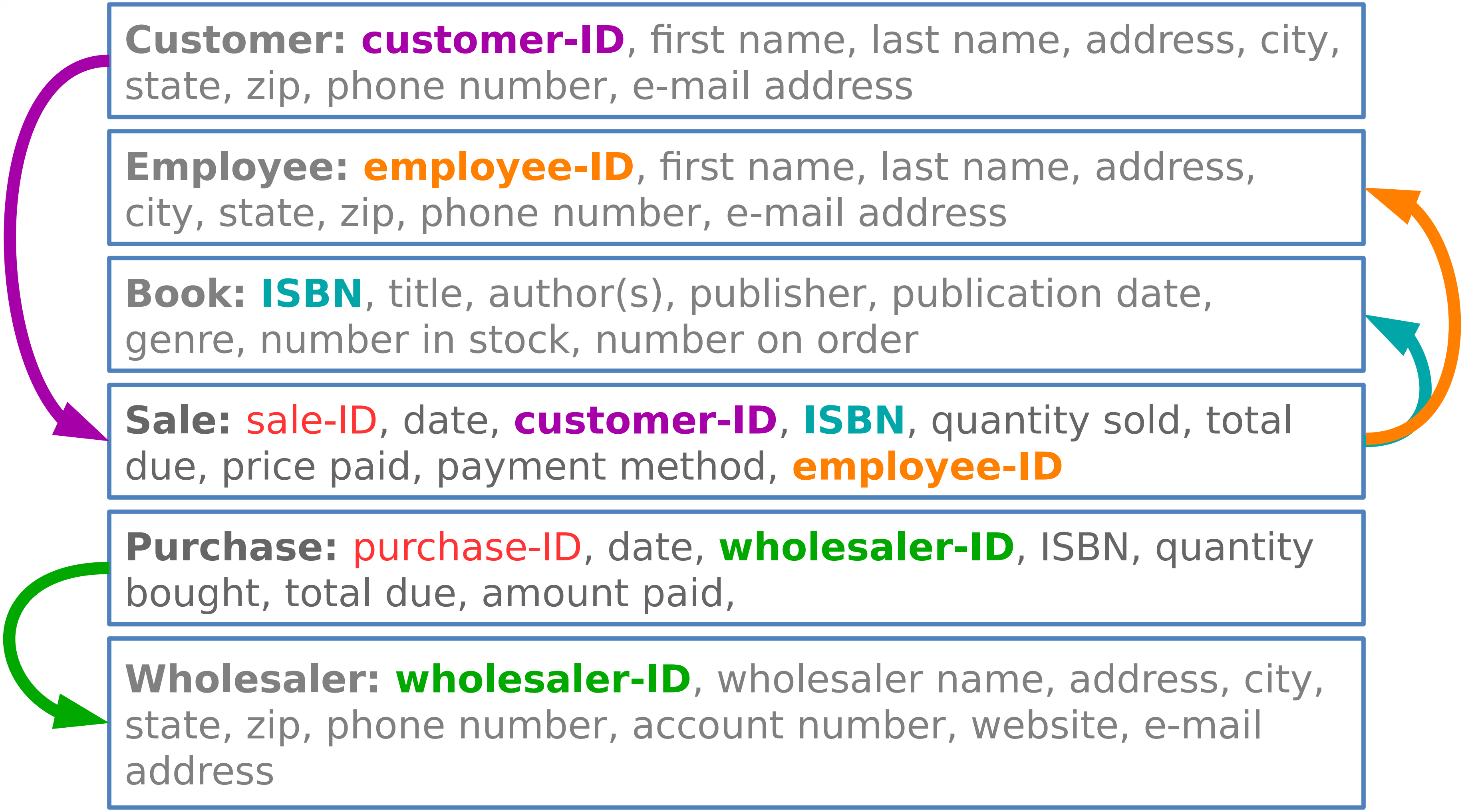
Sale: sale-ID, date, customer first name, last name, ISBN, title, quantity sold, total due, price paid, payment method , employee first name, last name

Purchase: purchase-ID, date, wholesaler name, ISBN, title, quantity bought, total due, amount paid,

Wholesaler: wholesaler-ID, wholesaler name, address, city, state, zip, phone number, account number, website, e-mail address

What about linking your tables?

Entities & Attributes *Version 3.0*



Customer: **customer-ID**, first name, last name, address, city, state, zip, phone number, e-mail address

Employee: **employee-ID**, first name, last name, address, city, state, zip, phone number, e-mail address

Book: **ISBN**, title, author(s), publisher, publication date, genre, number in stock, number on order

Sale: sale-ID, date, **customer-ID**, **ISBN**, quantity sold, total due, price paid, payment method, **employee-ID**

Purchase: purchase-ID, date, **wholesaler-ID**, ISBN, quantity bought, total due, amount paid,

Wholesaler: **wholesaler-ID**, wholesaler name, address, city, state, zip, phone number, account number, website, e-mail address

Are you sure that there is
no redundancy in your tables?

Entities & Attributes *Version 4.0*

Customer: customer-ID, first name, last name, address, city, state, zip, phone number, e-mail address

Employee: employee-ID, first name, last name, address, city, state, zip, phone number, e-mail address

Book: ISBN, title, publisher-ID, publication date, genre, number in stock, number on order

Sale: sale-ID, date, customer-ID, total due, price paid, payment method, employee-ID

Purchase: purchase-ID, date, wholesaler-ID, total due, amount paid,

Wholesaler: wholesaler-ID, wholesaler name, address, city, state, zip, phone number, account number, website, e-mail address

Publisher: **Publisher**-ID, publisher name, address, city, state, zip, phone number

Special case: connect Books to Authors

Author: Author-ID, first name, last name

Sale-Line: sale-ID, ISBN, quantity sold

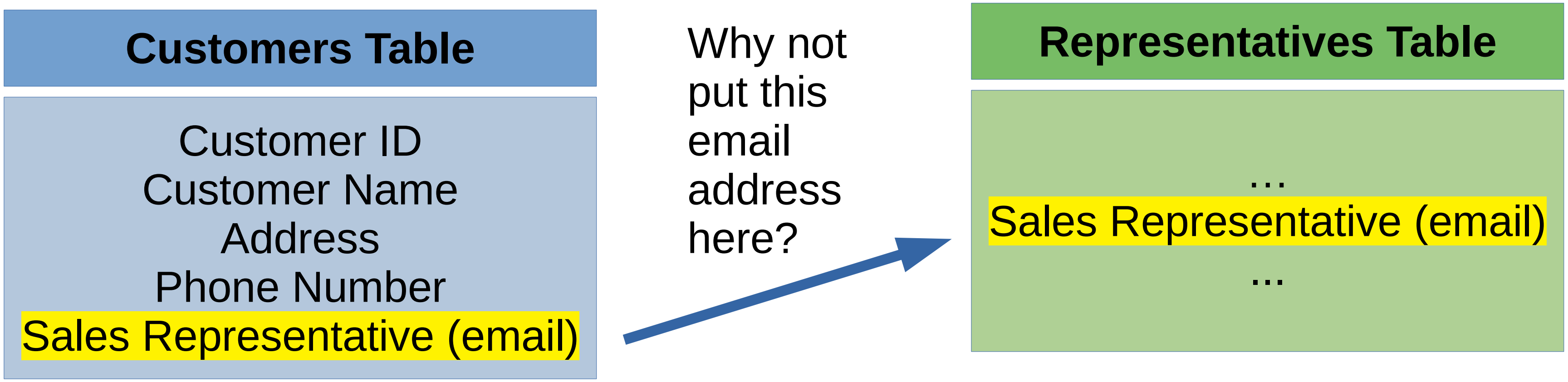
Purchase-Line: Purchase-ID, ISBN, quantity bought

I like the connectivity.
Now, what about
the ***normalization***?

“Normalization” of a Base

***Normalization**:*
*The process of organizing data in a database. This includes creating tables and establishing **relationships** between them according to rules designed both to protect the data and to make the database more flexible by eliminating redundancy and inconsistent dependency.*

Inconsistant Dependancies



Customers Table

Customer ID
Customer Name
Address
Phone Number
Sales Representative (email)

Why not put this email address here?

Representatives Table

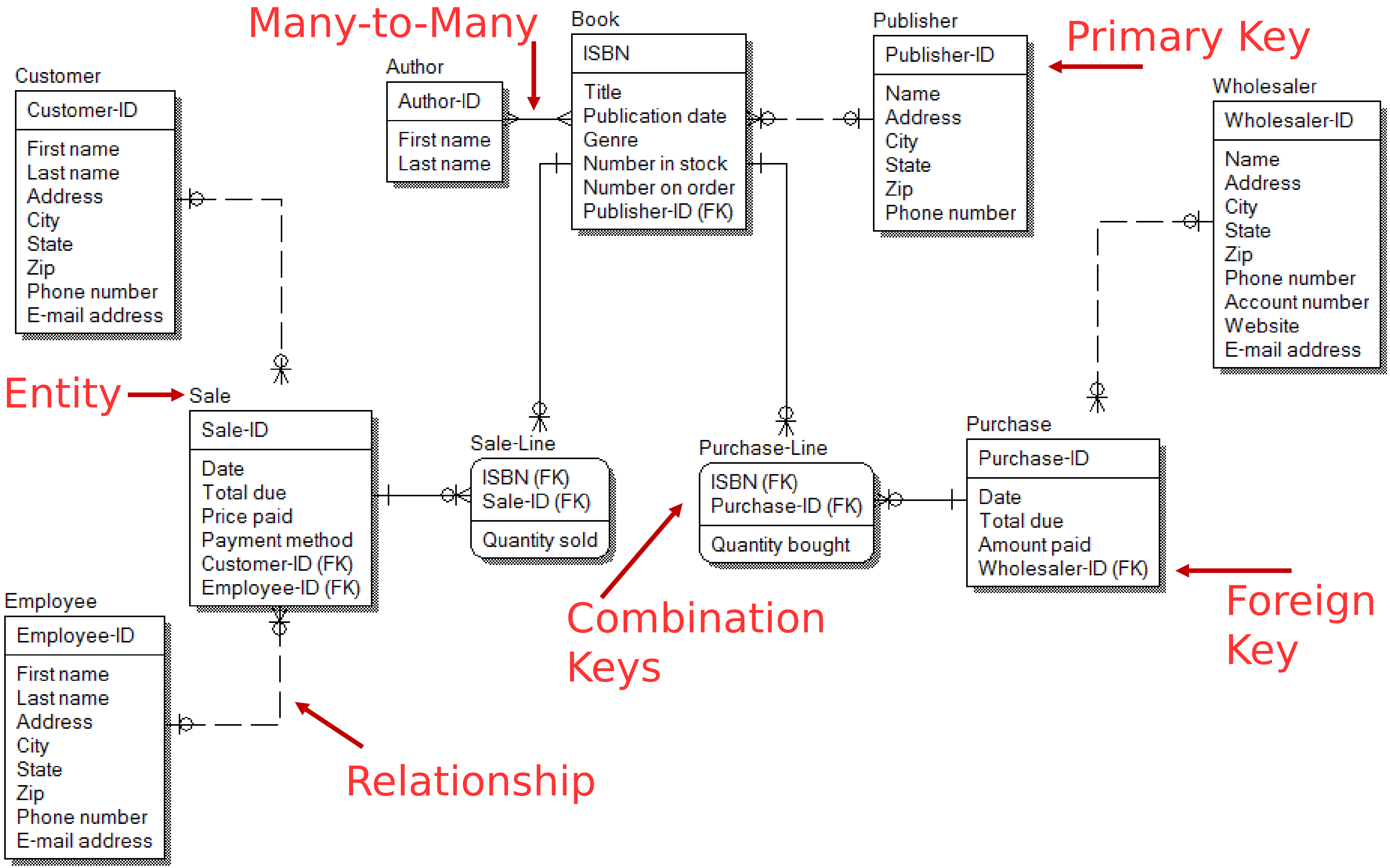
…
Sales Representative (email)
…

- Each of the fields in a table should have a *dependency* on the table (because it logically belongs there and **nowhere else**.)

- A sales representative (the person who visits the customer and generates sales) does interact with the customer and so this person’s name belongs in the **Customers Table** but we only really need a link to this person in the **Representatives Table** **(and not the name itself)**.

- If the representative’s email (or other information concerning representatives) appeared in the "**Customers Table**" table, then we would say that there was an "**inconsistent dependency**."

Entity Relationship Diagram



Many-to-Many

Primary Key

Entity

Combination Keys

Foreign Key

Relationship

Customer
- Customer-ID
- First name
- Last name
- Address
- City
- State
- Zip
- Phone number
- E-mail address

Author
- Author-ID
- First name
- Last name

Book
- ISBN
- Title
- Publication date
- Genre
- Number in stock
- Number on order
- Publisher-ID (FK)

Publisher
- Publisher-ID
- Name
- Address
- City
- State
- Zip
- Phone number

Wholesaler
- Wholesaler-ID
- Name
- Address
- City
- State
- Zip
- Phone number
- Account number
- Website
- E-mail address

Sale
- Sale-ID
- Date
- Total due
- Price paid
- Payment method
- Customer-ID (FK)
- Employee-ID (FK)

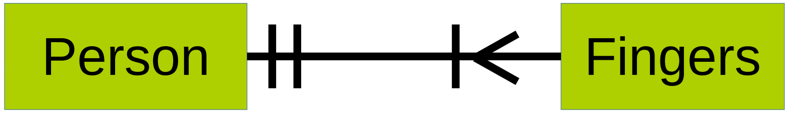
Sale-Line
- ISBN (FK)
- Sale-ID (FK)
- Quantity sold

Purchase-Line
- ISBN (FK)
- Purchase-ID (FK)
- Quantity bought

Purchase
- Purchase-ID
- Date
- Total due
- Amount paid
- Wholesaler-ID (FK)

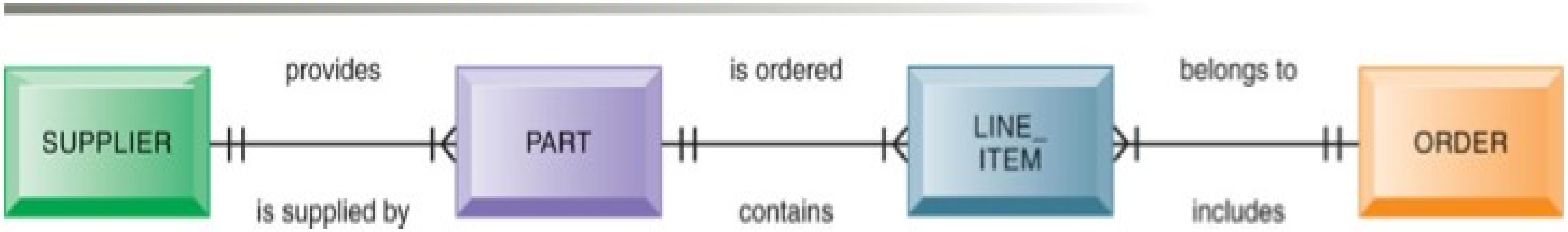
Employee
- Employee-ID
- First name
- Last name
- Address
- City
- State
- Zip
- Phone number
- E-mail address

The *One to Many* Relationship



Each person has
many fingers,
But each finger belongs
to only one person.

FIGURE 6-11 AN ENTITY-RELATIONSHIP DIAGRAM



Note: Other books use double-strikes, *greater-than* and *less-than* notation to denote these relationships

The *One to Many* Relationship

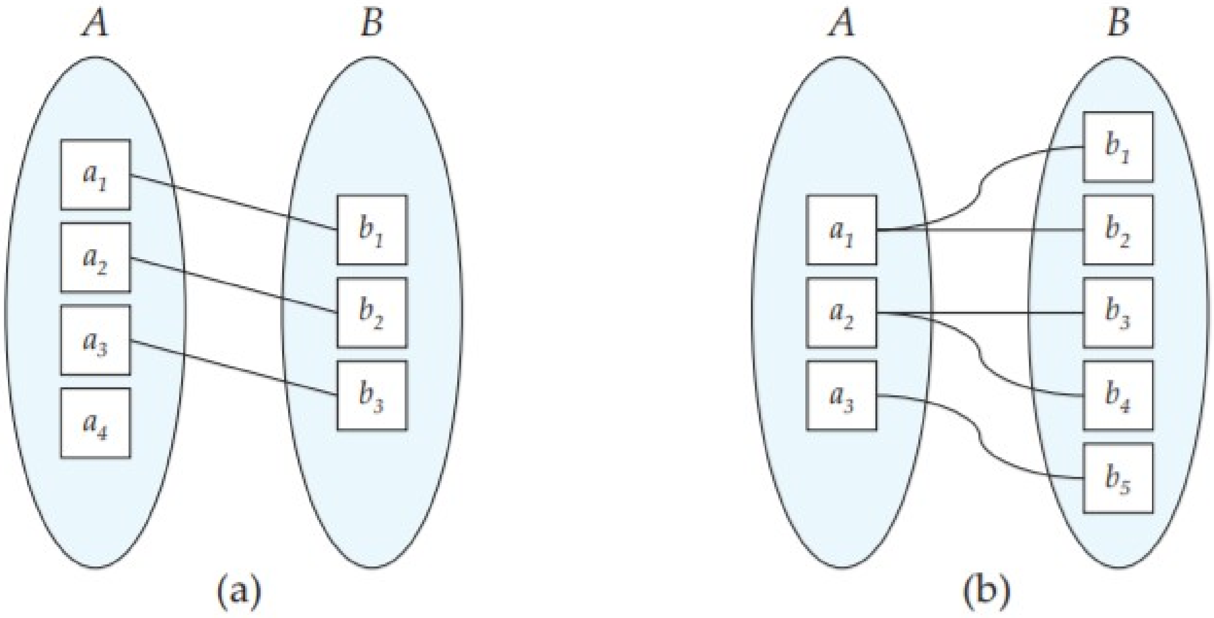


Figure 7.5 Mapping cardinalities. (a) One-to-one. (b) One-to-many.

- **One-to-one**. An entity in A is associated with at most one entity in B, and an entity in B is associated with at most one entity in A. (See left Figure.)
- **One-to-many**. An entity in A is associated with any number (zero or more) of entities in B. An entity in B, however, can be associated with at most one entity in A. (See right Figure)

The Many to Many Relationship

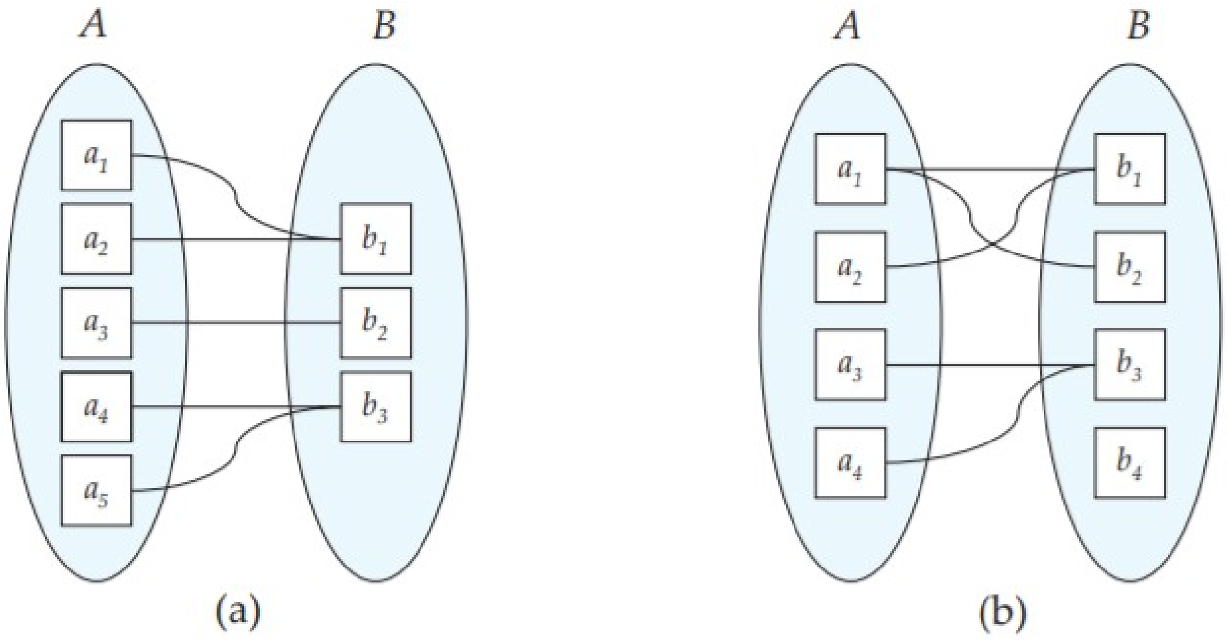


Figure 7.6 Mapping cardinalities. (a) Many-to-one. (b) Many-to-many.

- **Many-to-one**. An entity in A is associated with at most one entity in B. An entity in B, however, can be associated with any number (zero or more) of entities in A. (See left Figure)
- **Many-to-many**. An entity in A is associated with any number (zero or more) of entities in B, and an entity in B is associated with any number (zero or more) of entities in A. (See right Figure)

The Many to Many Relationship

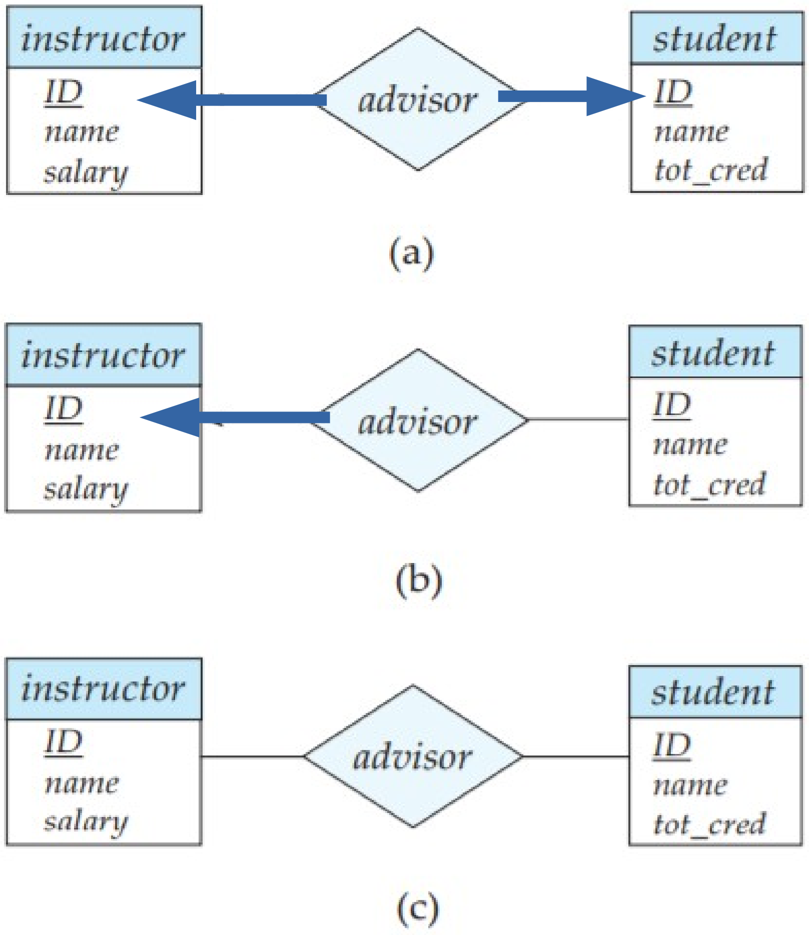
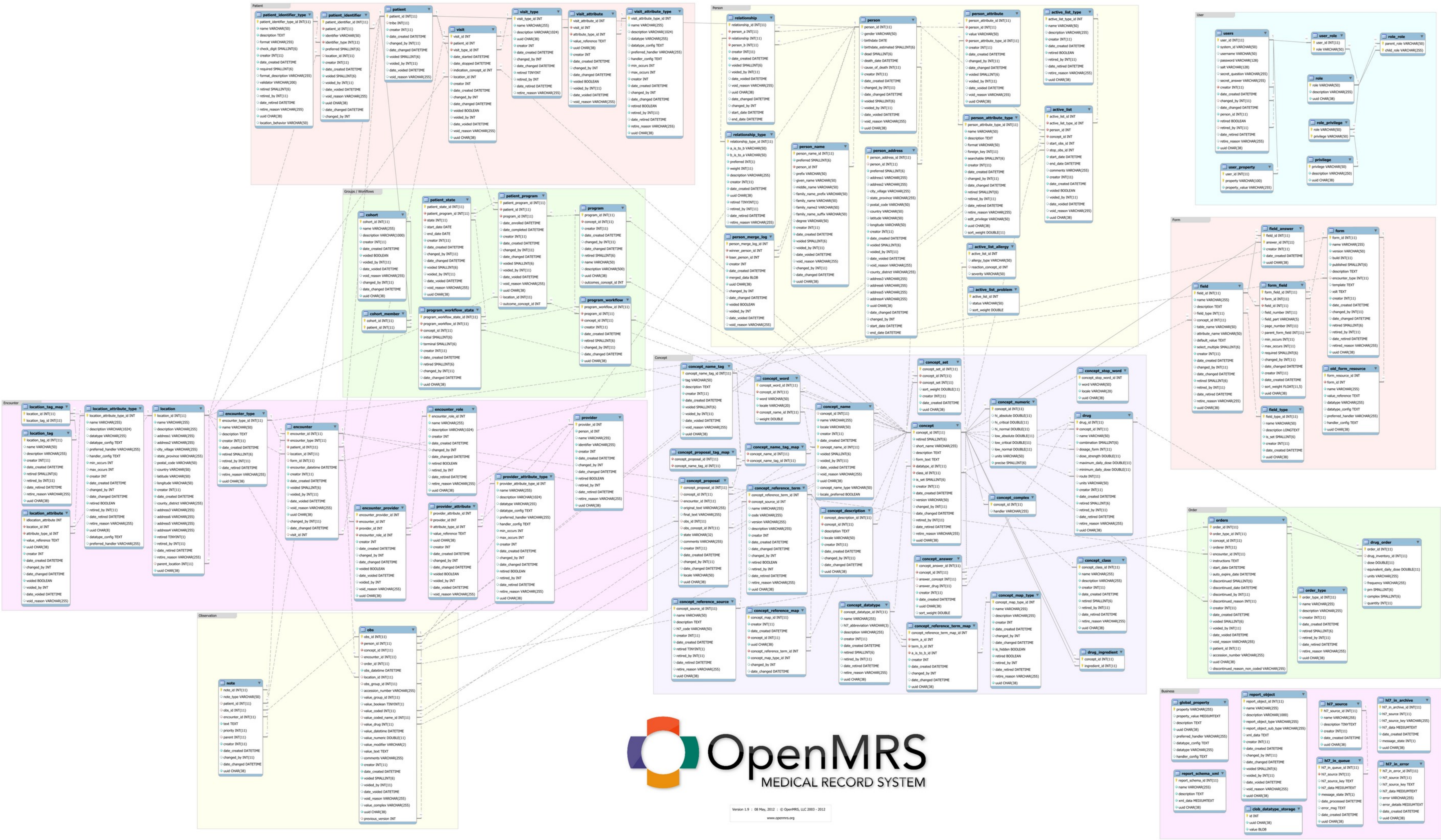


Figure 7.9 Relationships. (a) One-to-one. (b) One-to-many. (c) Many-to-many.

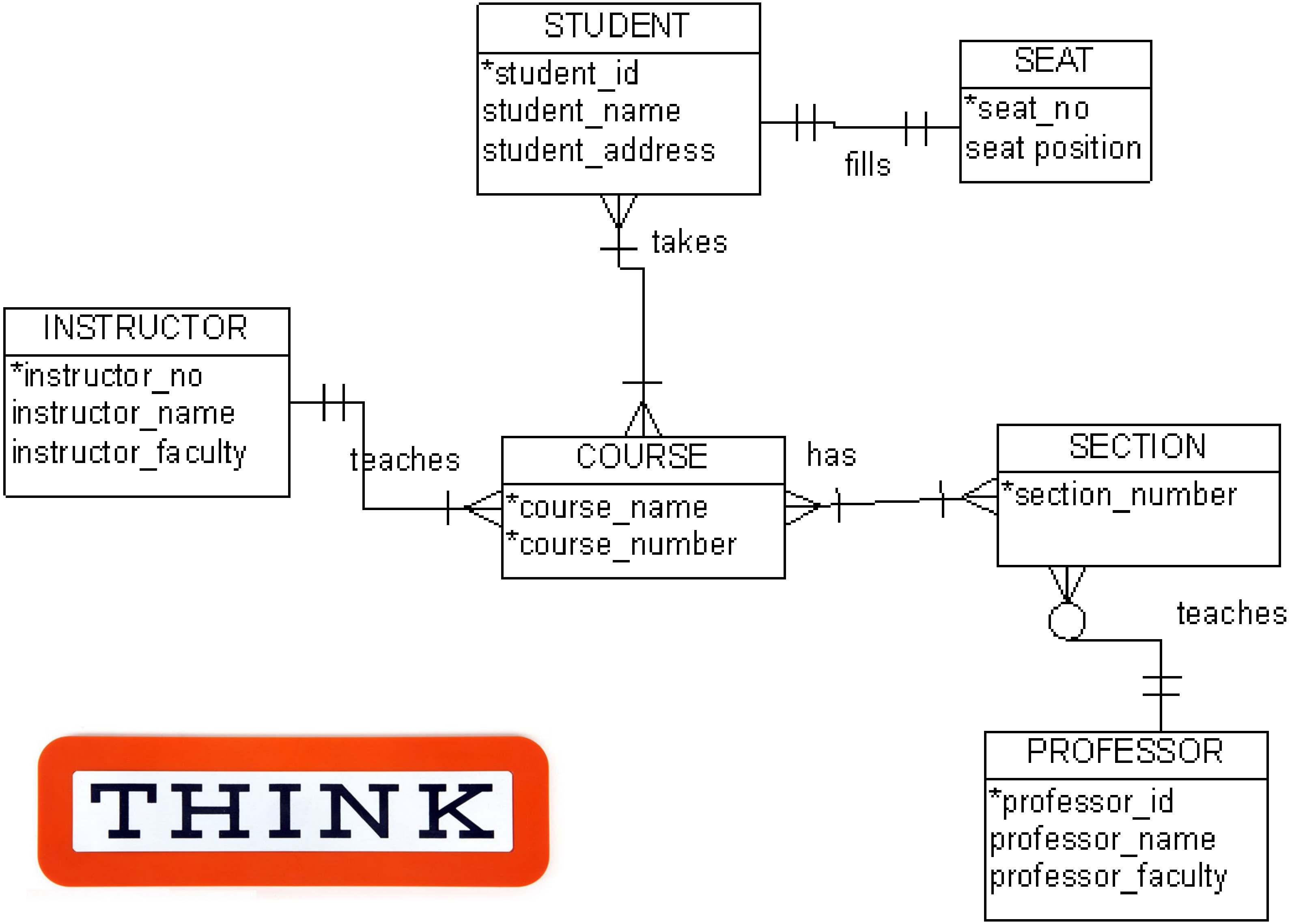
Note: The book uses arrows to denote these relationships

Schema: A Medical Database

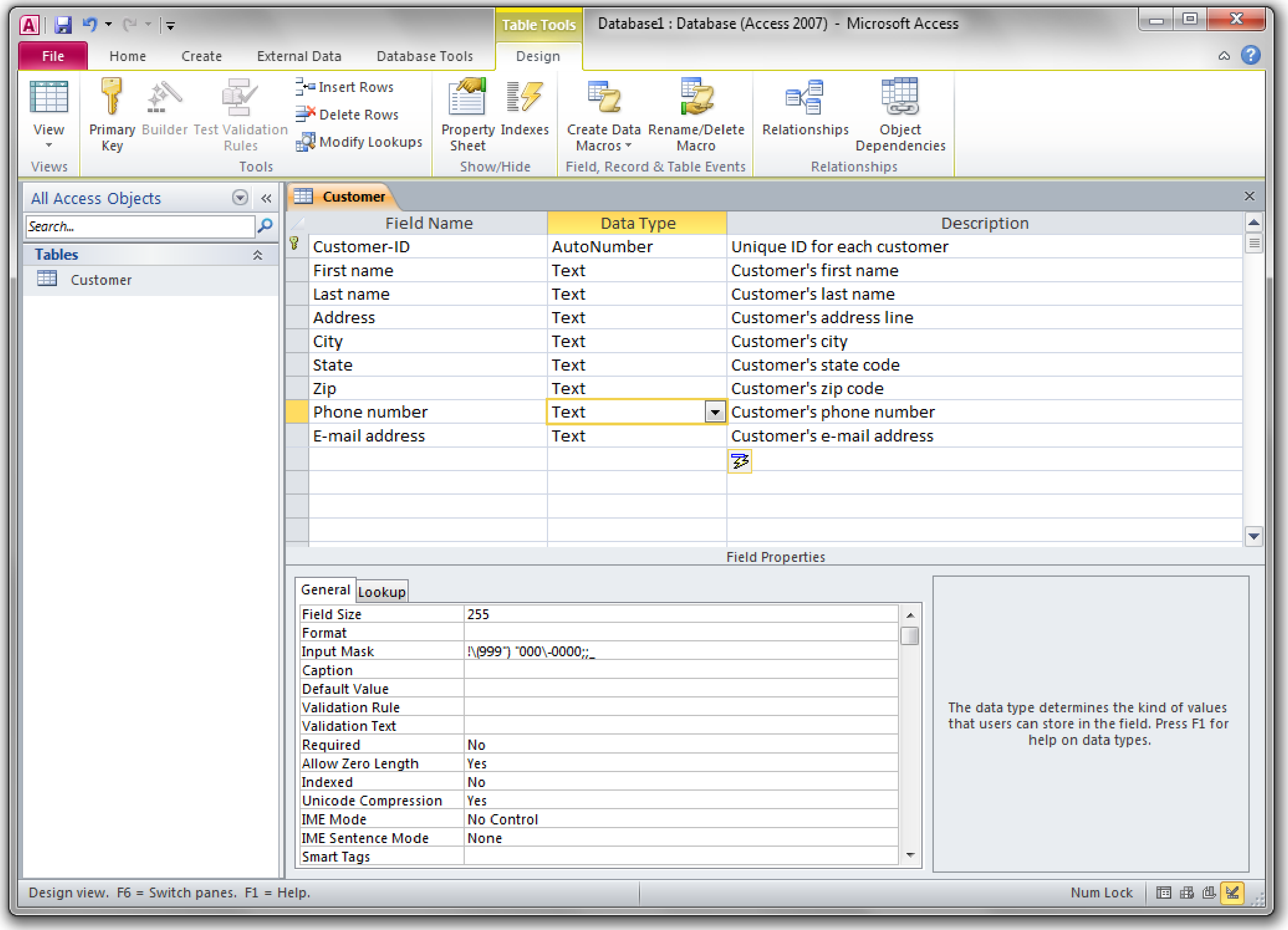


Consider This Schema!

Can you explain the *one-to-many* relationships?



Software to Help Create Tables



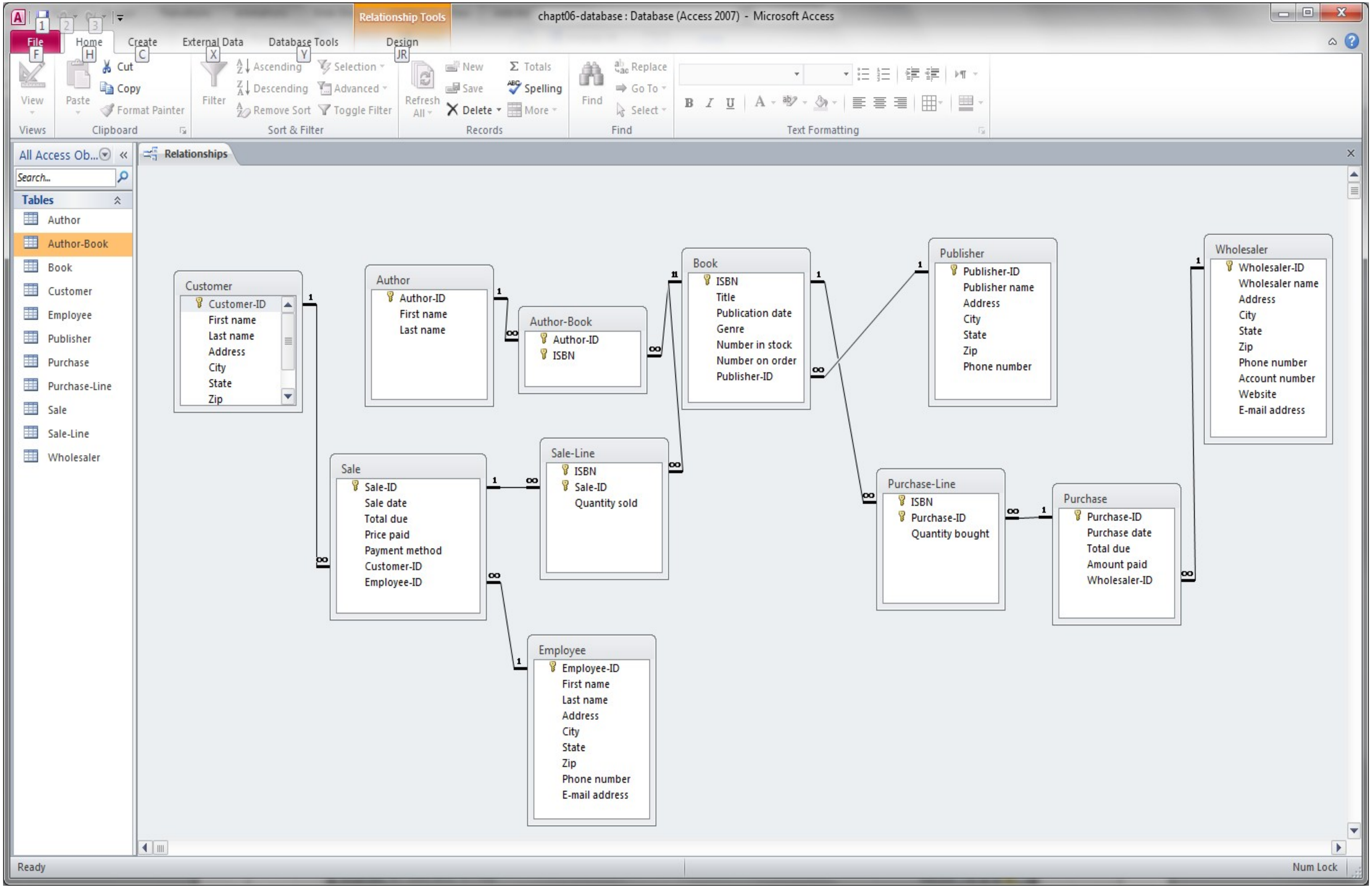
Field Name	Data Type	Description
Customer-ID	AutoNumber	Unique ID for each customer
First name	Text	Customer's first name
Last name	Text	Customer's last name
Address	Text	Customer's address line
City	Text	Customer's city
State	Text	Customer's state code
Zip	Text	Customer's zip code
Phone number	Text	Customer's phone number
E-mail address	Text	Customer's e-mail address

Field Size	255
Format	
Input Mask	!\(999") "000\-0000;;_
Caption	
Default Value	
Validation Rule	
Validation Text	
Required	No
Allow Zero Length	Yes
Indexed	No
Unicode Compression	Yes
IME Mode	No Control
IME Sentence Mode	None
Smart Tags	

The data type determines the kind of values that users can store in the field. Press F1 for help on data types.

Design view. F6 = Switch panes. F1 = Help.

Creating all Tables and Relationships



chapt06-database : Database (Access 2007) - Microsoft Access

Tables
- Author
- Author-Book
- Book
- Customer
- Employee
- Publisher
- Purchase
- Purchase-Line
- Sale
- Sale-Line
- Wholesaler

Customer: Customer-ID, First name, Last name, Address, City, State, Zip

Author: Author-ID, First name, Last name

Author-Book: Author-ID, ISBN

Book: ISBN, Title, Publication date, Genre, Number in stock, Number on order, Publisher-ID

Publisher: Publisher-ID, Publisher name, Address, City, State, Zip, Phone number

Wholesaler: Wholesaler-ID, Wholesaler name, Address, City, State, Zip, Phone number, Account number, Website, E-mail address

Sale: Sale-ID, Sale date, Total due, Price paid, Payment method, Customer-ID, Employee-ID

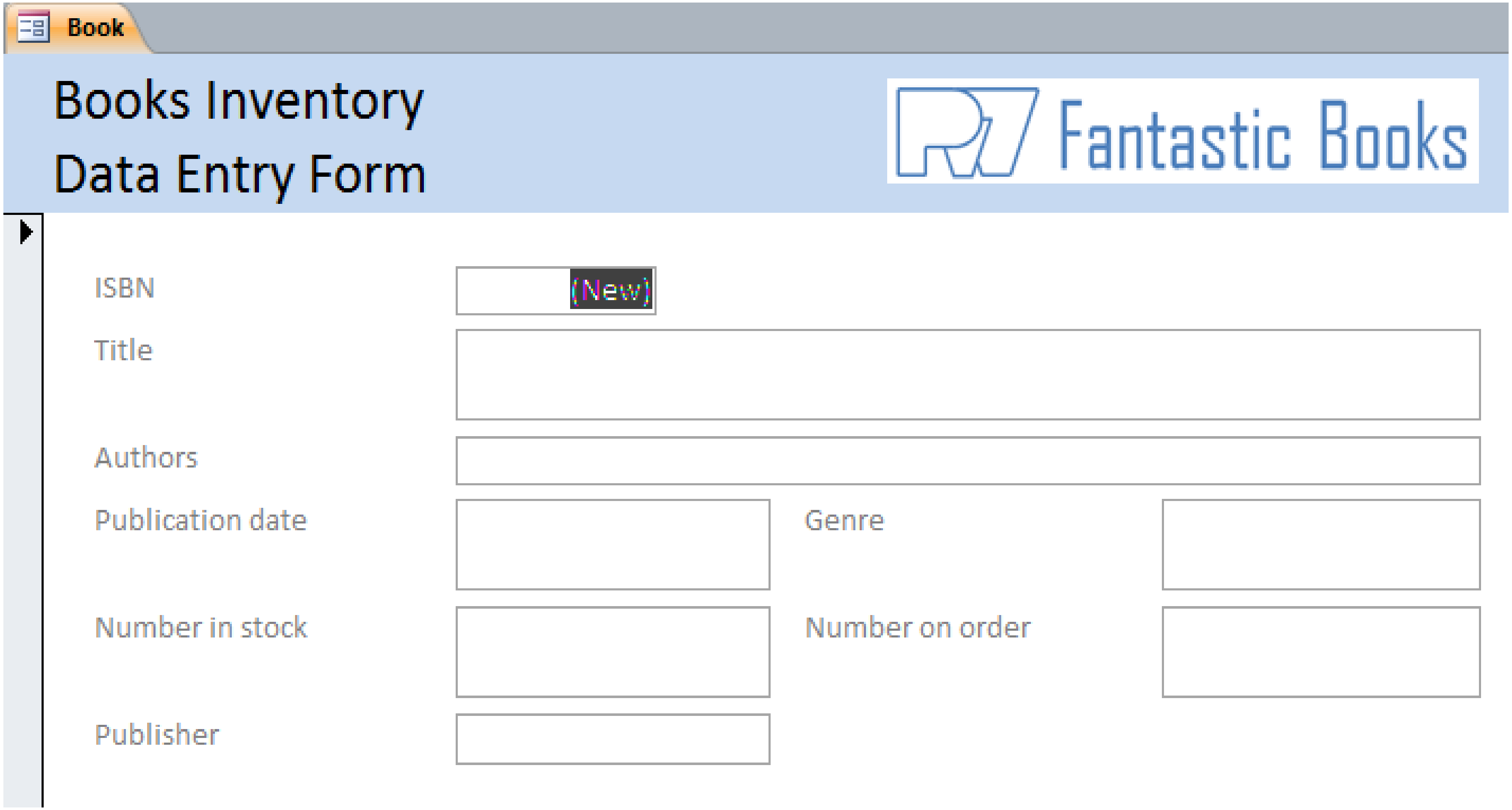
Sale-Line: ISBN, Sale-ID, Quantity sold

Purchase-Line: ISBN, Purchase-ID, Quantity bought

Purchase: Purchase-ID, Purchase date, Total due, Amount paid, Wholesaler-ID

Employee: Employee-ID, First name, Last name, Address, City, State, Zip, Phone number, E-mail address

Forms Capture Data



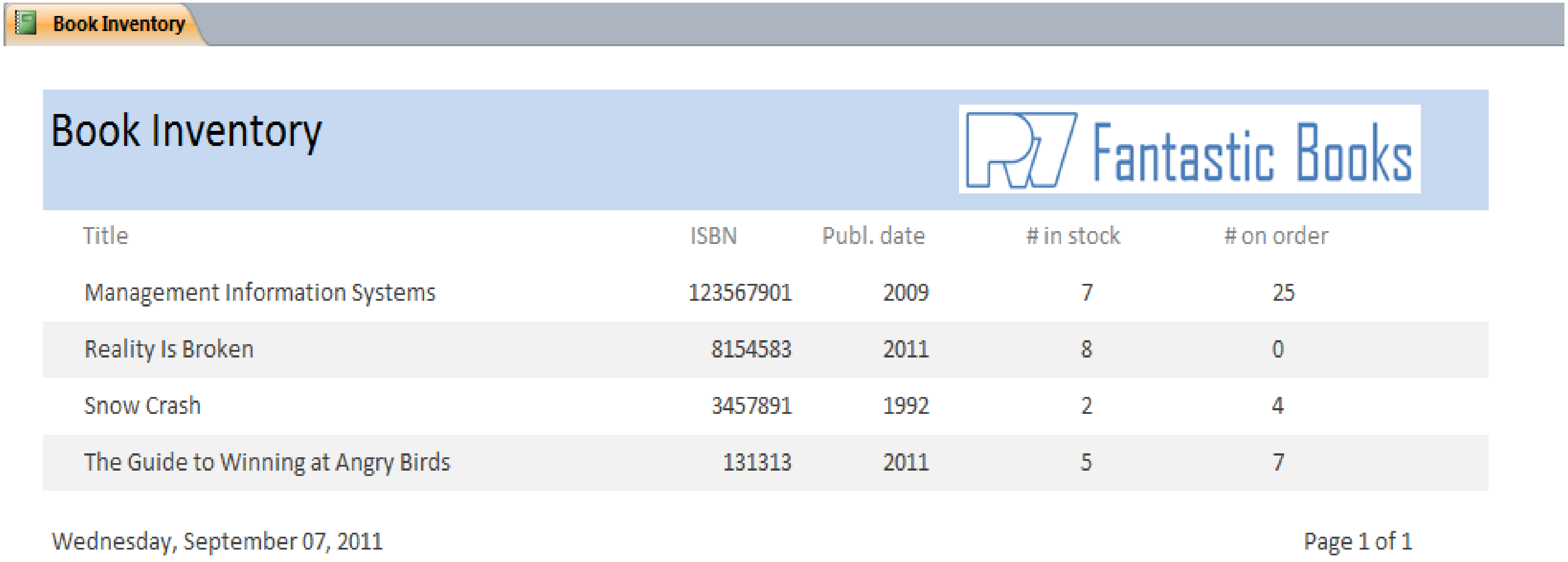
Book

Books Inventory
Data Entry Form

ISBN	(New)		
Title			
Authors			
Publication date		Genre	
Number in stock		Number on order	
Publisher			

Databases are often found behind forms to input and output information

Reports Provide Output



Book Inventory

Fantastic Books

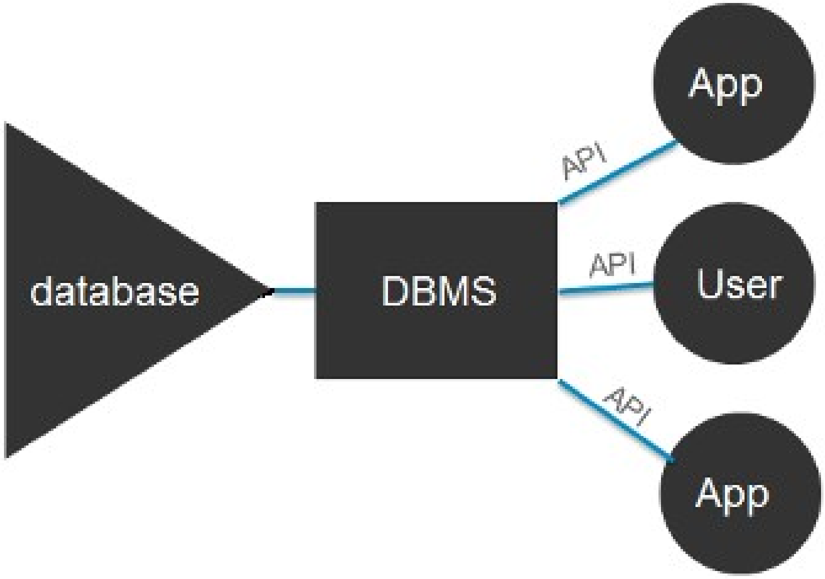
Title	ISBN	Publ. date	# in stock	# on order
Management Information Systems	123567901	2009	7	25
Reality Is Broken	8154583	2011	8	0
Snow Crash	3457891	1992	2	4
The Guide to Winning at Angry Birds	131313	2011	5	7

Wednesday, September 07, 2011

Page 1 of 1

Database Administration

- Responsibilities
  - Defines & organizes database structure and content
  - Develops security procedures
  - Develops database documentation
  - Maintains database management system (DBMS)
    - software that handles the storage, retrieval, and updating of data in a computer system.



So, Why a Database, Again?

To keep track of the warehouse supplies (Amazon, for example...)

Data Warehouses vs Databases

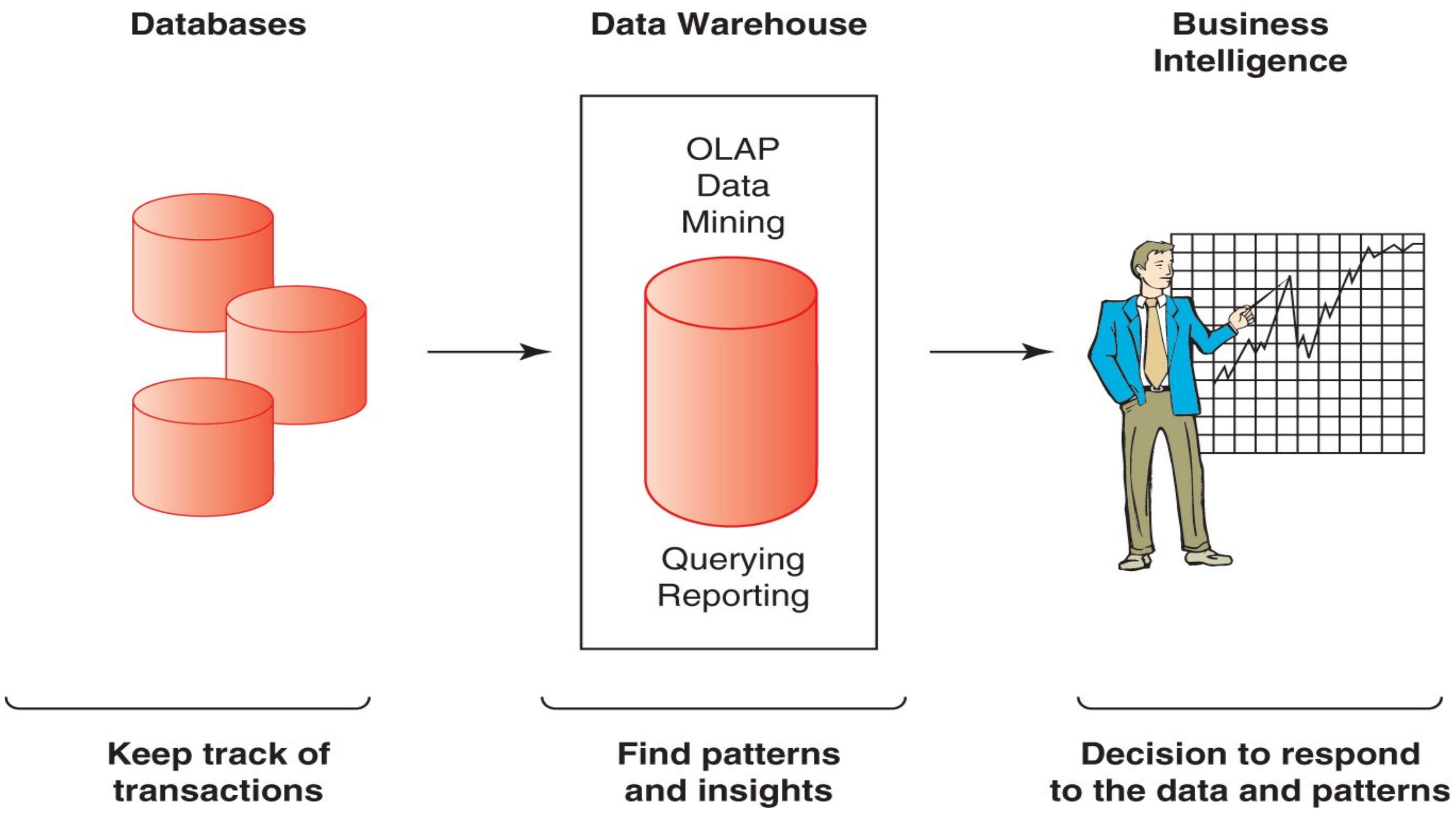
- **Data Warehouses**
  - Non-volatile
    - Non-editable: no changes to data
  - Archived data from fiscal periods,
  - Historical records

Storage of Data

- **Databases**
  - Dynamic
  - Able to change data
  - Able to be updated

Access and
Ability to
Update the Data

Database Trends –
Data Warehouse & Data Mining



Databases

Data Warehouse

Business Intelligence

OLAP
Data
Mining

Querying
Reporting

Keep track of transactions

Find patterns and insights

Decision to respond to the data and patterns

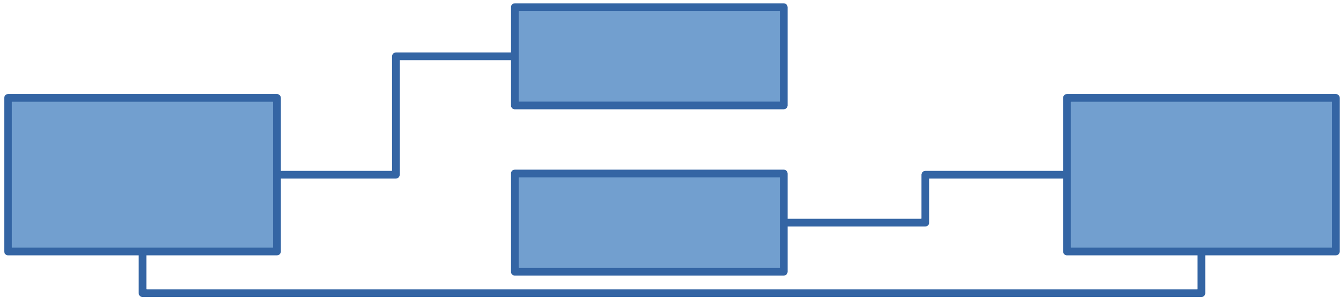
OLAP: Online Analytical Processing

Consider this...

1) Can you organize these terms into tables?

student_id
course_name
seat_number
professor_id
student_name
course_number
professor_name
section_number
seat_position
student_address
instructor_name
instructor_number
professor_department
instructor_department

2) Connect these tables based on some logic.



THINK