

COPENHAGEN BUSINESS ACADEMY



Database

Marjahan Begum

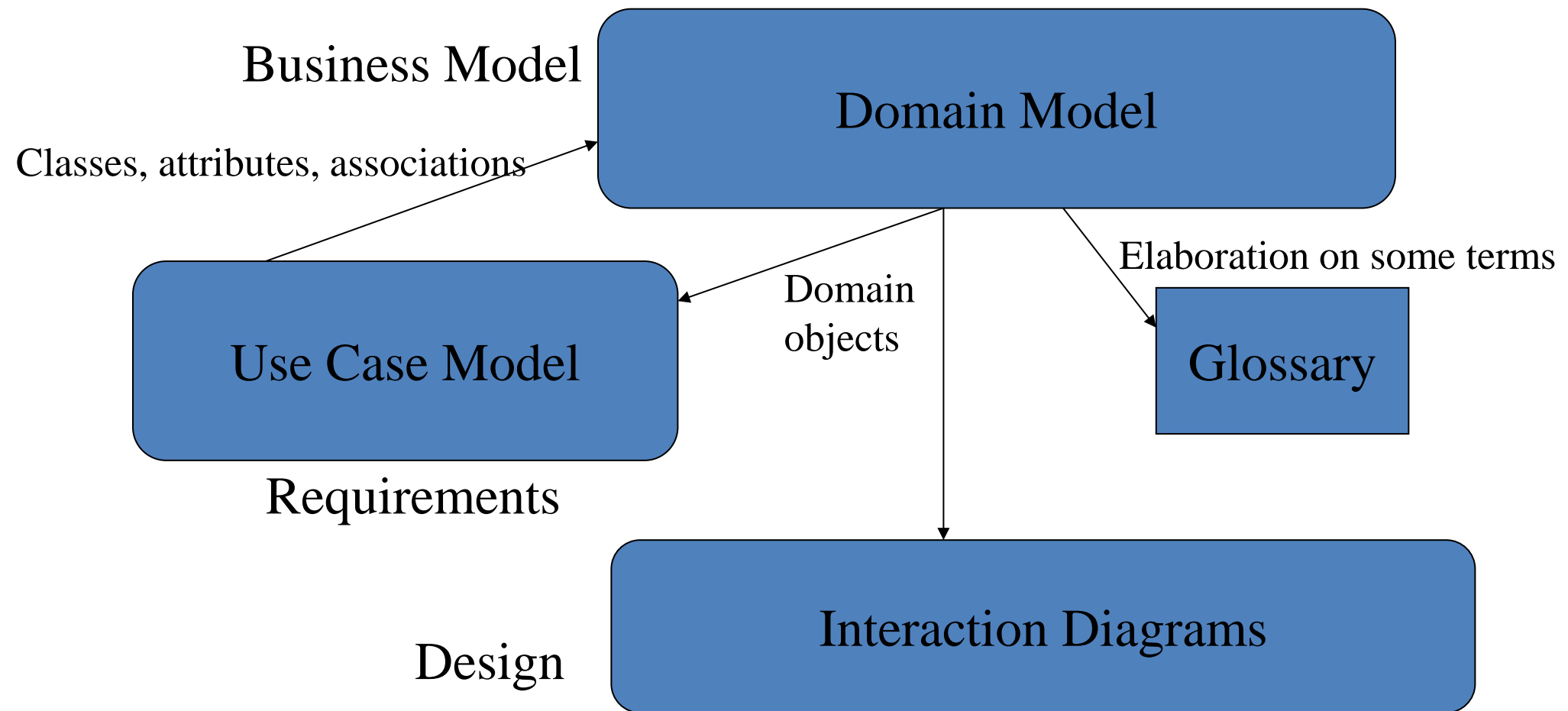
Week plan

- Monday
 - Requirements Specification → Domain model → E-R Model
 - Model driven database design
 - Reverse Engineering
- Tuesday
 - SQL Queries
- Wednesday
 - Data Definition Language and Data Manipulation Language
- Thursday
 - Further database design and Normalisation
- Friday
 - Study point assignment

Monday

- Recap of domain model
- How do you translate business domain model to database design

Domain Model Relationships

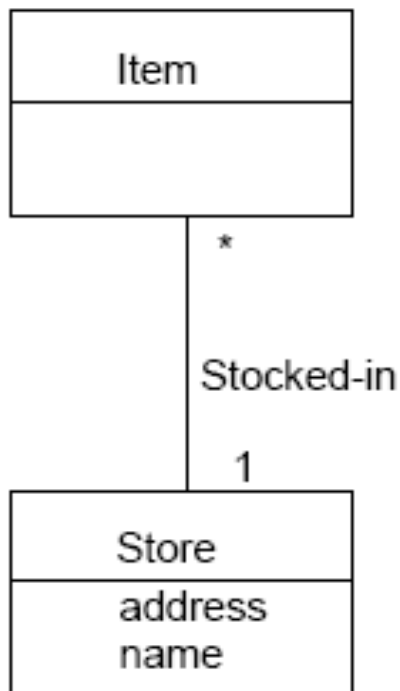


A Domain model

- illustrates meaningful conceptual classes in a problem domain.
- is a representation of real-world concepts, not software components.
- is NOT a set of diagrams describing software classes, or software objects and their responsibilities.
- It may show:
 - concepts
 - associations between concepts
 - attributes of concepts

Domain Model and UML Notation

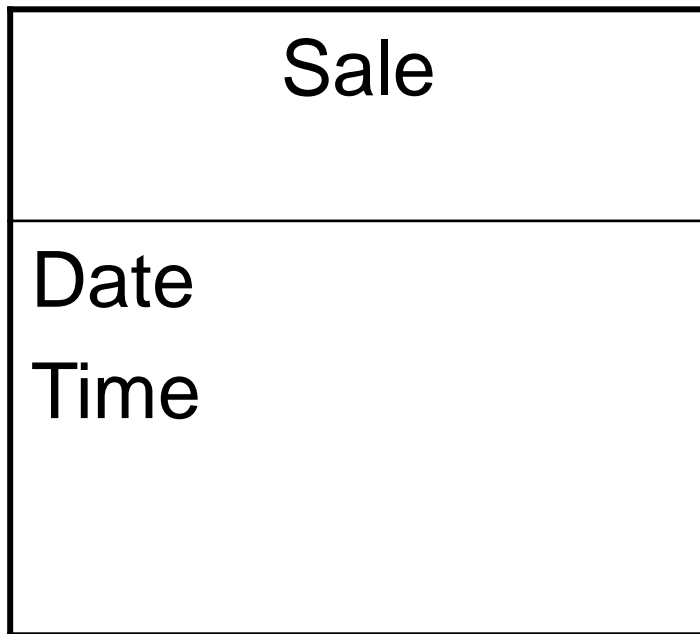
- Illustrated using a set of class diagrams for which no operations are defined.



- A Domain Model is a description of things in the real world.
- A Domain Model is not a description of the software design.
- A concept is an idea, thing, or object.

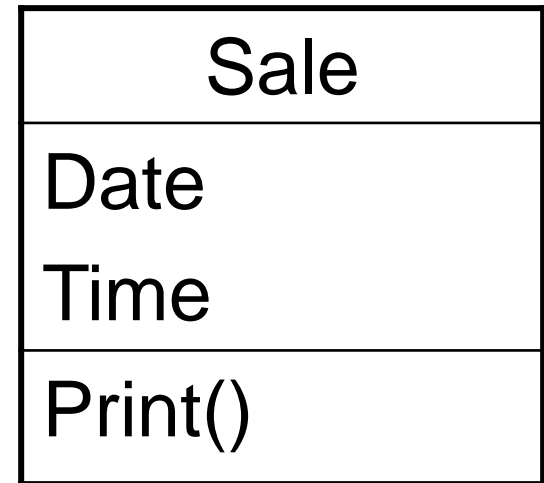
A Domain Model is not a Software Artifact

A Conceptual class:



vs.

Software Artifacts:



Steps to create a Domain Model

1. Identify Candidate Conceptual classes
2. Draw them in a Domain Model
3. Add associations
 - relationships that must be retained
 - Roles, describing the multiplicity and participation of a class in a relationship
4. Add attributes
 - necessary for information to be preserved
 - each instance of the class separately holds a value
5. Additional rules
 - Add to notes

Description of a Service Example (Flight)

Flight
Date
Time
Number

Flies-to

vs.

Airport
Name

Flight
Date
Time

Described
-by

Flight Desc
Date
Time

Describes
-flights-to

Airport
Name

1/2 Conceptual Class Category

<ul style="list-style-type: none">Physical or tangible objects<ul style="list-style-type: none">Register, Airplane	<ul style="list-style-type: none">Containers of other things<ul style="list-style-type: none">Store, Hangar, Airplane
<ul style="list-style-type: none">Specifications, or descriptions of things<ul style="list-style-type: none">ProductSpecification, FlightDescription	<ul style="list-style-type: none">Things in a container<ul style="list-style-type: none">Item, Passenger
<ul style="list-style-type: none">Places<ul style="list-style-type: none">Store, Airport	<ul style="list-style-type: none">Computer or electro mechanical systems<ul style="list-style-type: none">CreditPaymentAuthorizationSystem , AirTrafficControl
<ul style="list-style-type: none">Transactions<ul style="list-style-type: none">Sale, Payment, Reservation	<ul style="list-style-type: none">Catalogs<ul style="list-style-type: none">ProductCatalog, PartsCatalog
<ul style="list-style-type: none">Transaction items<ul style="list-style-type: none">SalesLineItem	<ul style="list-style-type: none">Organisations<ul style="list-style-type: none">SalesDepartment, Airline
Roles of people Cashier, Pilot	

Monopoly Concepts (candidates)

Monopoly Game

Player

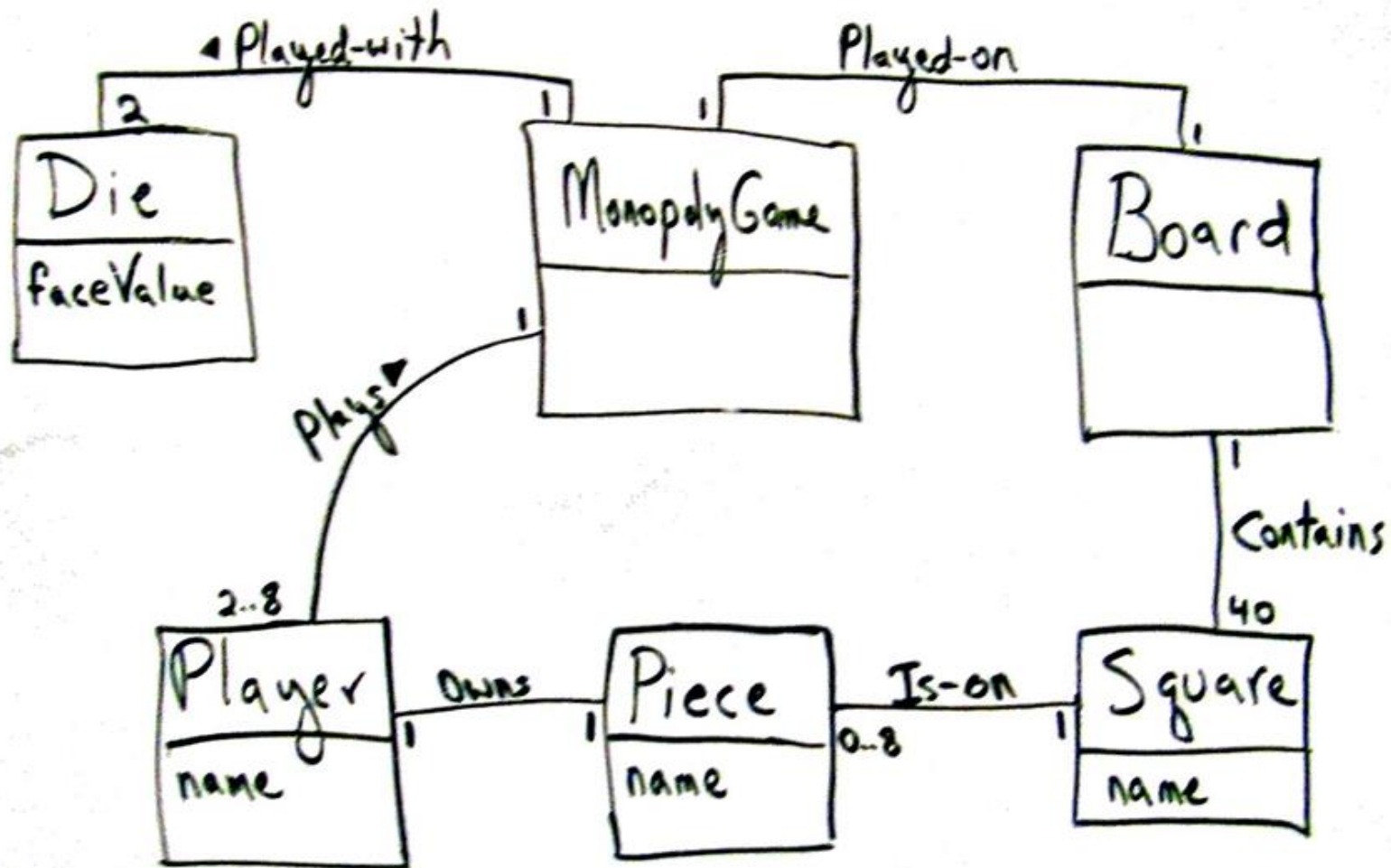
Piece

Die

Board

Square

Monopoly Game Domain Model
Source : Applying UML and Patterns by C. Larman



3/4 Associations and Attributes

- A link between two classes (“has a”)
 - Typically modeled as a member reference
- A Person works for a Company
- Roles names and multiplicity at association ends
- Direction arrow to aid association name
- Logical data value of the object

Let's try!

- This school has many students, distributed into classes, each student goes in one class. Each class has many hours in various subjects, and these subjects are held in many different rooms. This school has several teachers who teach more classes

RED - possible domain classes

GREEN – associations between classes

PURPLE – could tell about multiplicity

Exercise

Think of your local Pizza shop.

They offer a lot of products, chosen from a menu.

Typically, you can then select additional accessories, eg. extra cheese, extra ham, pineapples or other.

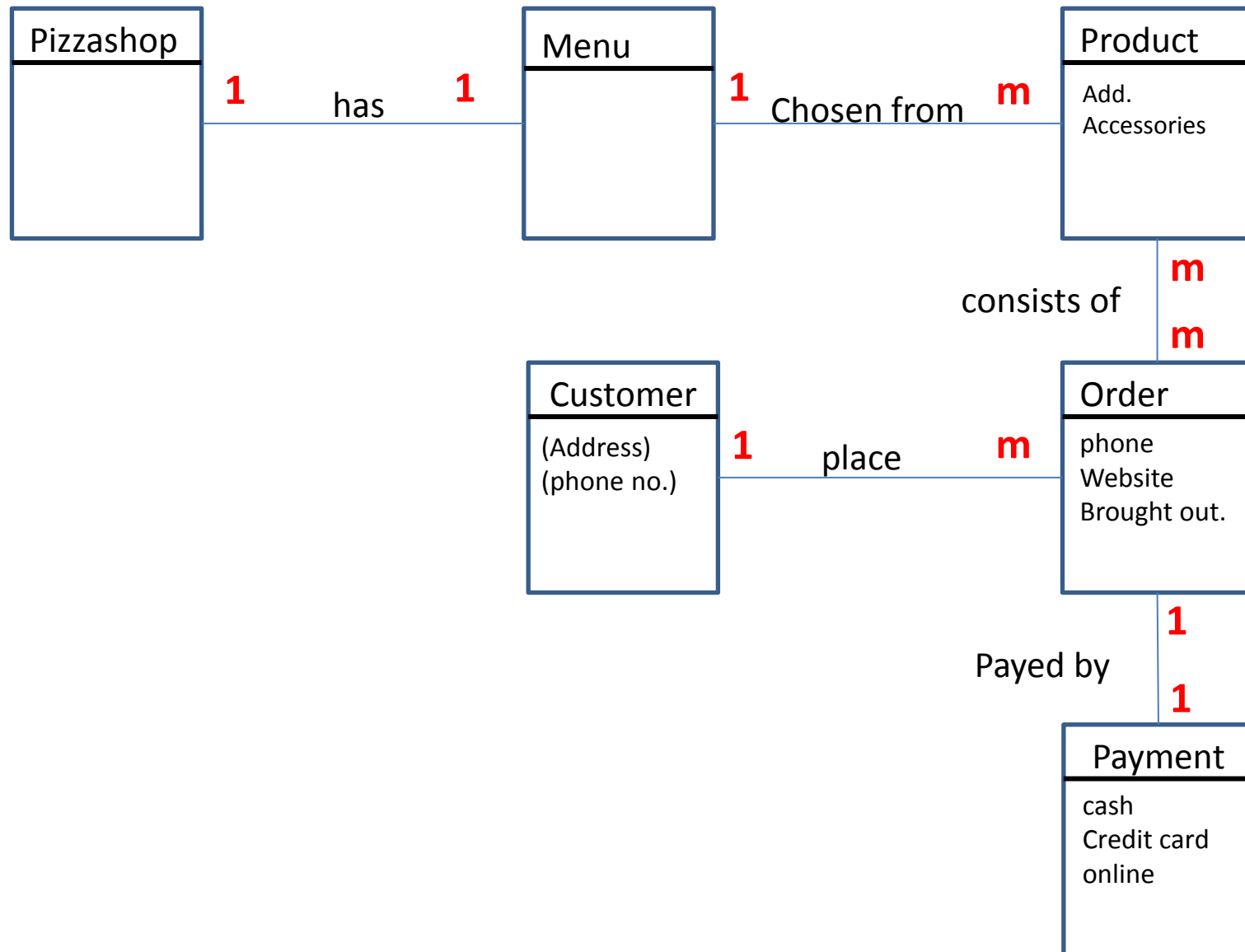
Ordering is done either by phone or on a website.

The customer can select the food picked up or to be brought out. Payment is made either online when ordering or with Credit Card or cash on pickup.

Create a domain model

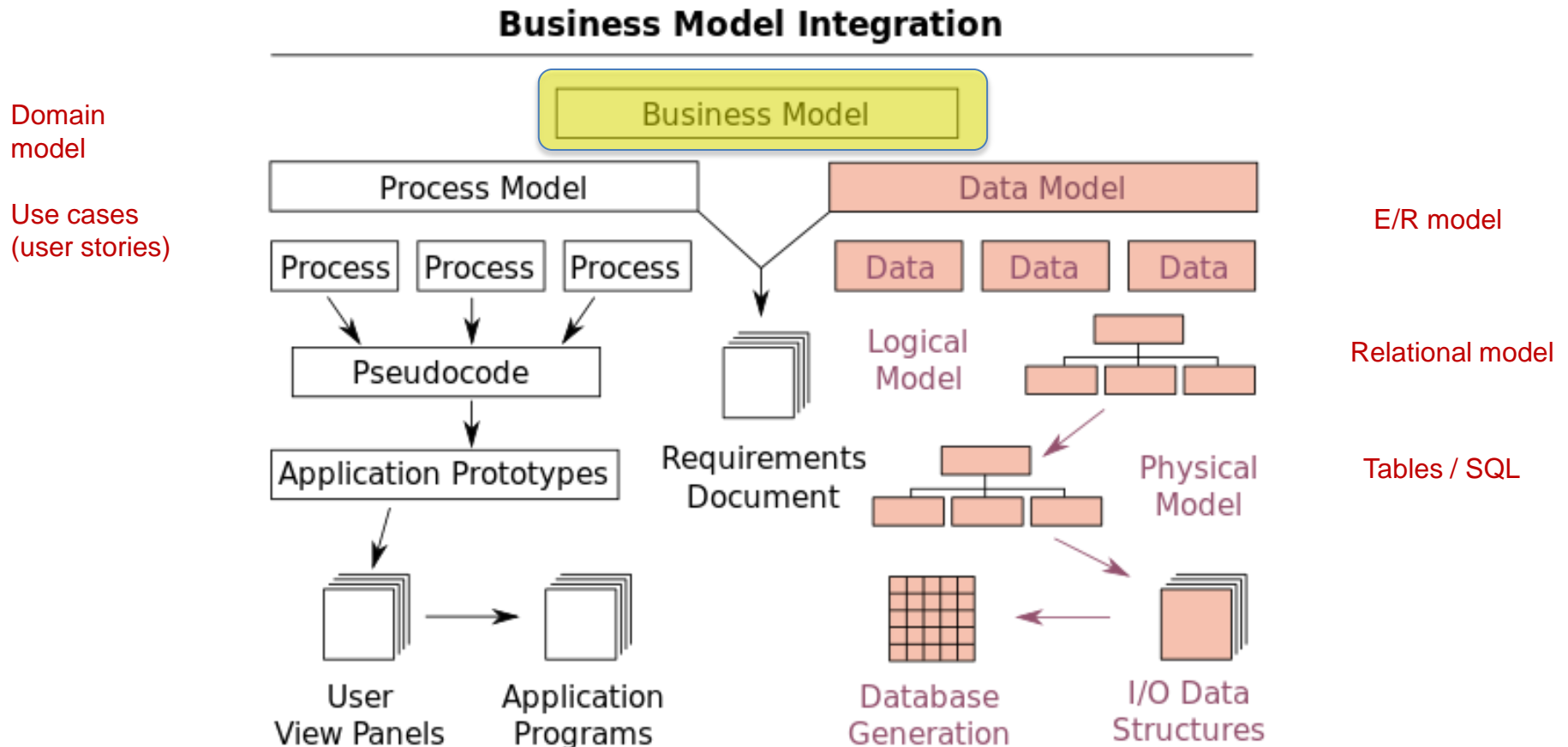
From Palle's slides	
Pizza-shop.	C
Products	C
menu	C
additional accessories	A
extra cheese	A
extra ham	A
pineapples	A
Ordering	C
phone	A
website	A
Customer	C
food	Synonym
Payment	C
Credit Card	A
Cash	A
“brought out”	A
”pick up”	A

Domain Model Pizzashop



Database design – it starts with business modeling

Let's go back to your January activity with Palle for a moment ...



Data models at several levels

Data models transform from logical into physical form:

- **Conceptual model**

- Presentation of data in end user terminology and context
- Technology free
- Domain model (business focus), **E/R model** (persistence focus)

- **Logical model**

- Can partly be understood by end-users
- Relies on technology, but doesn't show it directly
- **Relational model**, network, hierarchical, document, graph, OO models

- **Physical model**

- Close to the physical representation of the database
- Technology specific, e.g. **MySQL (SQL scripts)**
- Disk usage, security, etc.

Database design

- Later this semester, you will work in project groups of 4-5 students.
- Make a suggestion of
 - which tables you will need to represent the project groups
 - Which columns these tables must contain

From business model to database modeling

- Entity – Relationship modelling (ERD) – is graphical representation of data requirement for database for a from a specific domain
- ERD – Entity, Attributes, Primary key, Relationship and Cardinality
- ERD
 - is same as table/database table
 - Entity properties/**attributes** – are set of columns in each table
 - Every entity must have at least one attribute that uniquely identify the entity/**primary key**

From business model to database modeling

- Entity
 - A person, place or a thing that you want to keep track e.g Product, Book, Student
 - Becomes a table in the database
 - Each occurrence is an instance and in physical terms becomes rows the table (individual student, specific book etc..)
- Attributes
 - Characteristics of an individual identity
 - More about the entities e.g customer name, phone number, address
 - Columns in table
- Primary key
 - An attribute or group of attributes that uniquely identifies an instance/row

Naming Principles

Table names are like class names:

- Upper case first letter
- **Singular**
- no_ - use LargeandSmallLetters

Column names are:

- Lower case first letter
- **Singular**
- no_ - use smallAndLargeLetters

Simple Data Types

Numeric data types :

- INTEGER, **INT**, SMALLINT, TINYINT, MEDIUMINT, BIGINT
- FLOAT, **DOUBLE**

Number in parenthesis “INT(11)” indicates display size when printed in Workbench

Text comes in two fundamental types:

- CHAR and VARCHAR. **Use VARCHAR!**

Number in parenthesis “VARCHAR(20)” indicates maximum length– if nothing is specified it can be up till 65535 characters.

- VARCHAR uses storage according to concrete content
- CHAR uses storage as indicated in parenthesis and can be up till 255 characters

Date Types

The **DATE** type is used for values with a date part, but no time part. MySQL retrieves and displays DATE values in 'YYYY-MM-DD' format. The supported range is '1000-01-01' to '9999-12-31'.

The **DATETIME** type is used for values that contain both date and time parts. MySQL retrieves and displays DATETIME values in 'YYYY-MM-DD HH:MM:SS' format.

The supported range is '1000-01-01 00:00:00' to '9999-12-31 23:59:59'.

The **TIMESTAMP** data type is used for values that contain both date and time parts.

TIMESTAMP has a range of '1970-01-01 00:00:01' UTC to '2038-01-19 03:14:07' UTC

Unique Columns

If you declare a column unique, there cannot be two rows with the same value

- Emails and phone numbers are often unique

Notice— even though a column is declared as unique there can be null values in several rows

Not Null & Default Value

A column can be declared to never to contain any null values in any rows

You can specify a default value for a column

You can combine not null with a default value

Key

A key is a unique column that cannot be null

The database creates an **index** for keys which make it much faster to retrieve a row based on its key.

Example:

```
select *  
from student  
where email= "cph-rg54@cphbusiness.dk"
```

Sometimes a key is composite

Example:

```
CREATE TABLE orderdetails (  
  orderNumber int(11) NOT NULL,  
  productCode varchar(15) NOT NULL,  
  quantityOrdered int(11) NOT NULL,  
  priceEach decimal(10,2) NOT NULL,  
  orderLineNumber smallint(6) NOT NULL,  
  PRIMARY KEY (orderNumber,productCode)  
)
```

Primary Key

A primary key is a key chosen for identification of rows in a table

You should not change a primary key over time.

This means that phone numbers and email addresses are poor primary keys

Foreign Key

A foreign key is a column that has a value which exists as primary keys in a (other) table

Example: The **office** table in the classicmodels database has primary key “officeCode”. In the table **employees**, there is a column which represents the office that the employee belongs to. This column is a foreign key.

Notice: Since more employees can belong to the same office, the foreign key is not unique in the **employees** table. Also, freelancers do not belong to an office and therefore the foreign key might also be null.

From business model to database modeling

- Relationships and Cardinality
 - Nature of interactions between the entities
 - Draw a line
 - Count of instances that are needed between entity relationships
 - Number of row from one table that can be linked to another table
 - Minimum/Maximum

Relationship Between Tables







Primary key and foreign keys are used to create relationships between tables.

There are different characteristics of relationships

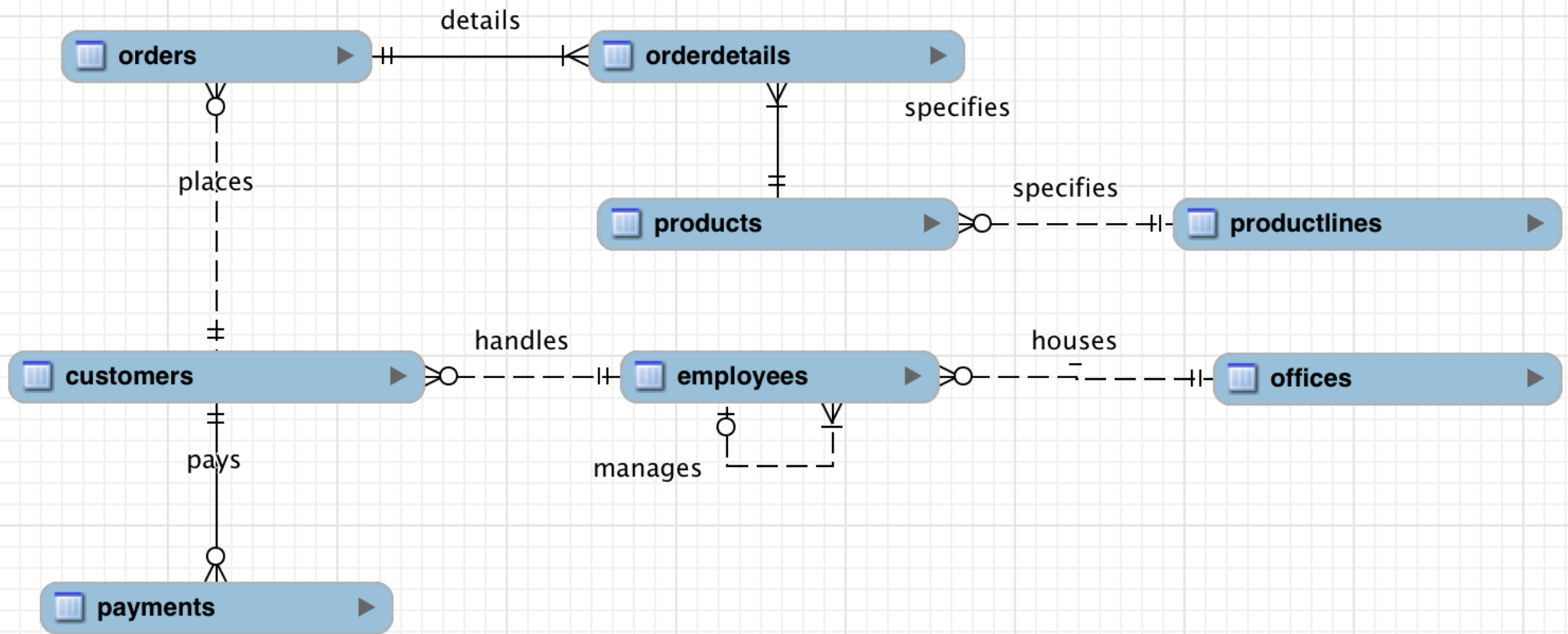
- One to one
- One to many
- Many to many
- Zero or one

E/R Model - crow foot multiplicity notation

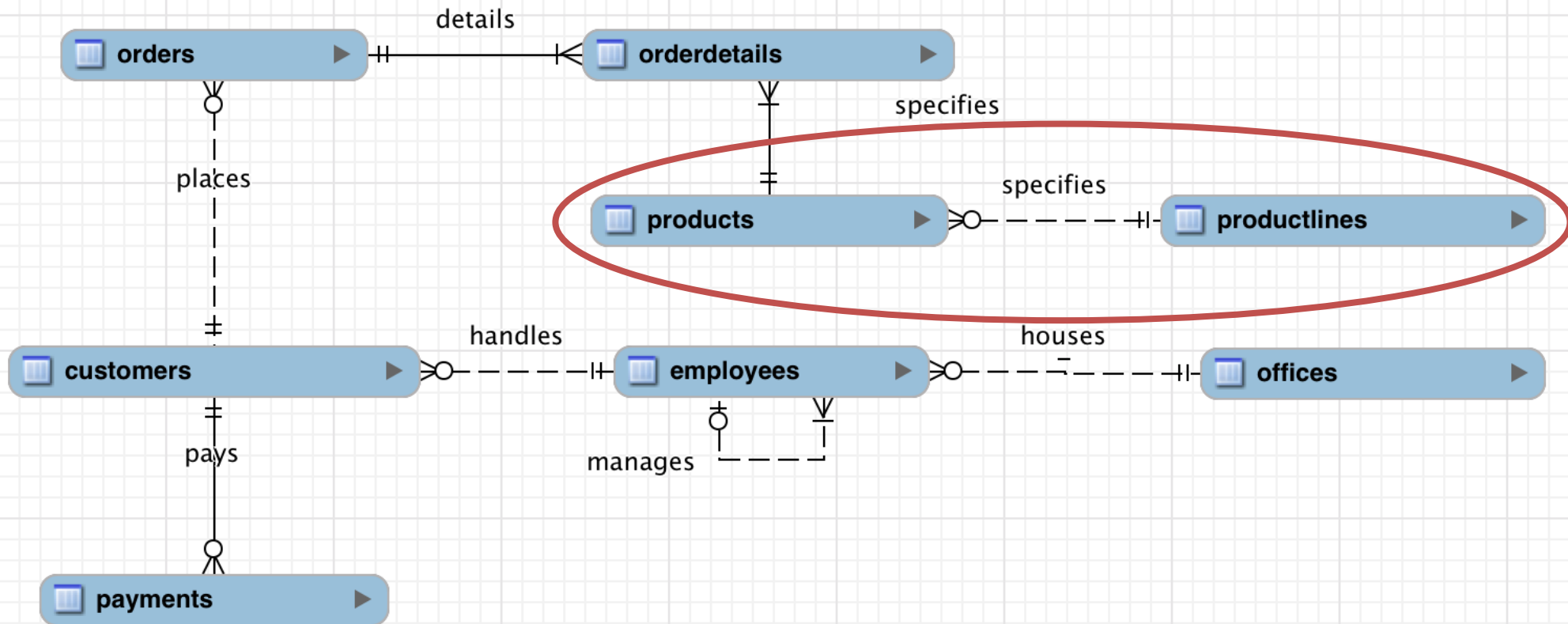
Notation style for multiplicity

	One
	Many
	One (and only one)
	Zero or one
	One or many
	Zero or many

Relationship - 1



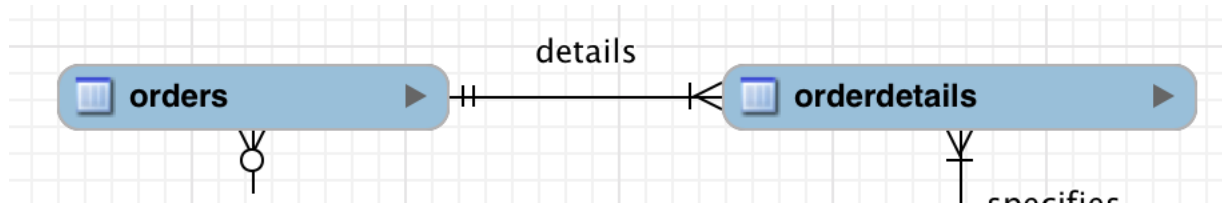
Relationship - 2



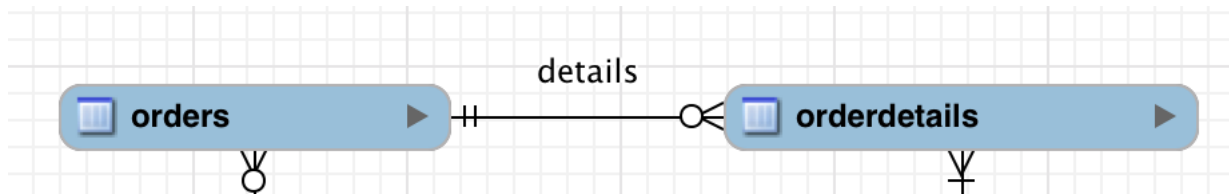
0 M relationship that says that each product specified by the product line
each product line specifies multiple products (or no)

Relationship

Comparing 1-M and 0-M relationship



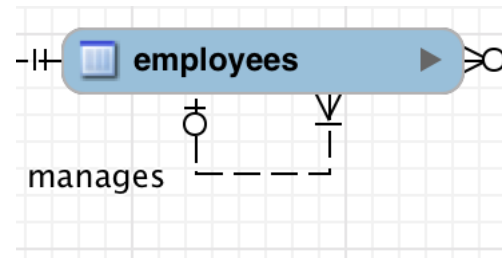
We must create at least one detail when creating an order. Often the order must be created first, and then details afterwards. One should therefore often have this model in place



Rather than leaving it up to the business application to ensure that all orders have at least one detail.

Relationship 4

Look at this :



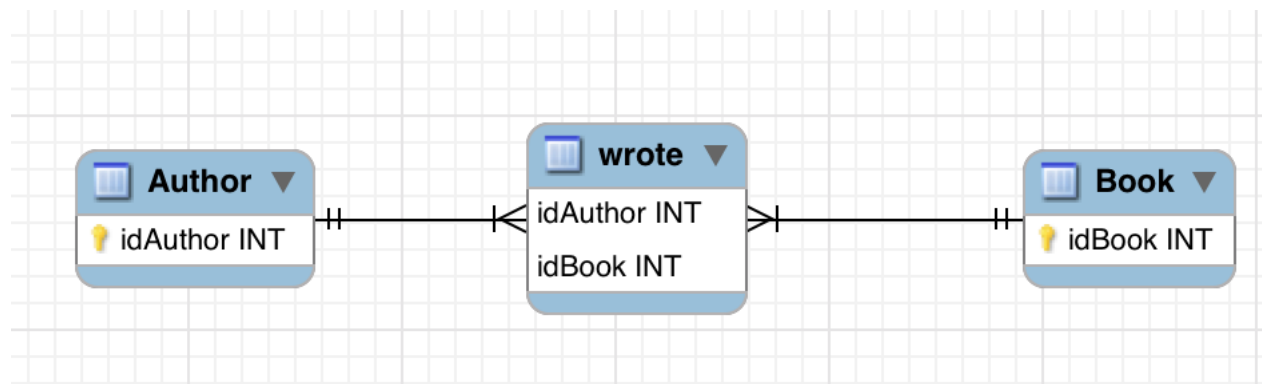
Is this correct modelling?

Many to Many Relationship

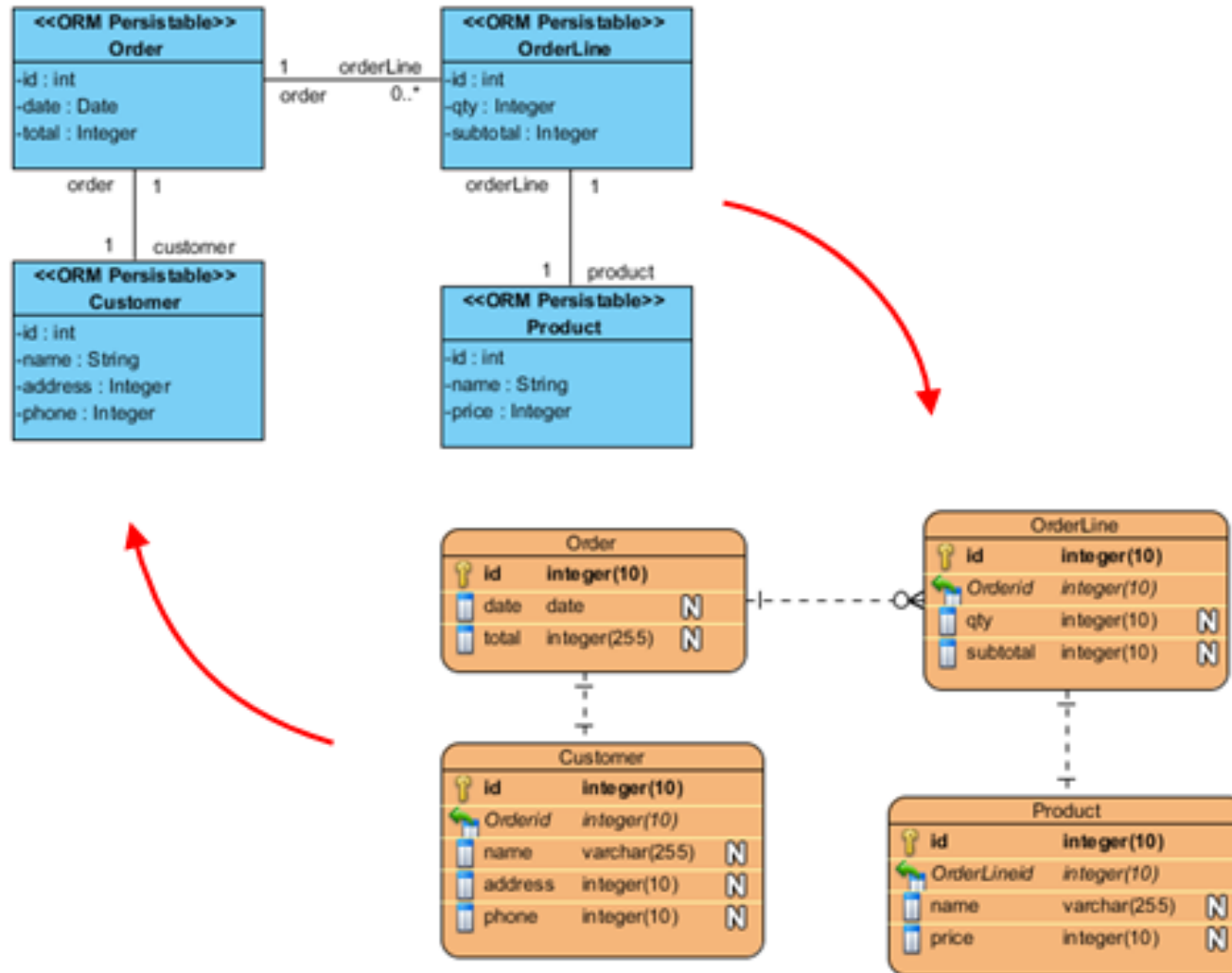
A database has two tables authors and books

An author can write many books and a book can be written by many more authors.

This relationship is many to many. This cannot be done without introducing additional table



Entity Relationship Diagram/Class Diagram



Demo –MySQL Workbench modeling

Forward engineering:

Start with model -> use tool to **generate** database

Instructions at:

<https://dev.mysql.com/doc/workbench/en/wb-getting-started-tutorial-creating-a-model.html>

Database Design / Modeling

- MySQL Workbench provides extensive capabilities for creating and manipulating database models, including these:
 - Create and manipulate a model graphically
 - Reverse engineer a live database to a model
 - Forward engineer a model to a script or live database
 - Create and edit tables and insert data
- <https://dev.mysql.com/doc/workbench/en/wb-data-modeling.html>
- The [MySQL Workbench](#) offers creating, editing and exporting EER Models. Exporting to PNG and PDF allows easy sharing for presentations.

Database Design Exercise (work in pairs)

Make a model of the library as specified by the file "library-system.pdf"

The model must be drawn in MySQL workbench. and hereafter created as a database.

Insert a borrower and a few materials into the database. Let the borrower borrow one material and let him/her make two reservations.

Make a SQL statement that can show a borrower's name and the material title of all his/hers reservations

Extra: Reverse Engineering Exercise

Diagrams can be made based on an existing database.

Useful when:

- You don't have the model already
- You have changed the database, but not the model

Be careful with 0/1 – M relationships – it is not certain that it is done properly when you reverse engineer a database into a model

Exercise

- Make a reverse engineering of classicmodels database
 - See: <https://dev.mysql.com/doc/workbench/en/wb-reverse-engineer-live.html>
- Adjust the diagram so it makes the most sense (= because a useful artifact)

