Creating and Displaying Many Products

Info 5001 – Application Modeling and Design

Dr. Kal Bugrara

The problem

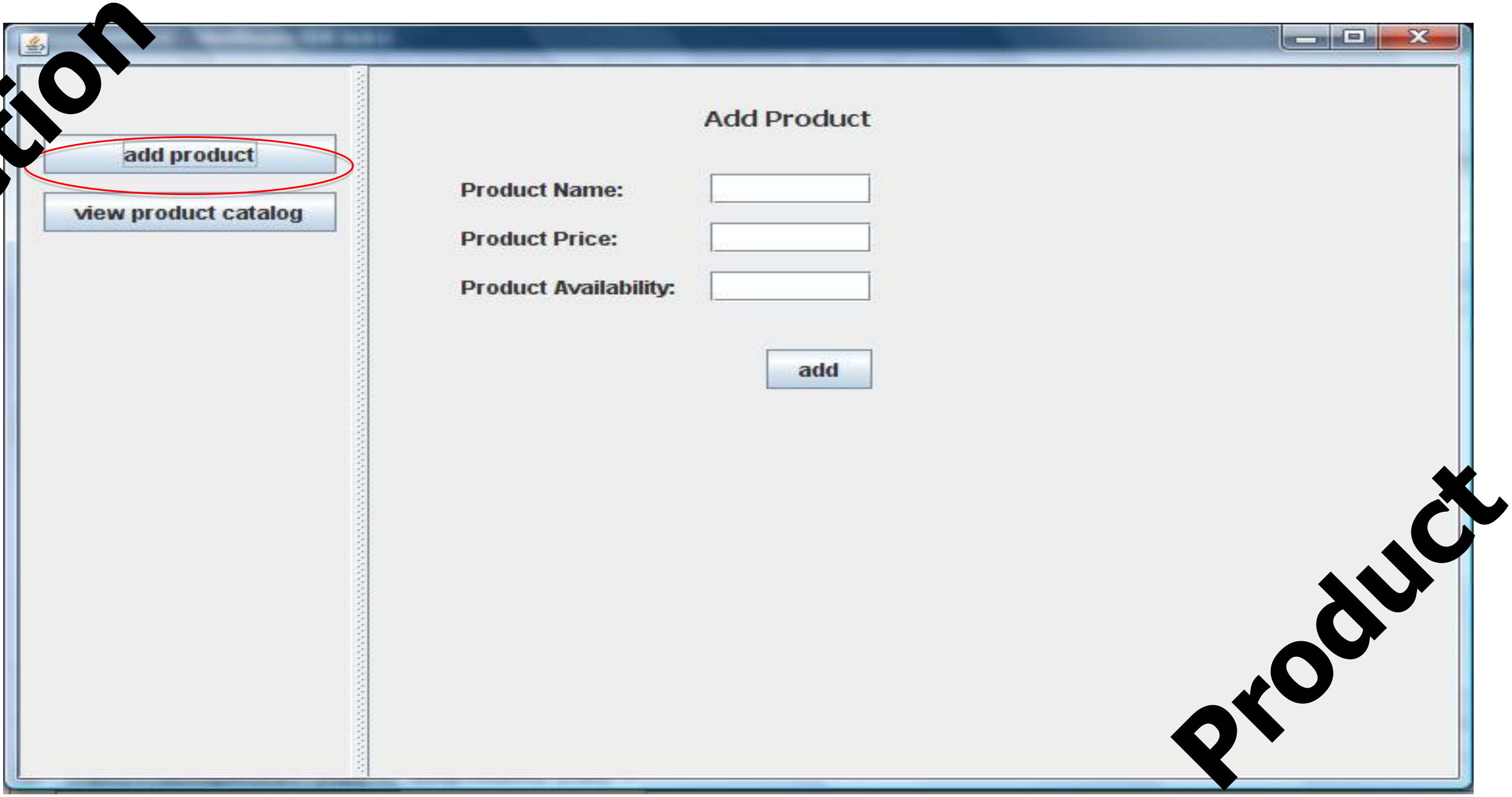
- Businesses sell products to customers.
- They must organize their products so customers know about them in terms of features, availability, and price.
- Sometimes, new products get added, discontinued, updated, etc.
- Business need software to help them managing their products (product catalog).

Objective

- Design Principles and their applications
  - Hierarchy –break components down to smaller meaningful pieces (simpler)
  - Modularity – Attributes that have difference timelines must be separated
  - Encapsulation
    - Components handle their own data and function
- How to handle group of components

The Application: Add Product

Action



Add Product

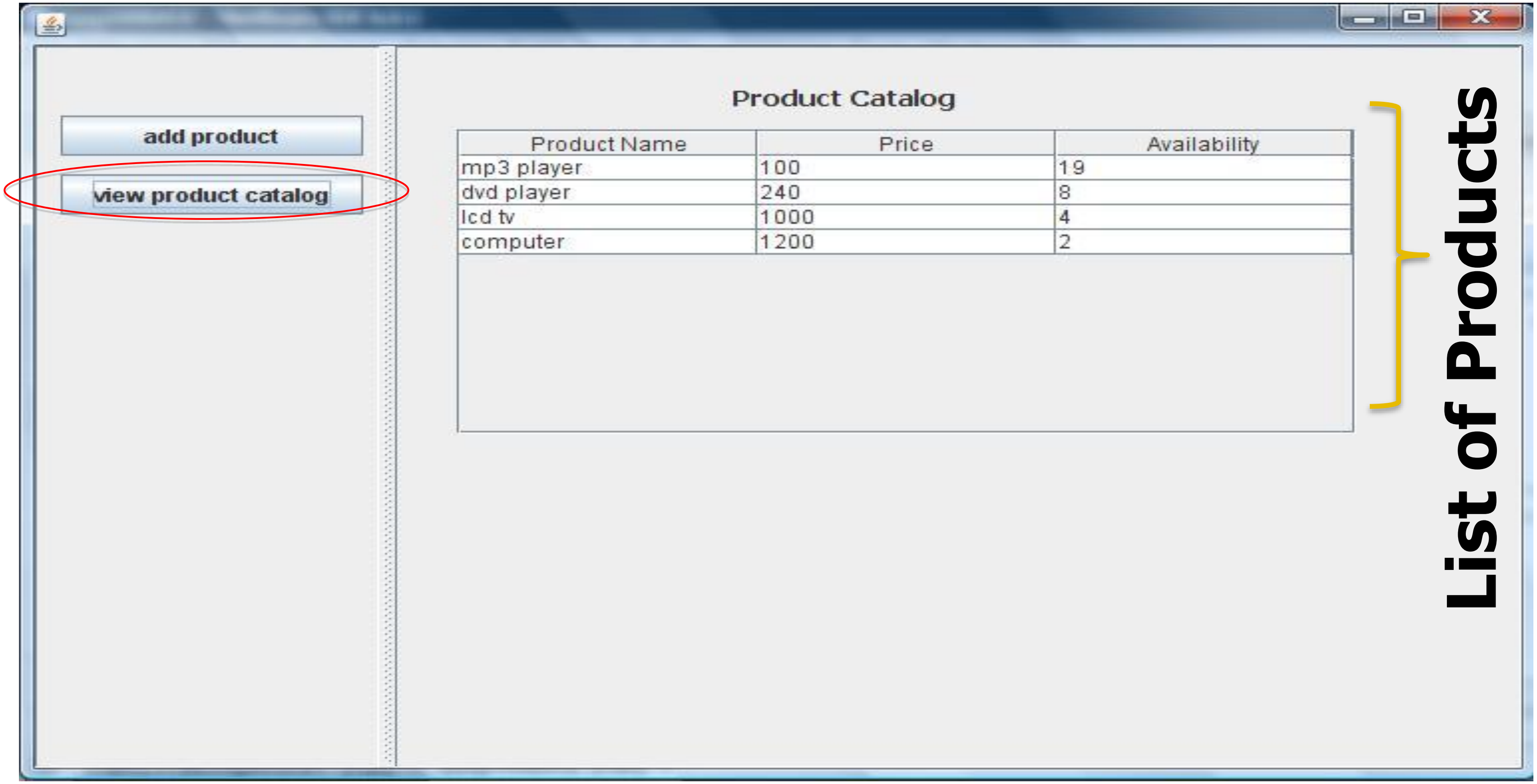
Product Name:

Product Price:

Product Availability:

Product

The Application: Browse products



Product Catalog

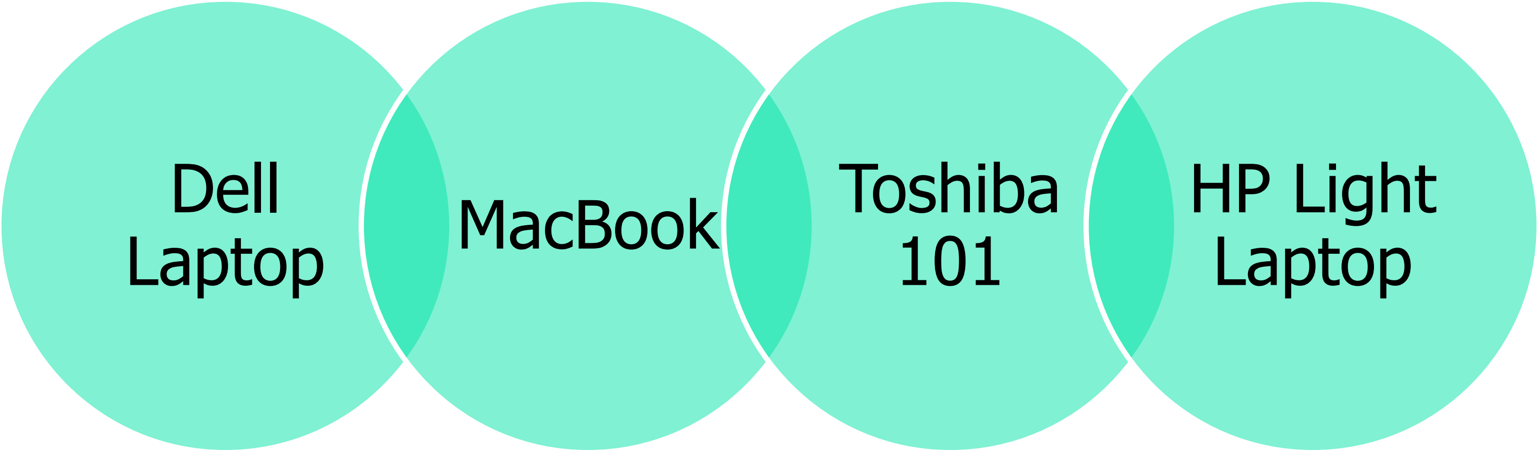
Product Name	Price	Availability
mp3 player	100	19
dvd player	240	8
lcd tv	1000	4
computer	1200	2

List of Products

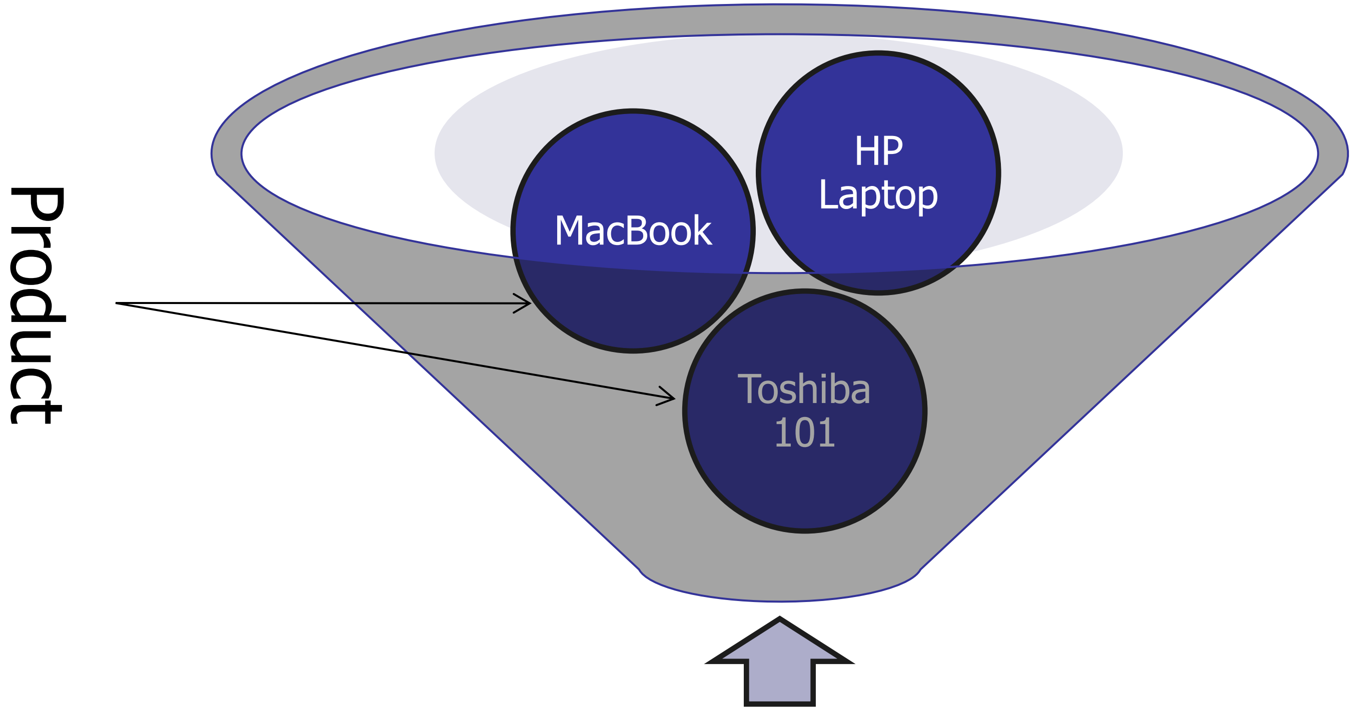
Key Question: How to organize the products?

- We know how to keep track of a single product through a reference variable, but what if we have many products?
- Where to keep the products?
- How find an existing product?
- How to list them?

The answer: Build an information model first



We have a collection of products: The product catalog



Product

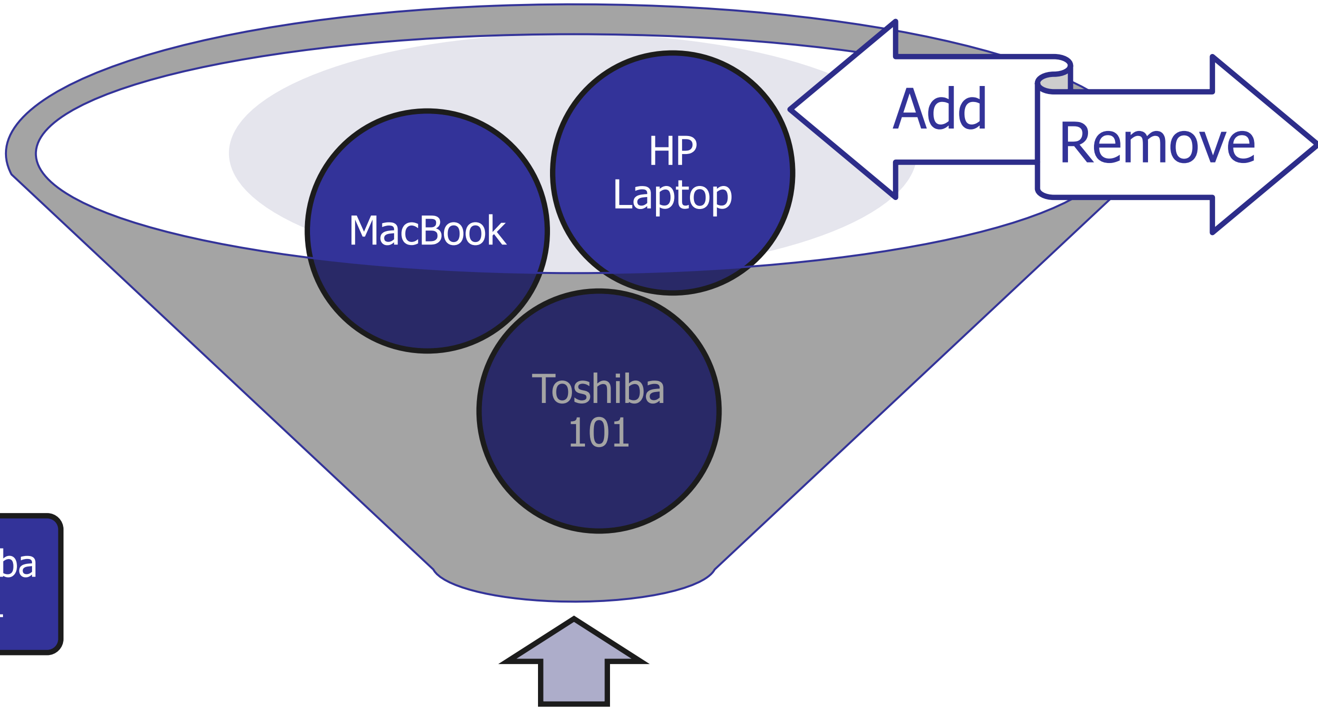
MacBook

HP Laptop

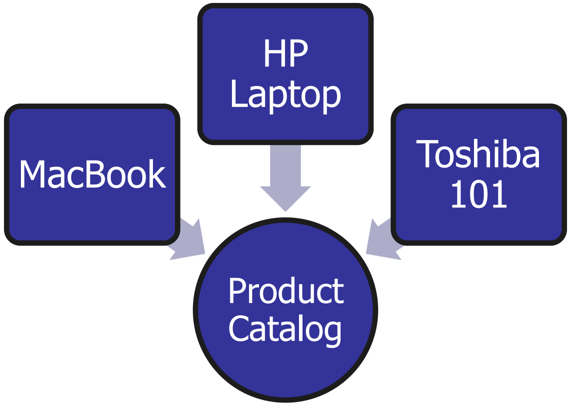
Toshiba 101

Product Catalog is the
container of the products

Operations on product catalog



Product Catalog



So what is the information model for the product catalog?

Is it

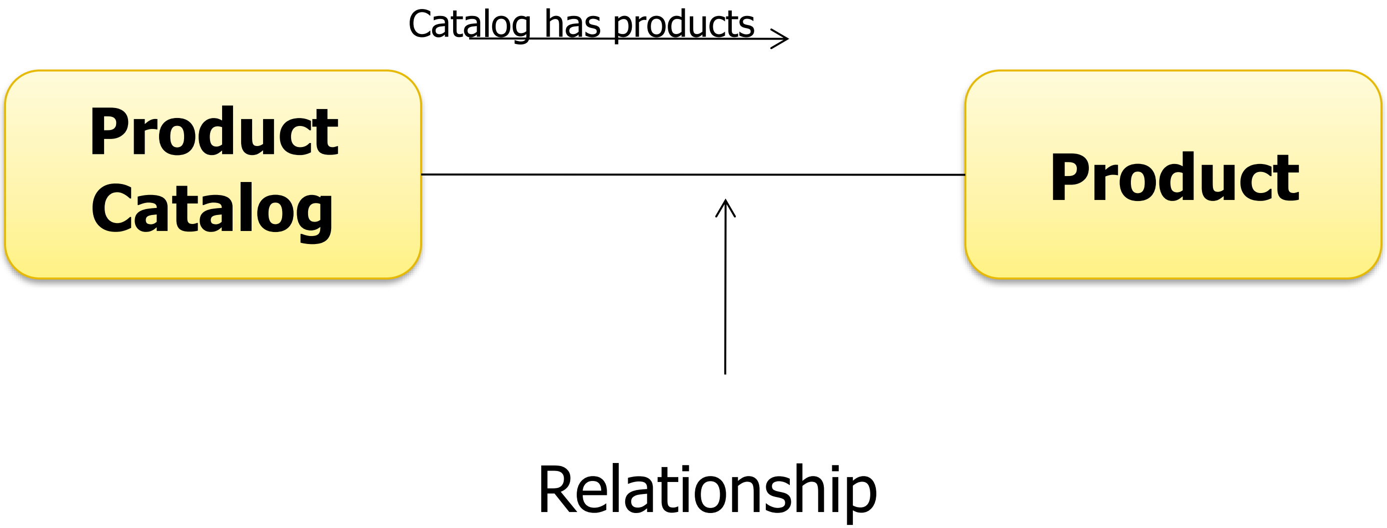
Product Catalog

Or

Product

Product catalog keeps track of products

Or



Catalog has products →

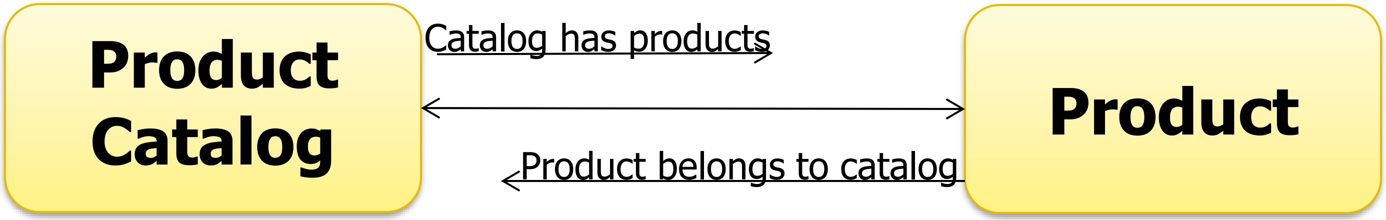
Product Catalog

Product

Relationship

Product catalog keeps track of products

Relationship connections give meaning to concepts:



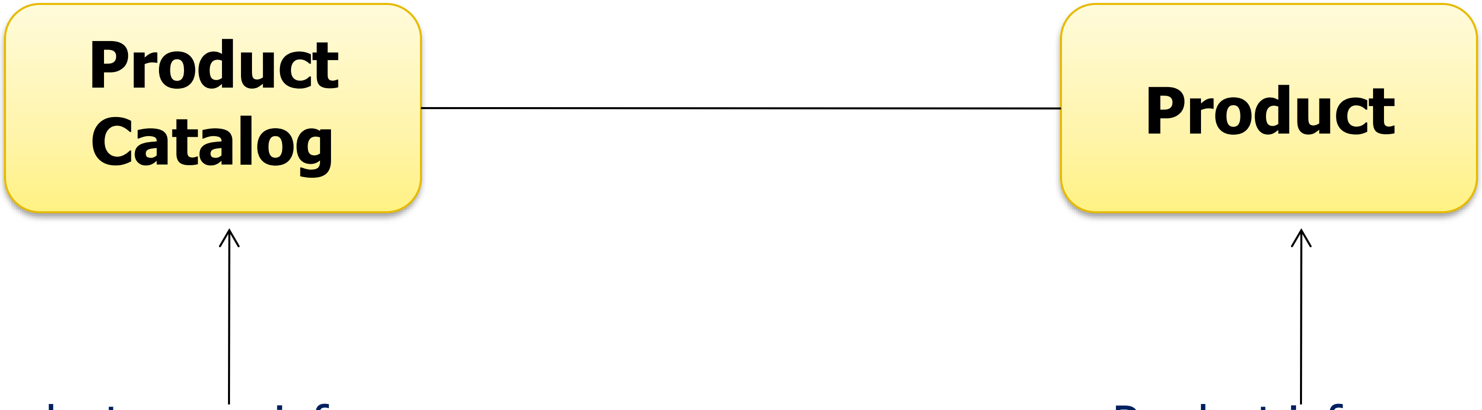
Product Catalog

Catalog has products

Product belongs to catalog

Product

Product catalog keeps track of products (manages products)



Product Catalog

Product

Product group info goes here

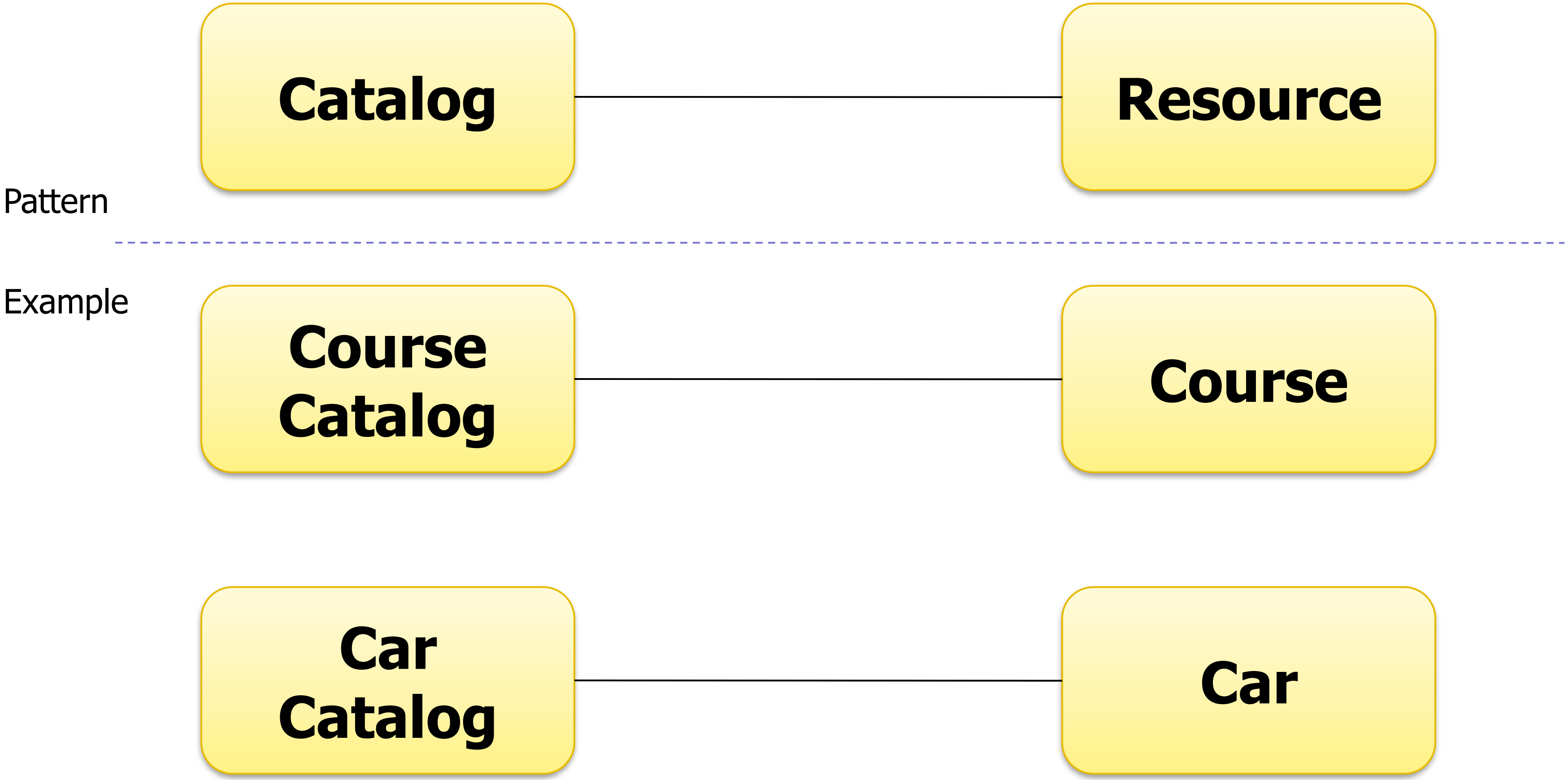
Product info goes here

Group specific and does not care about details of individual products: Its <u>responsibilities</u> include:

1) Creating new products
2) Add a product to the current list
3) Find and remove from the list
4) Find and update a specific product

Specific to a product like its price, avail, desc, etc.

Other Catalog Patterns (AKA factory pattern)



Catalog

Resource

Pattern

Example

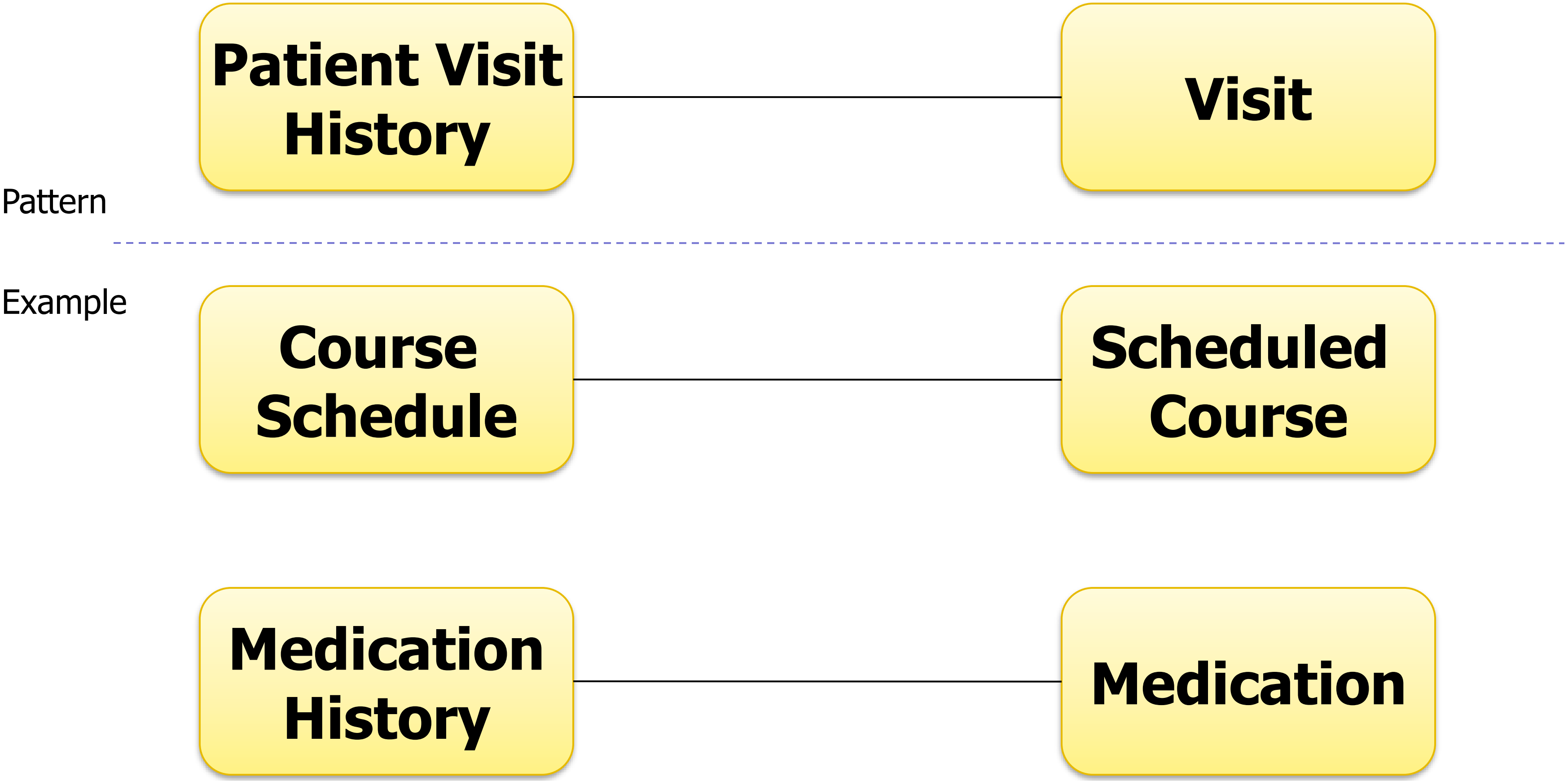
Course Catalog

Course

Car Catalog

Car

Other Catalog Patterns (AKA factory pattern)



Patient Visit History

Visit

Pattern

Example

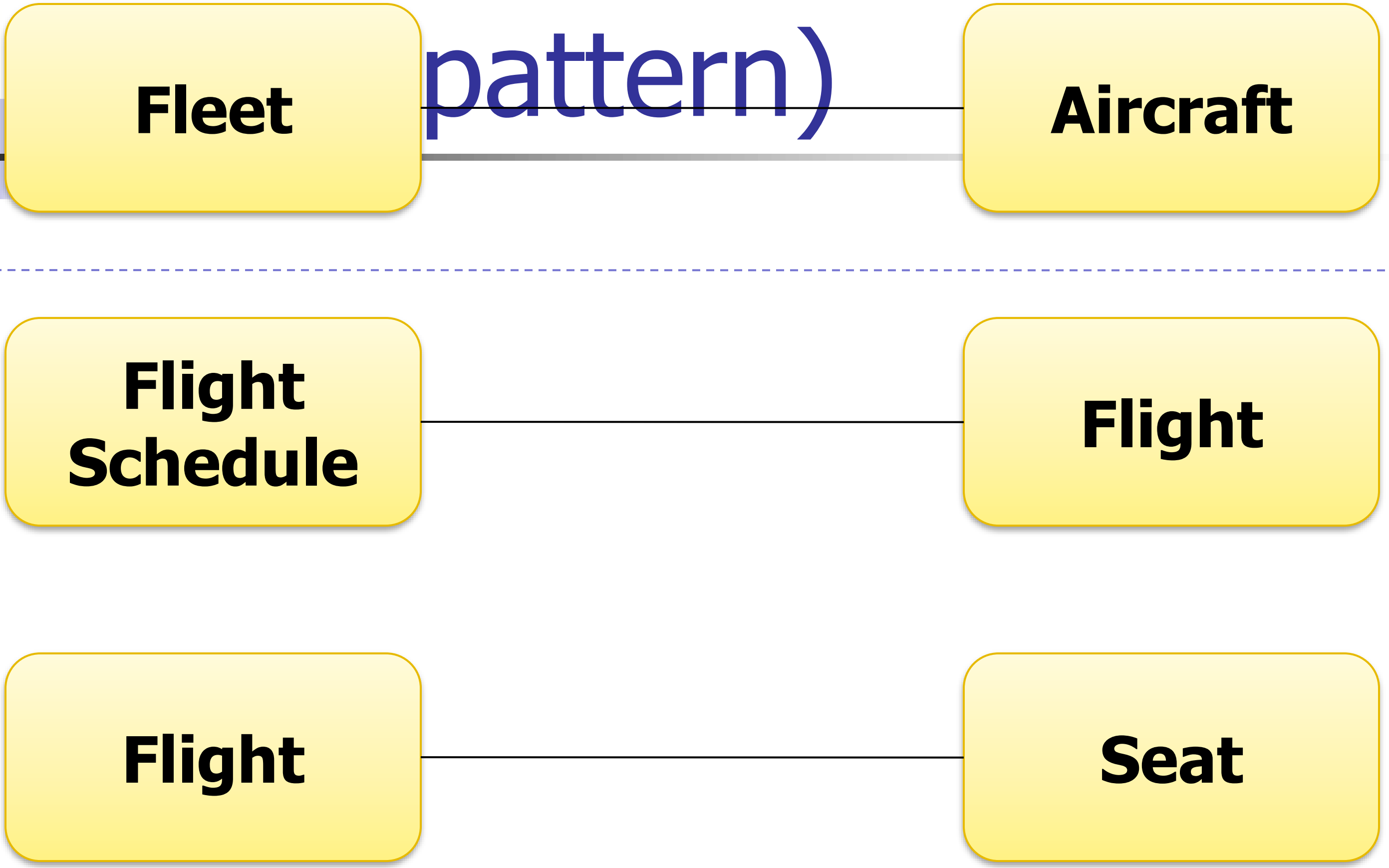
Course Schedule

Scheduled Course

Medication History

Medication

Other Catalog Patterns (AKA pattern)



Fleet

Aircraft

Pattern

Flight Schedule

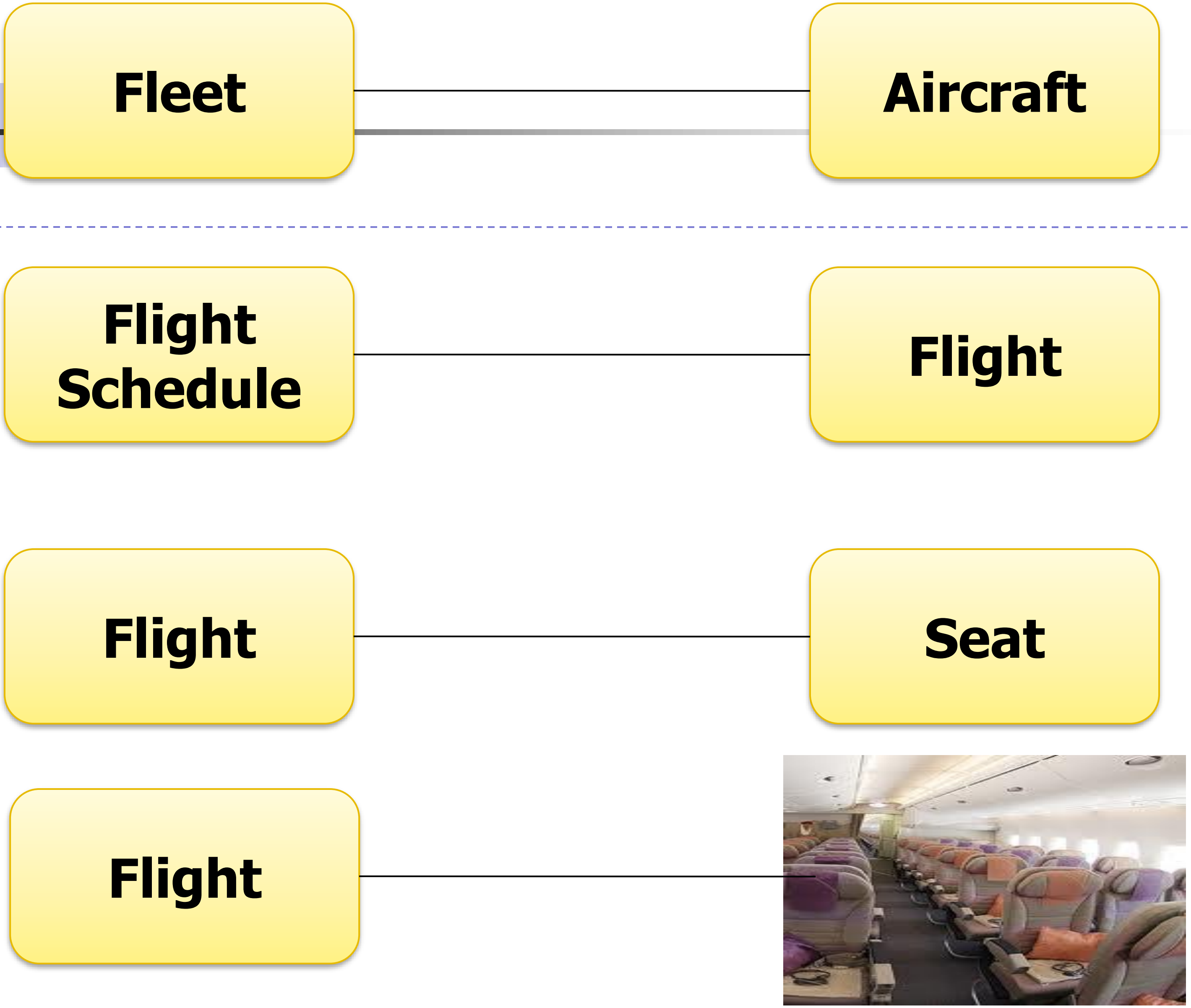
Flight

Example

Flight

Seat

Flight



Fleet

Aircraft

Pattern

Flight Schedule

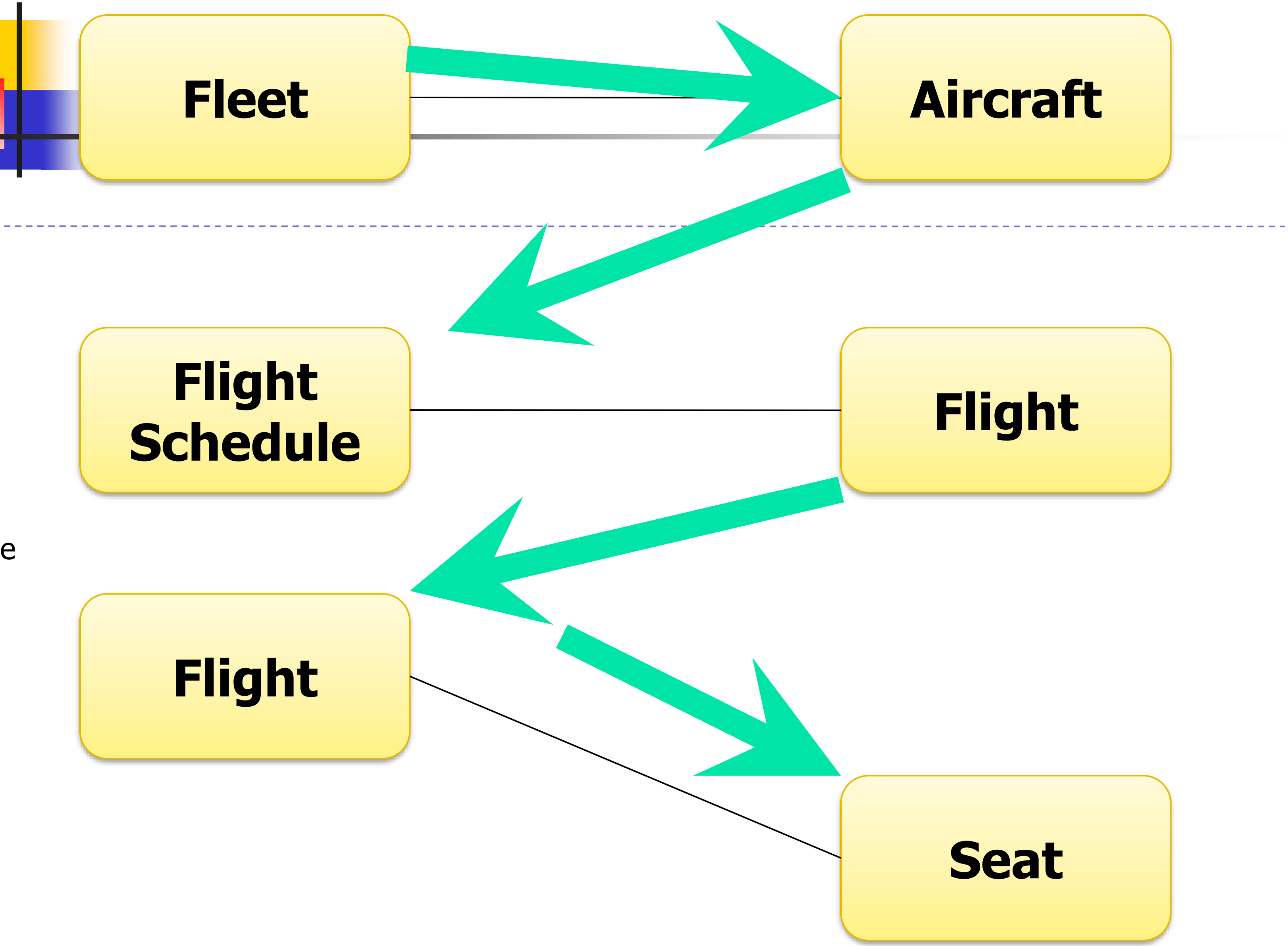
Flight

Example

Flight

Seat

Flight



Fleet

Aircraft

Flight Schedule

Flight

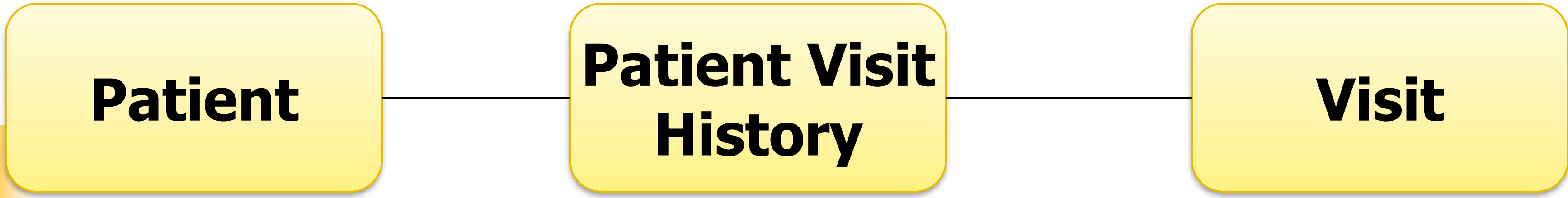
Flight

Seat

Pattern

Example

Model

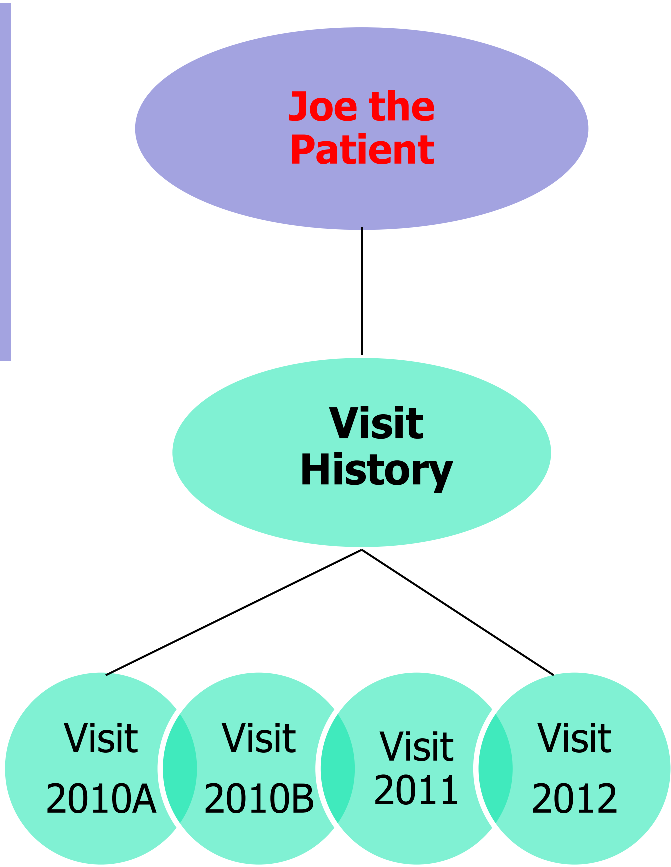


Patient

Patient Visit History

Visit

Instances



Joe the Patient

Visit History

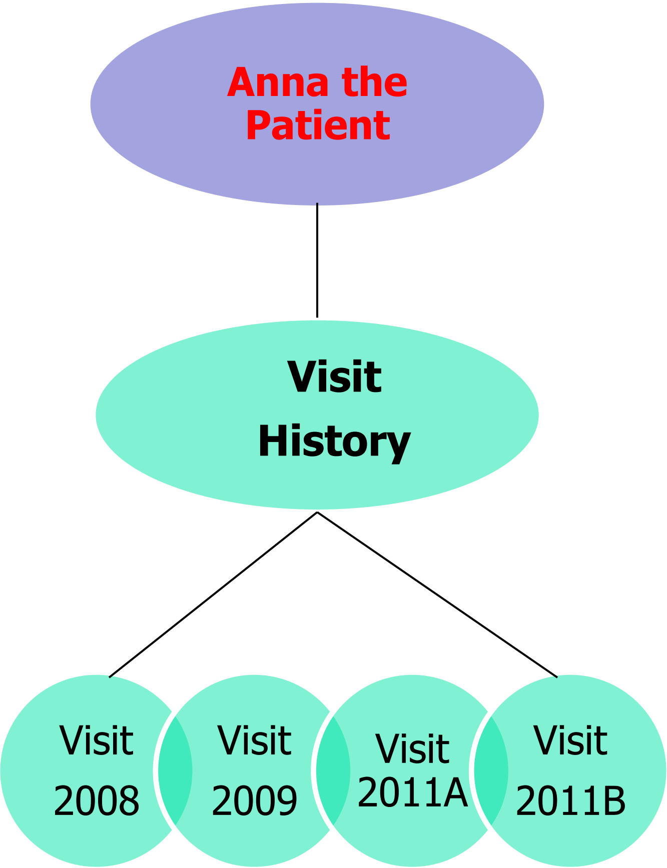
Visit 2010A

Visit 2010B

Visit 2011

Visit 2012

Instances



Anna the Patient

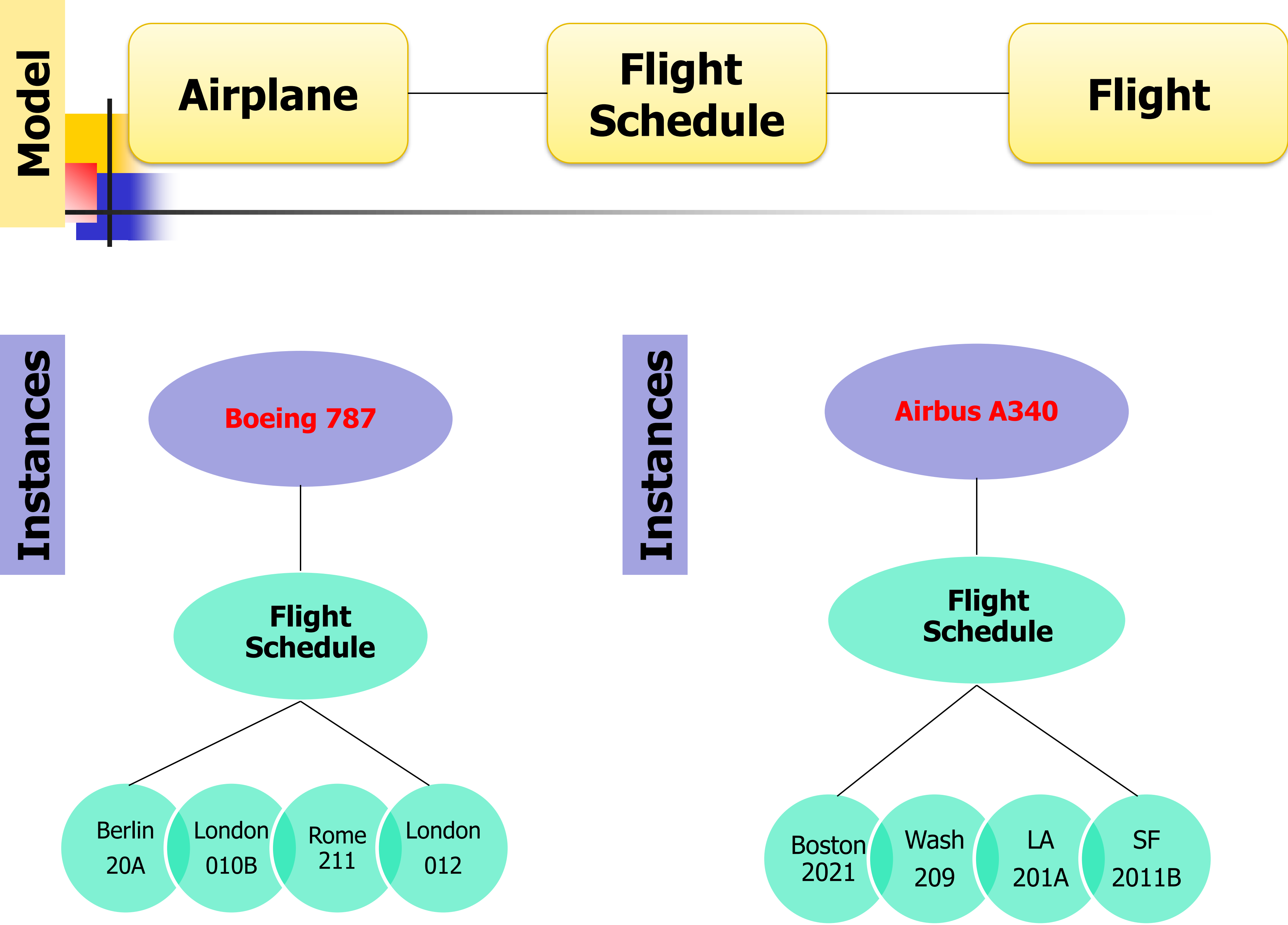
Visit History

Visit 2008

Visit 2009

Visit 2011A

Visit 2011B



Model

Airplane

Flight Schedule

Flight

Instances

Boeing 787

Flight Schedule

Berlin 20A

London 010B

Rome 211

London 012

Instances

Airbus A340

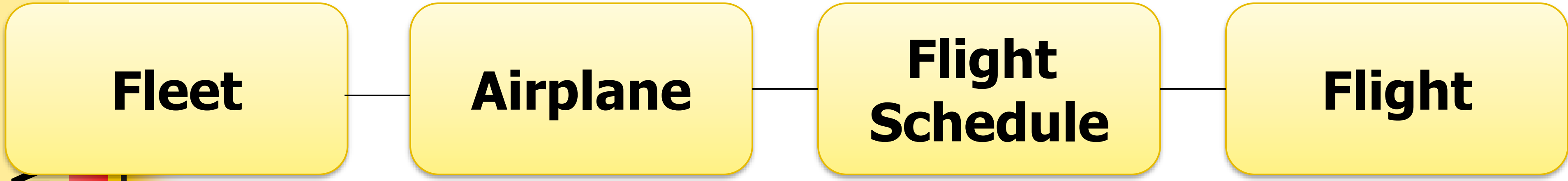
Flight Schedule

Boston 2021

Wash 209

LA 201A

SF 2011B

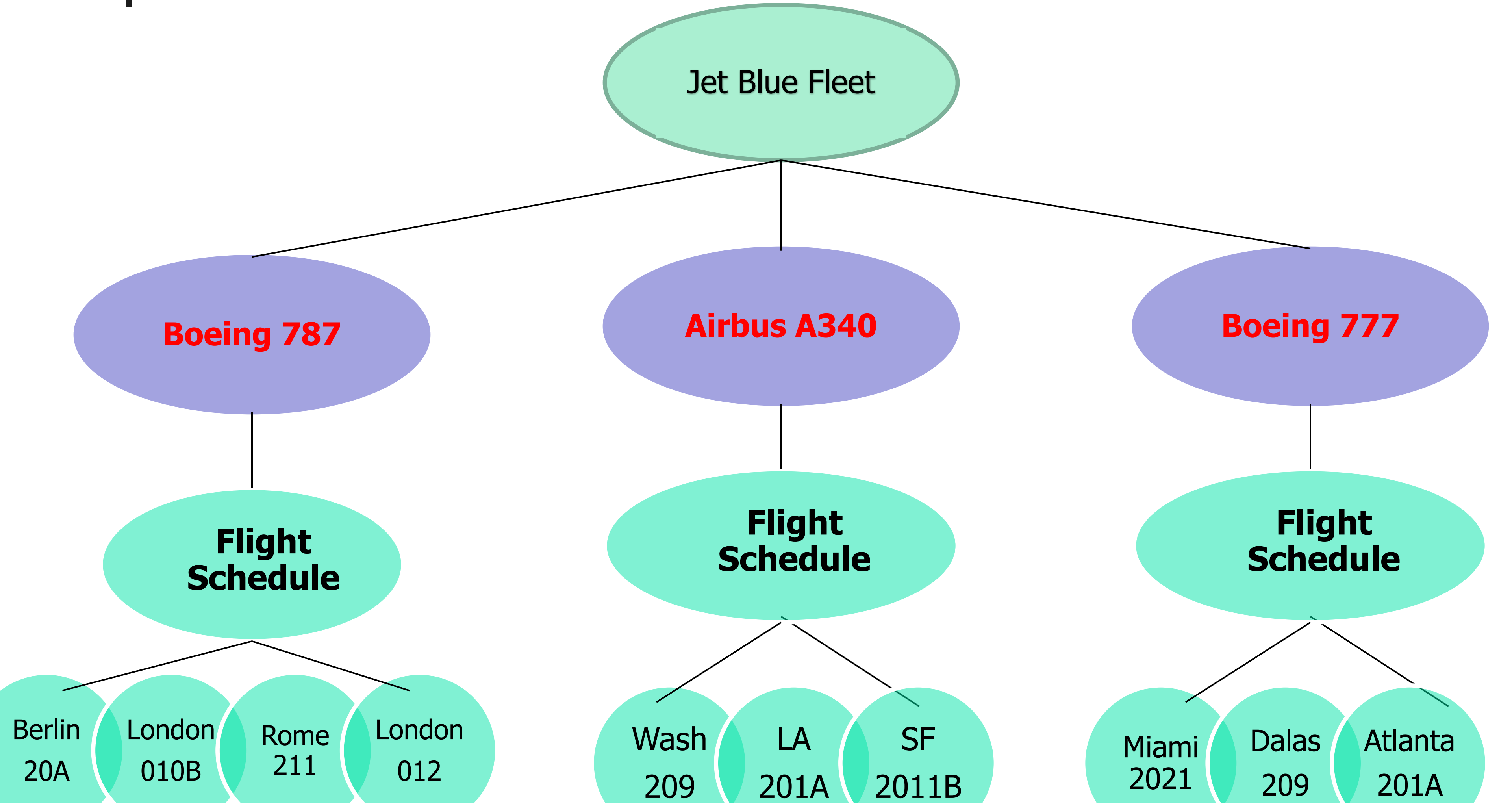


Fleet

Airplane

Flight Schedule

Flight



Jet Blue Fleet

Boeing 787

Airbus A340

Boeing 777

Flight Schedule

Flight Schedule

Flight Schedule

Berlin 20A

London 010B

Rome 211

London 012

Wash 209

LA 201A

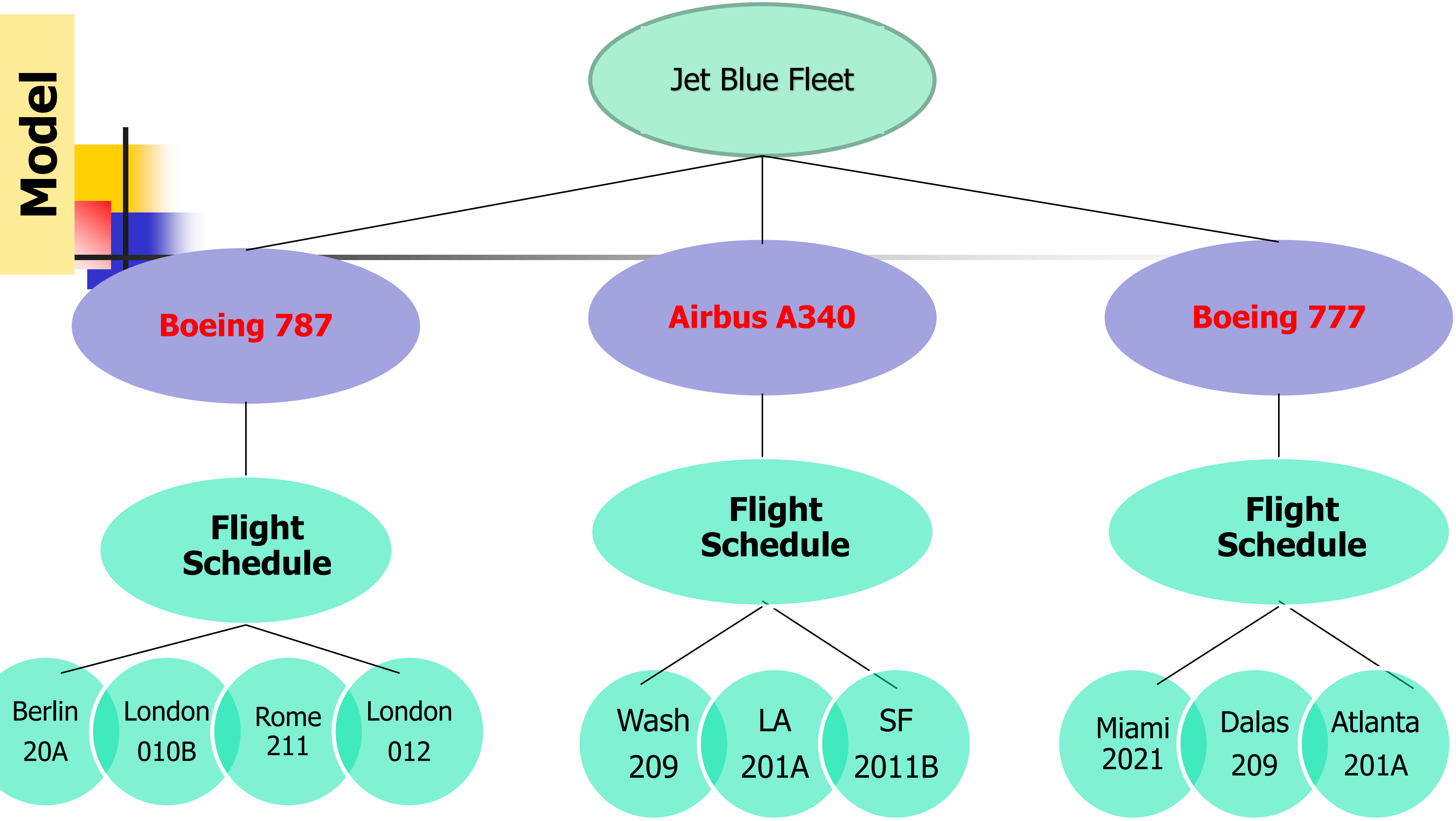
SF 2011B

Miami 2021

Dalas 209

Atlanta 201A

Model



Jet Blue Fleet

Boeing 787

Flight Schedule

Berlin 20A

London 010B

Rome 211

London 012

Airbus A340

Flight Schedule

Wash 209

LA 201A

SF 2011B

Boeing 777

Flight Schedule

Miami 2021

Dalas 209

Atlanta 201A

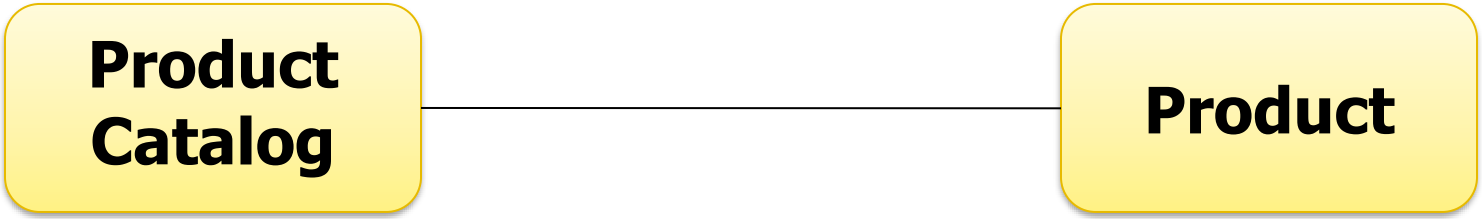
How Java will do this?

- Classes so we define what each class means (for example flight handles the smarts of how to deal with empty and available seats)
- Objects so we fill them with data that distinguish things
- Array Lists to do two things
  - glue objects together

How Java will do this?

- Arrays to do two things
  - glue components together
  - Relate one component to many components
    - For example an array is needed
      - a flight to house many empty seats
      - Medication history to keep track of multiple medications for a patient

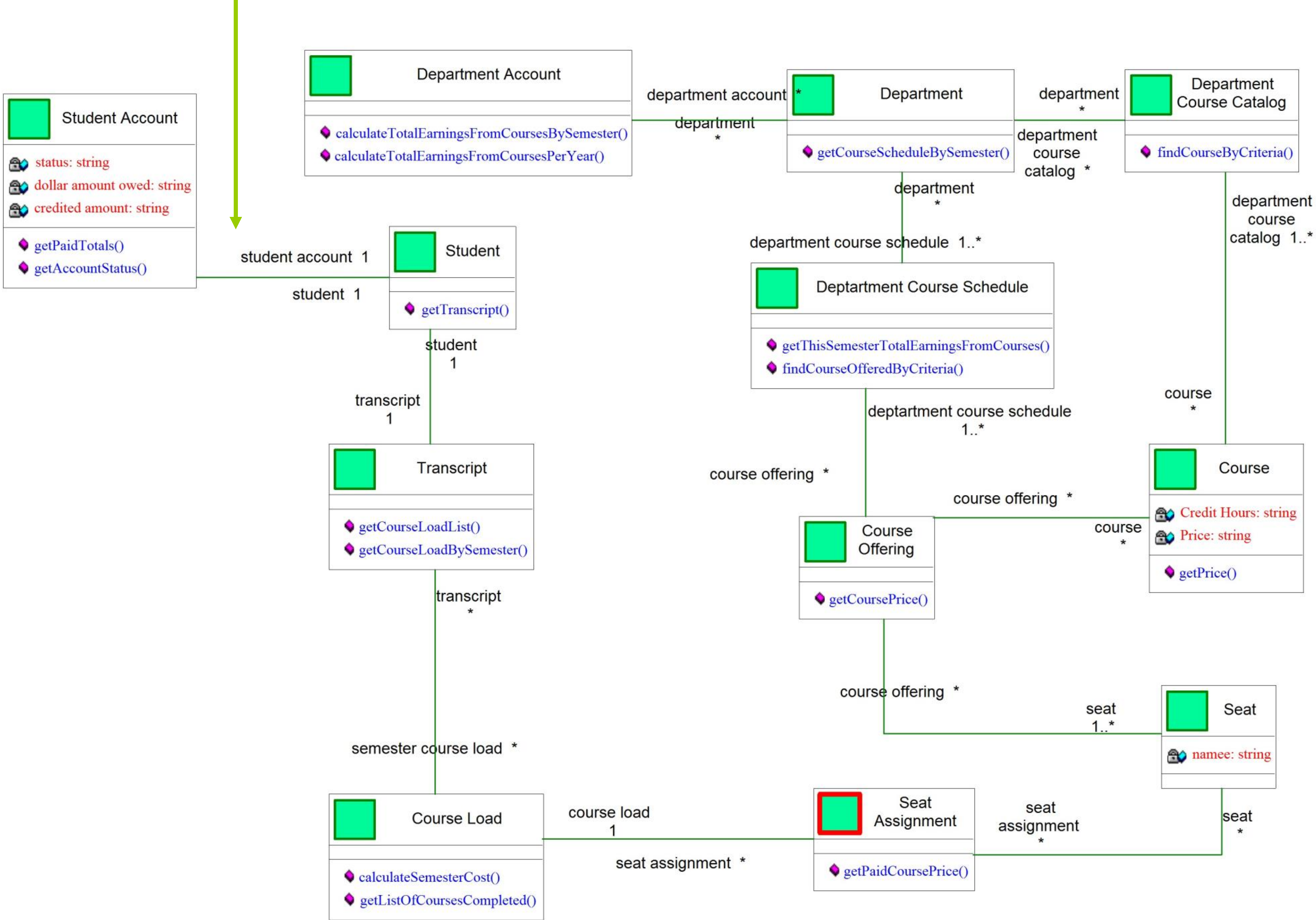
How to implement this pattern in java?



- Define Java class for the product catalog
- Define a java class for product
- The product catalog class must keep track of products
  - How?

Why learning how to implement relationship connections are important?

Student related questions start here



Student Account

status: string

dollar amount owed: string

credited amount: string

getPaidTotals()

getAccountStatus()

Department Account

calculateTotalEarningsFromCoursesBySemester()

calculateTotalEarningsFromCoursesPerYear()

Department

getCourseScheduleBySemester()

Department Course Catalog

findCourseByCriteria()

Student

getTranscript()

Deptartment Course Schedule

getThisSemesterTotalEarningsFromCourses()

findCourseOfferedByCriteria()

Transcript

getCourseLoadList()

getCourseLoadBySemester()

Course

Credit Hours: string

Price: string

getPrice()

Course Offering

getCoursePrice()

Seat

namee: string

Course Load

calculateSemesterCost()

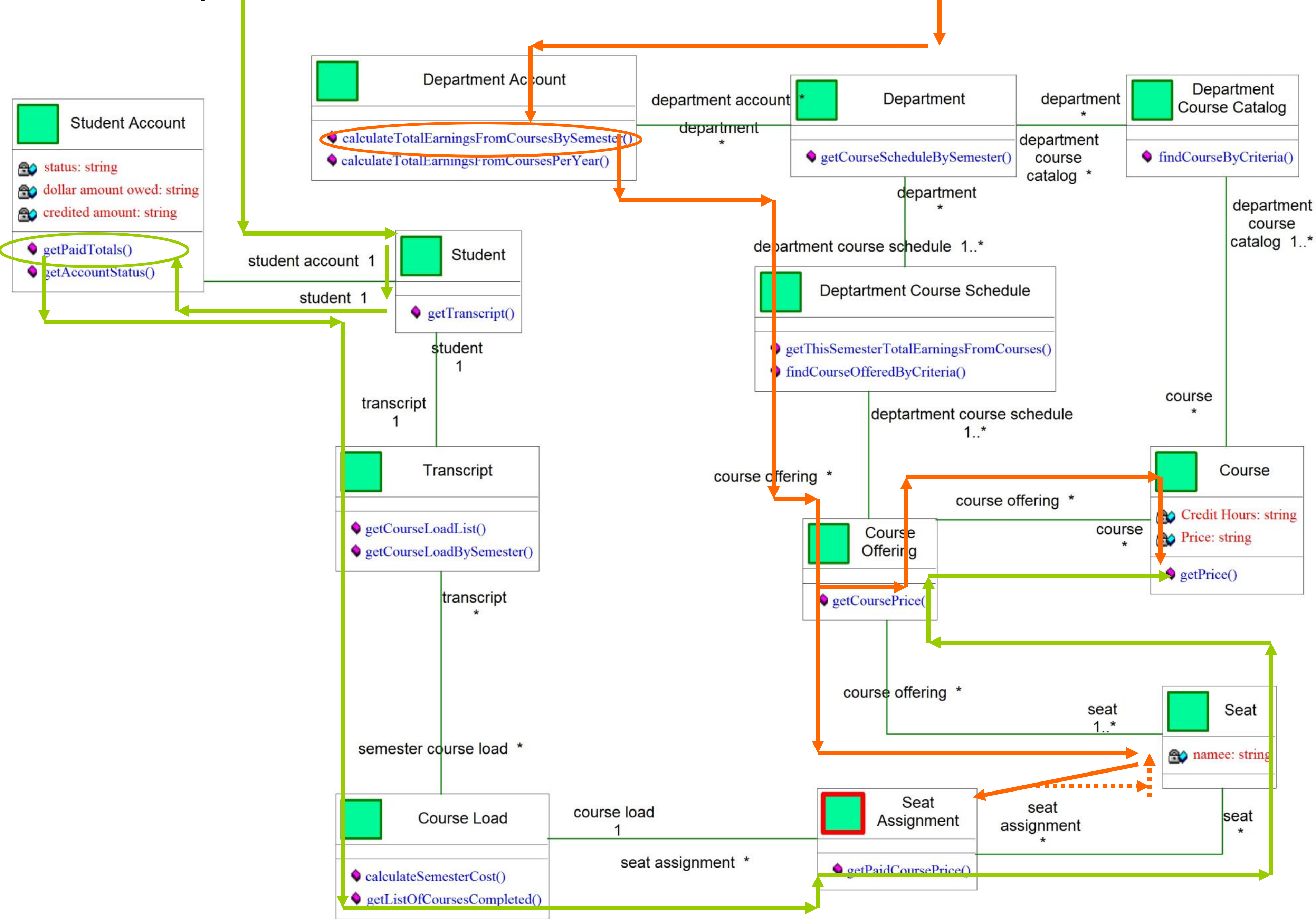
getListOfCoursesCompleted()

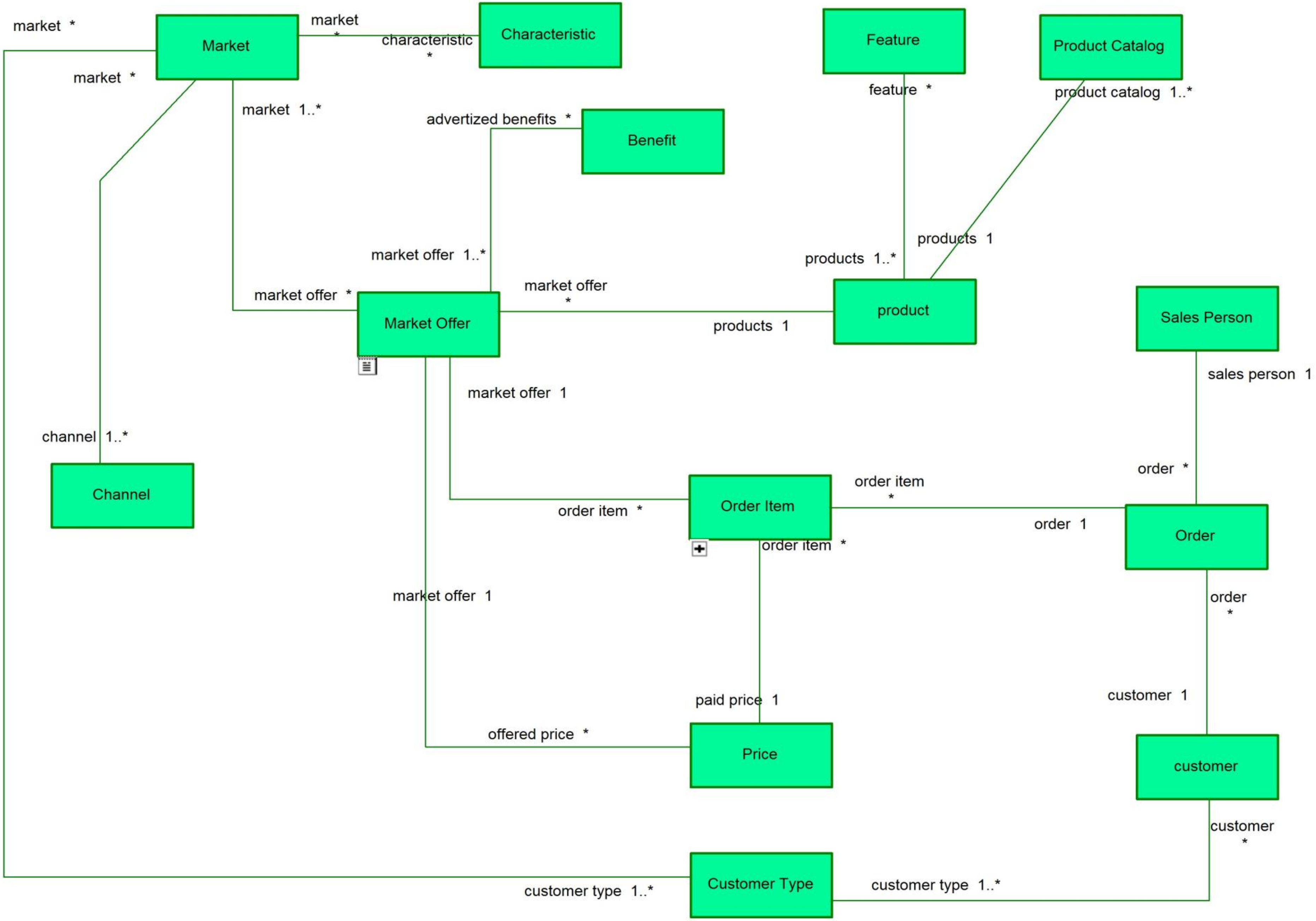
Seat Assignment

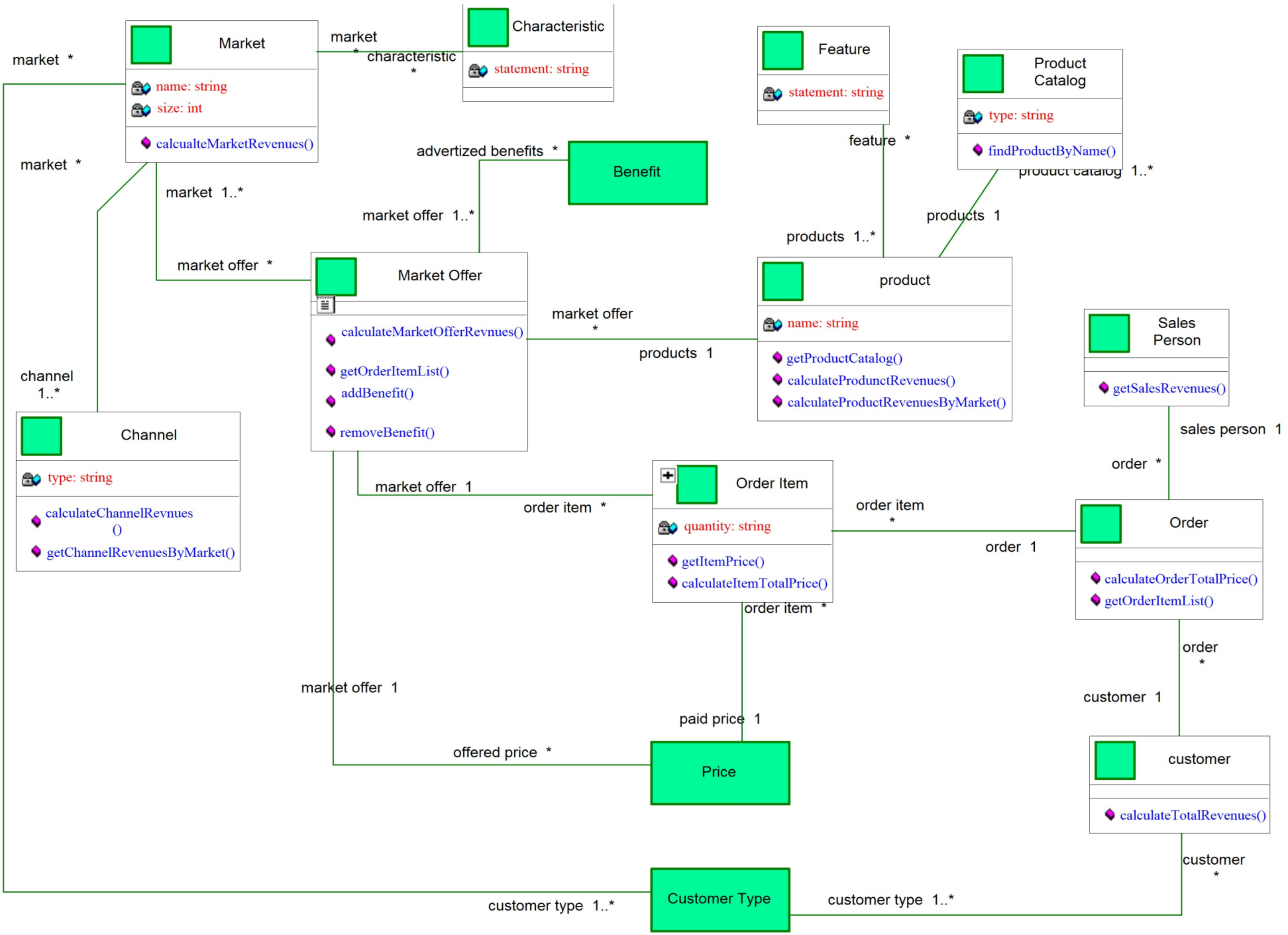
getPaidCoursePrice()

Department related questions start here

Student related questions start here

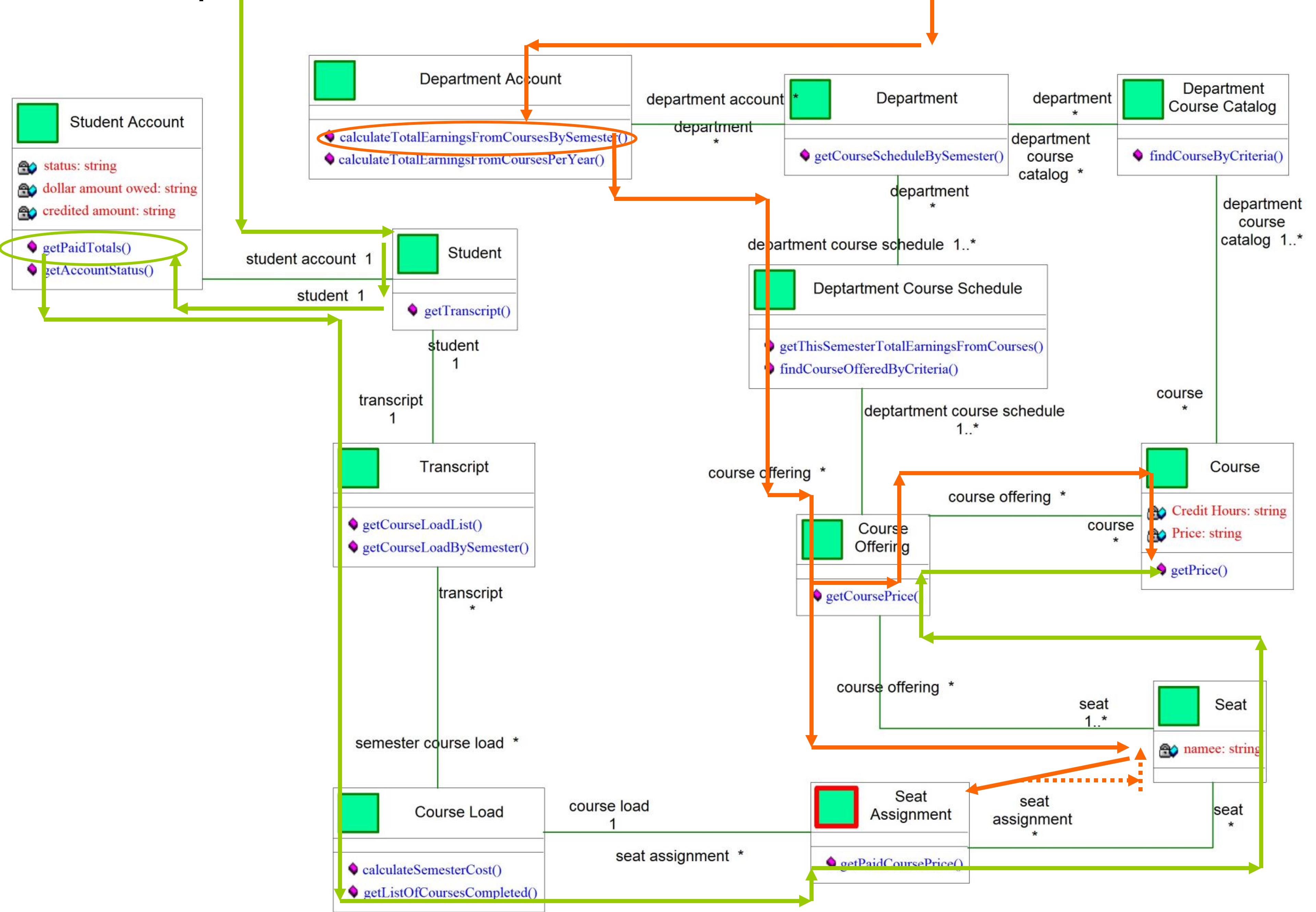



market *
Market
market
*
characteristic
*
Characteristic
Feature
Product Catalog
market *
market 1..*
feature *
product catalog 1..*
advertized benefits *
Benefit
products 1
products 1..*
market offer 1..*
market offer
*
market offer *
Market Offer
product
products 1
Sales Person
market offer 1
sales person 1
channel 1..*
order *
Channel
order item
*
order item *
Order Item
order 1
Order
order item *
market offer 1
order
*
customer 1
paid price 1
offered price *
Price
customer
customer
*
Customer Type
customer type 1..*
customer type 1..*


Market
name: string
size: int
calcualteMarketRevenues()
Characteristic
statement: string
market *
characteristic *
Feature
statement: string
Product Catalog
type: string
findProductByName()
market *
market *
market 1..*
advertized benefits *
Benefit
market offer 1..*
feature *
product catalog 1..*
products 1
products 1..*
market offer *
Market Offer
calculateMarketOfferRevnues()
getOrderItemList()
addBenefit()
removeBenefit()
market offer *
product
name: string
getProductCatalog()
calculateProdunctRevenues()
calculateProductRevenuesByMarket()
products 1
Sales Person
getSalesRevenues()
channel 1..*
Channel
type: string
calculateChannelRevnues ()
getChannelRevenuesByMarket()
sales person 1
order *
market offer 1
order item *
Order Item
quantity: string
getItemPrice()
calculateItemTotalPrice()
order item *
Order
calculateOrderTotalPrice()
getOrderItemList()
order 1
order item *
order *
customer 1
market offer 1
paid price 1
offered price *
Price
customer
calculateTotalRevenues()
customer *
Customer Type
customer type 1..*
customer type 1..*

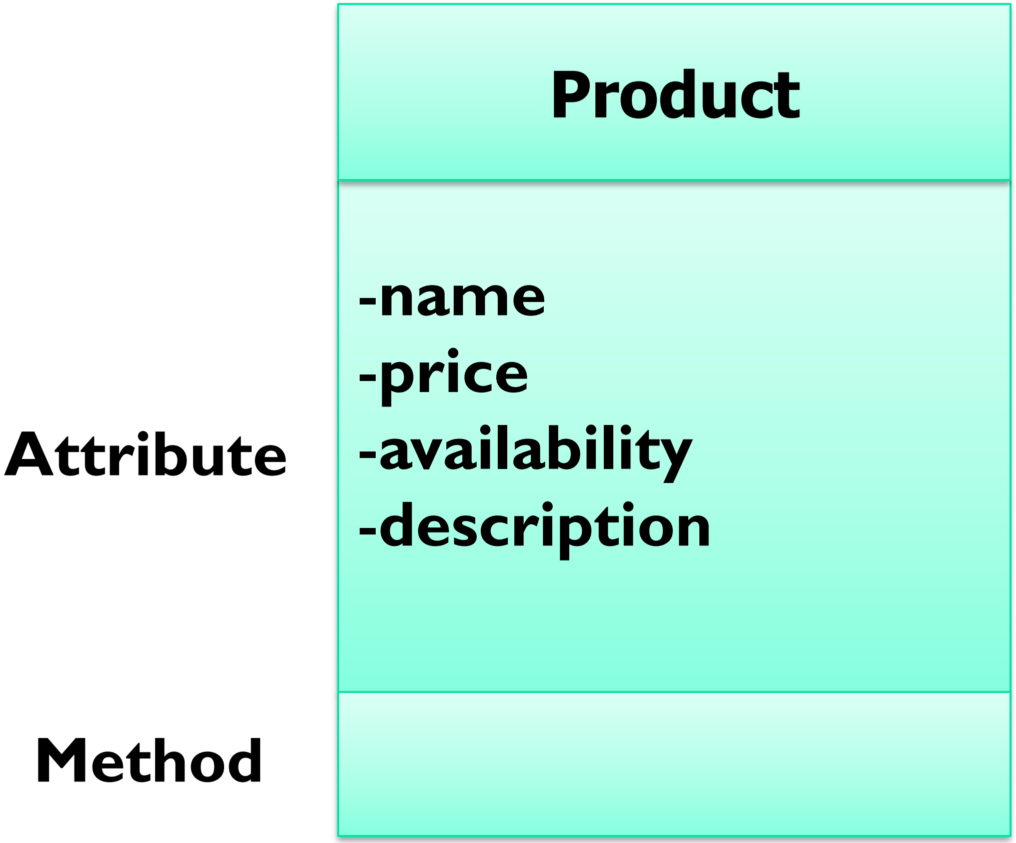
Department related questions start here

Student related questions start here



Create Product Class

under the business package



Attribute

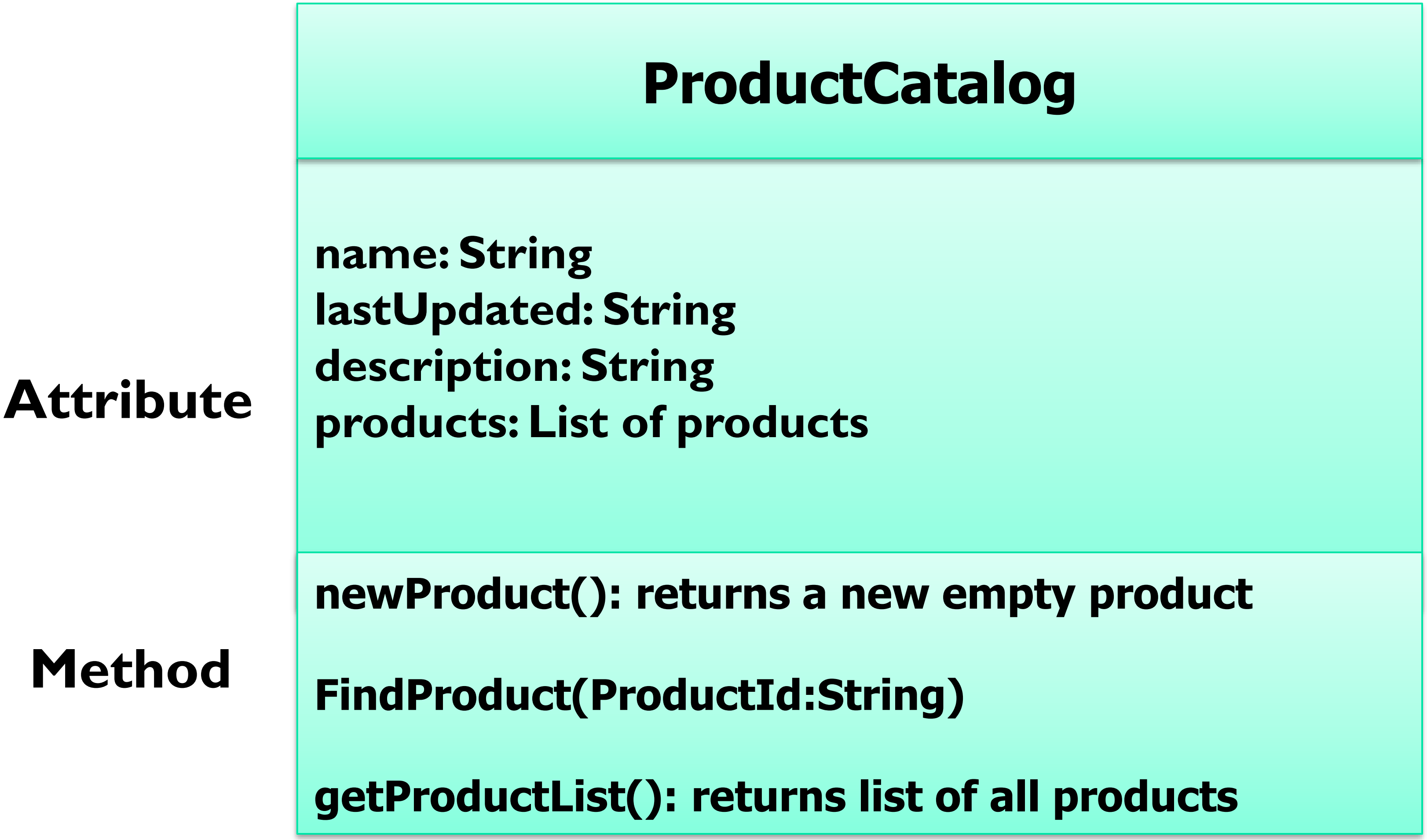
Method

Product

-name
-price
-availability
-description

ProductCatalog Class

under the business package



Attribute

Method

ProductCatalog

name: String
lastUpdated: String
description: String
products: List of products

newProduct(): returns a new empty product

FindProduct(ProductId:String)

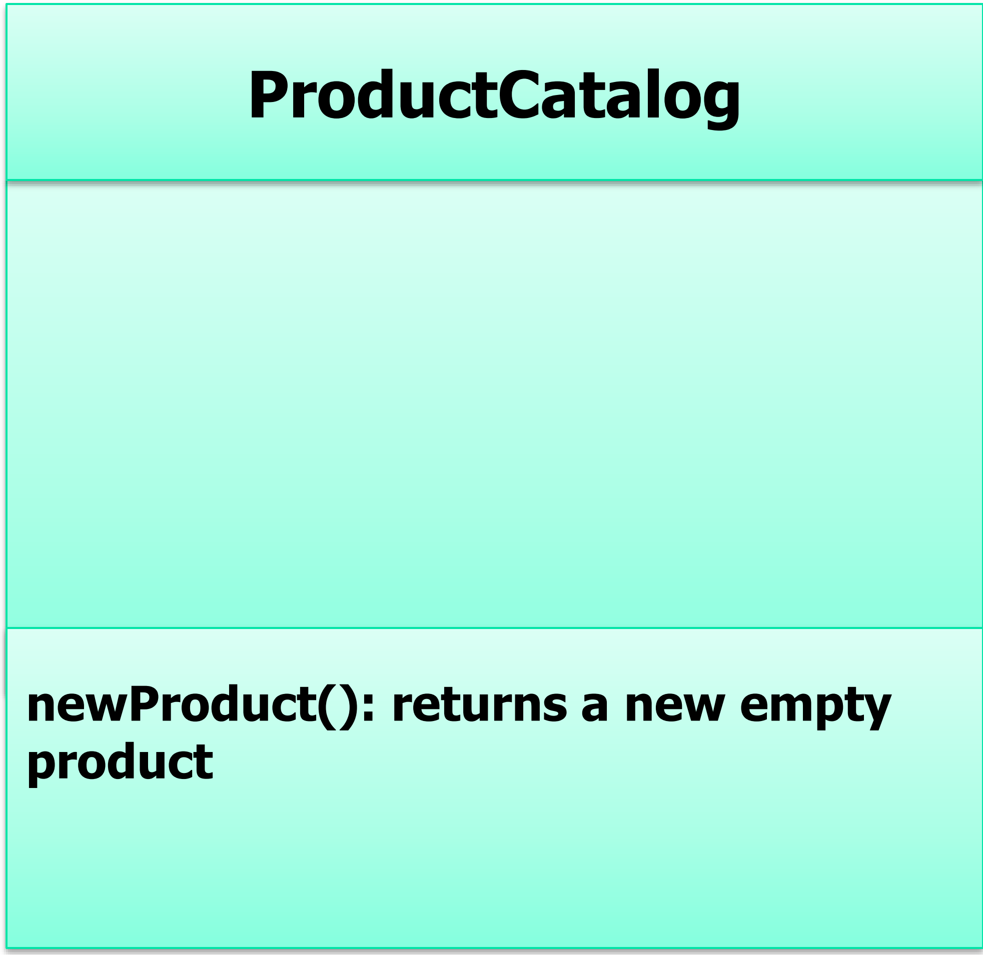
getProductList(): returns list of all products

ProductCatalog Class

under the business package

The newProduct() method does the following:

1. **Uses the java new operator to create a product object**
2. **Saves internally as part of a list**
3. **Returns the obejct to the caller (requester)**



ProductCatalog

newProduct(): returns a new empty product

How the product catalog will flow through the screens?

When the mainjframe is first executed, we create an object of type productcatalog

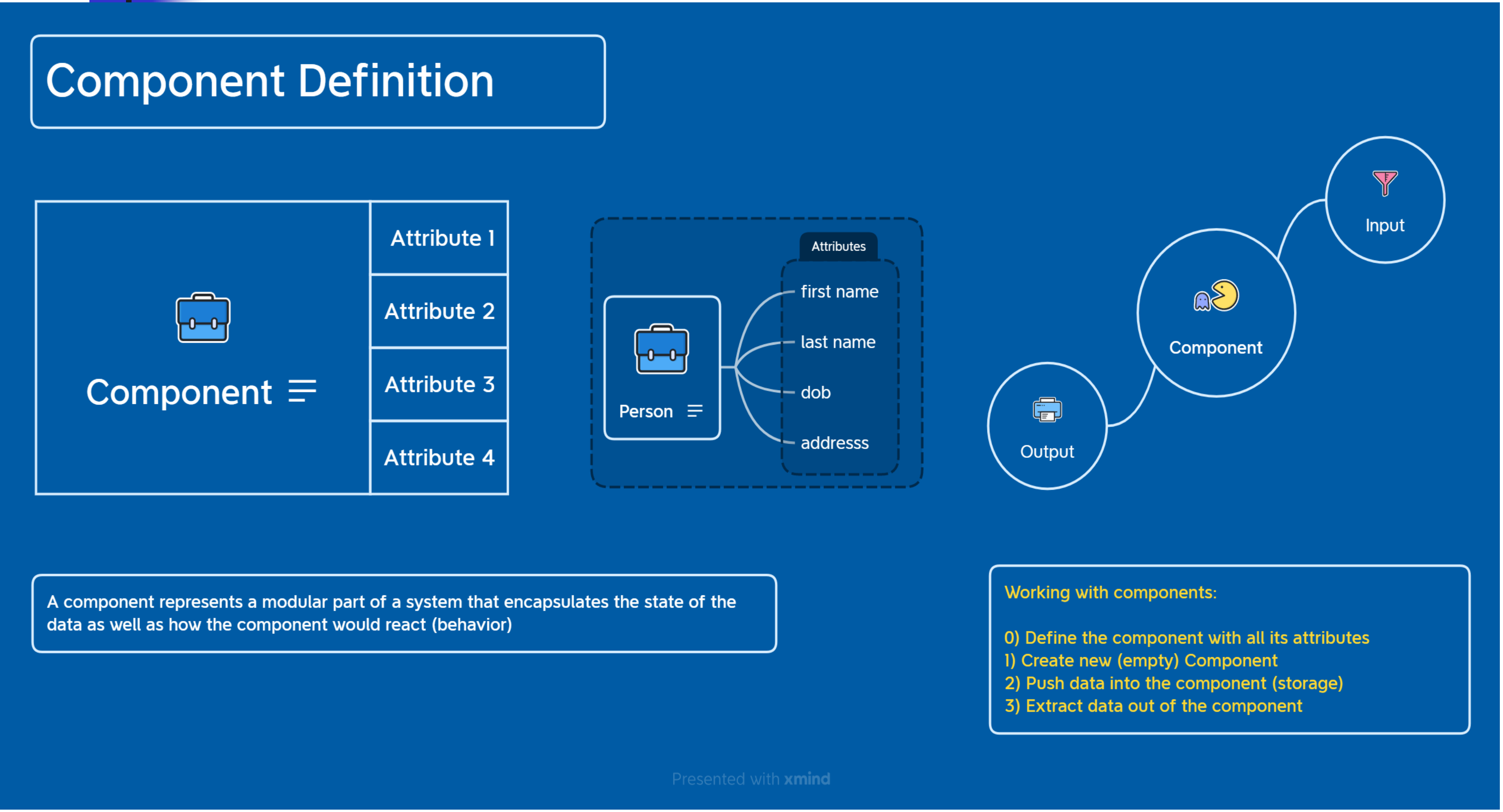
We keep the product catalog object in the MainJFrame for the duration of the application

When user wants to add a product we send the product catalog object to the add product screen

The product screen will use the product catalog object to create new product and fill it with input from the user.

The catalog should how to save the newly created product in its list of products

Aspects of Components



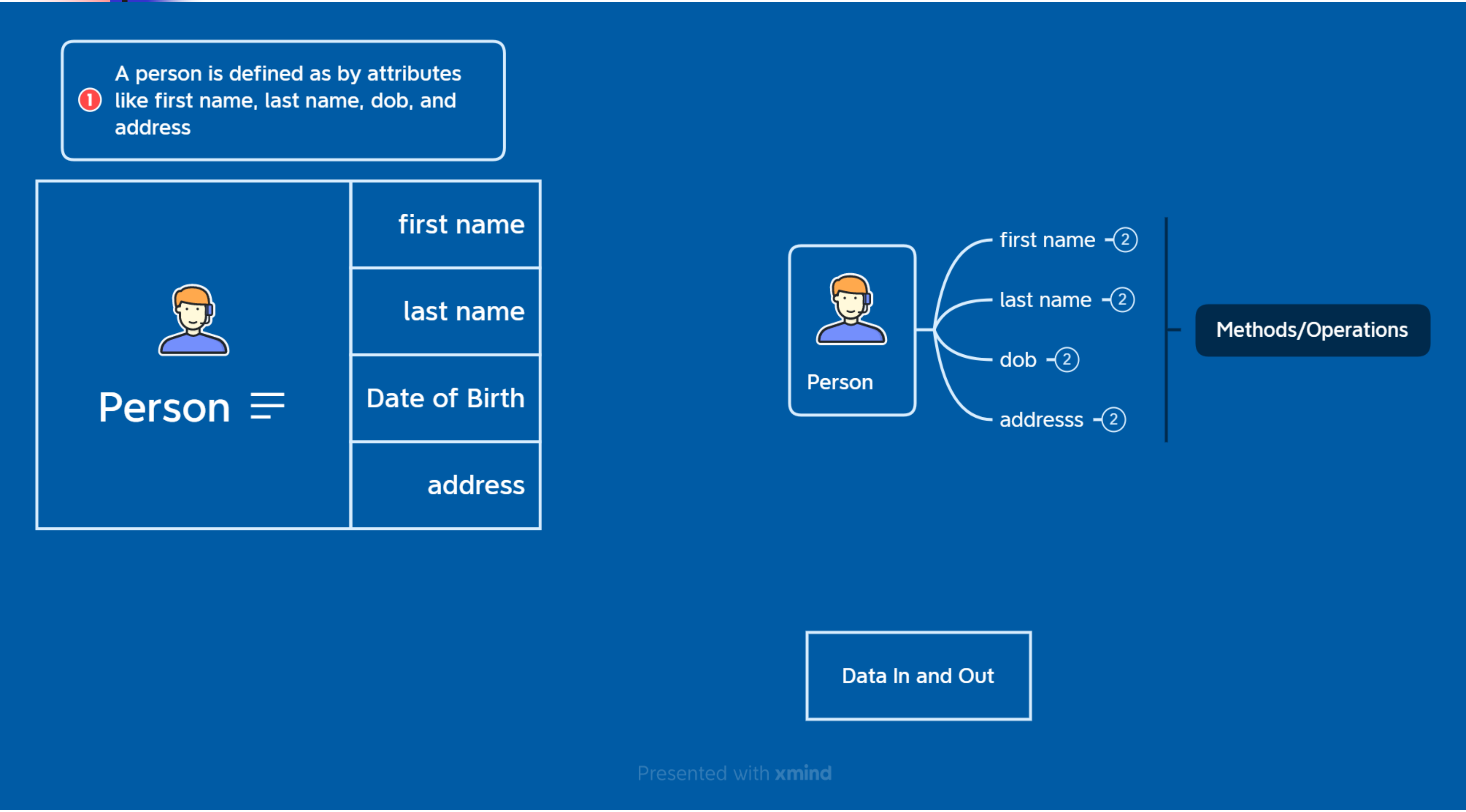
Component Definition

A component represents a modular part of a system that encapsulates the state of the data as well as how the component would react (behavior)

Working with components:

0) Define the component with all its attributes
1) Create new (empty) Component
2) Push data into the component (storage)
3) Extract data out of the component

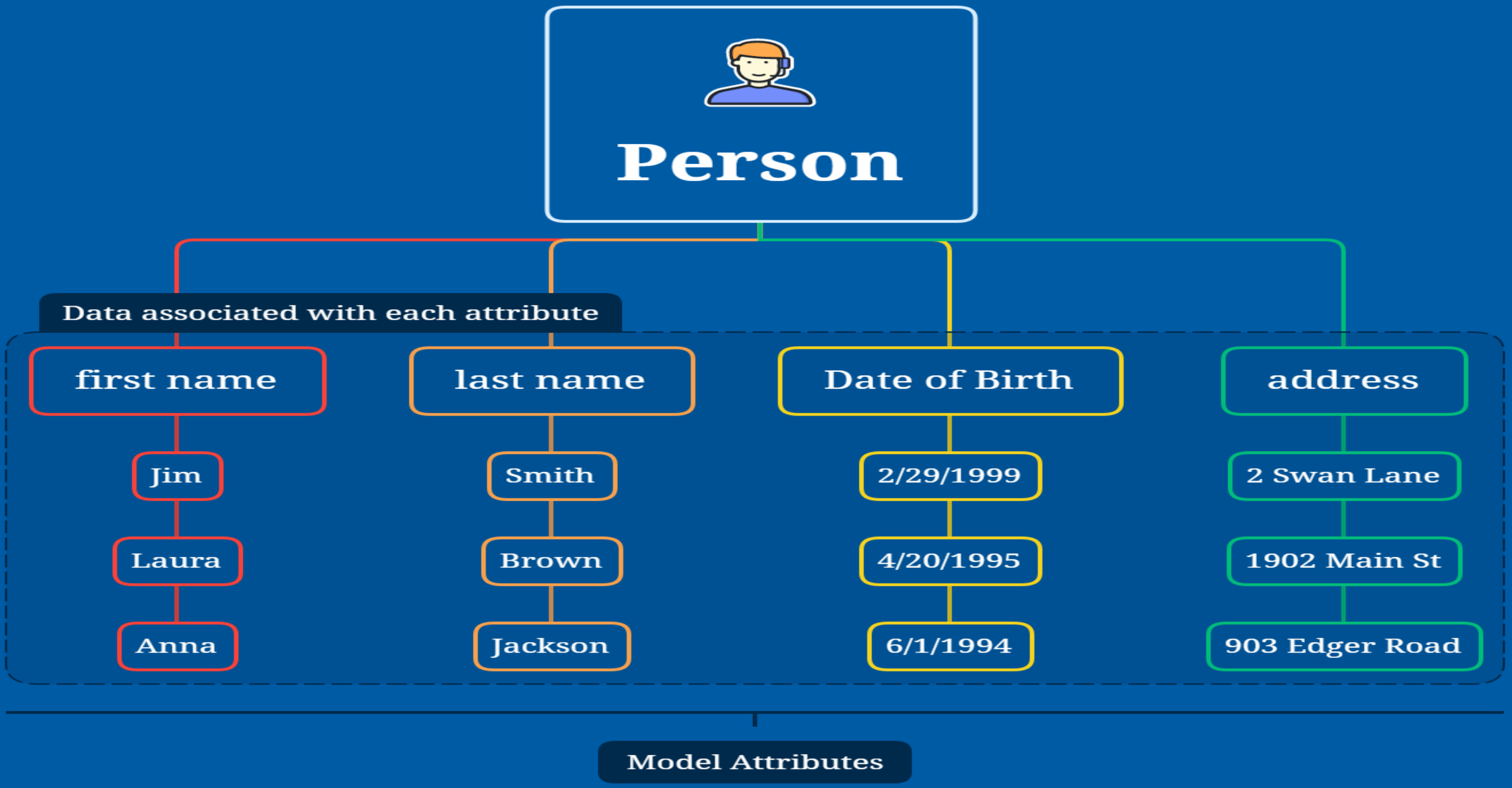
Component Example



1 A person is defined as by attributes like first name, last name, dob, and address

Person	first name
	last name
	Date of Birth
	address

Component (class) with many instances



Person

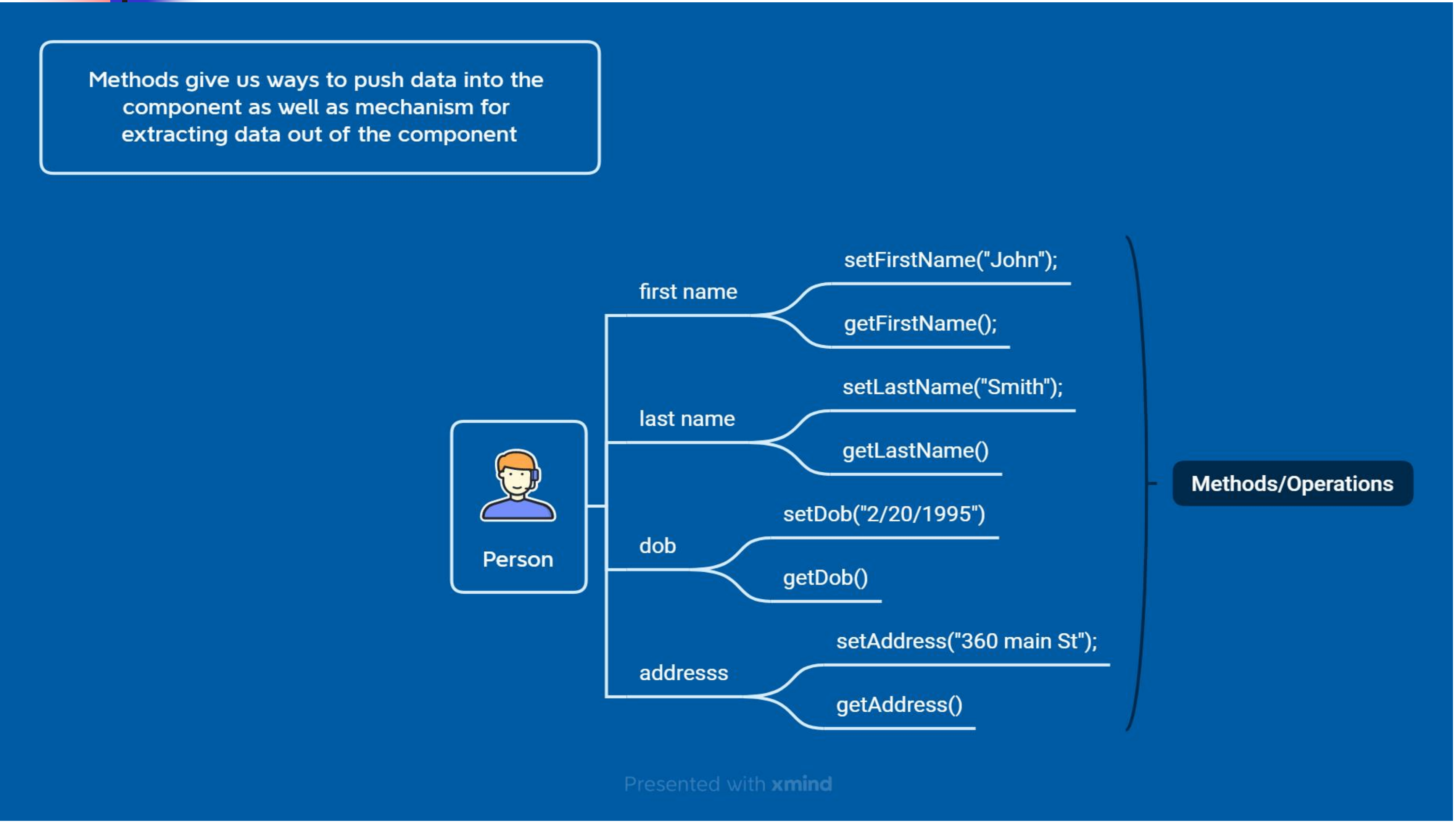
Data associated with each attribute

first name	last name	Date of Birth	address
Jim	Smith	2/29/1999	2 Swan Lane
Laura	Brown	4/20/1995	1902 Main St
Anna	Jackson	6/1/1994	903 Edger Road

Model Attributes

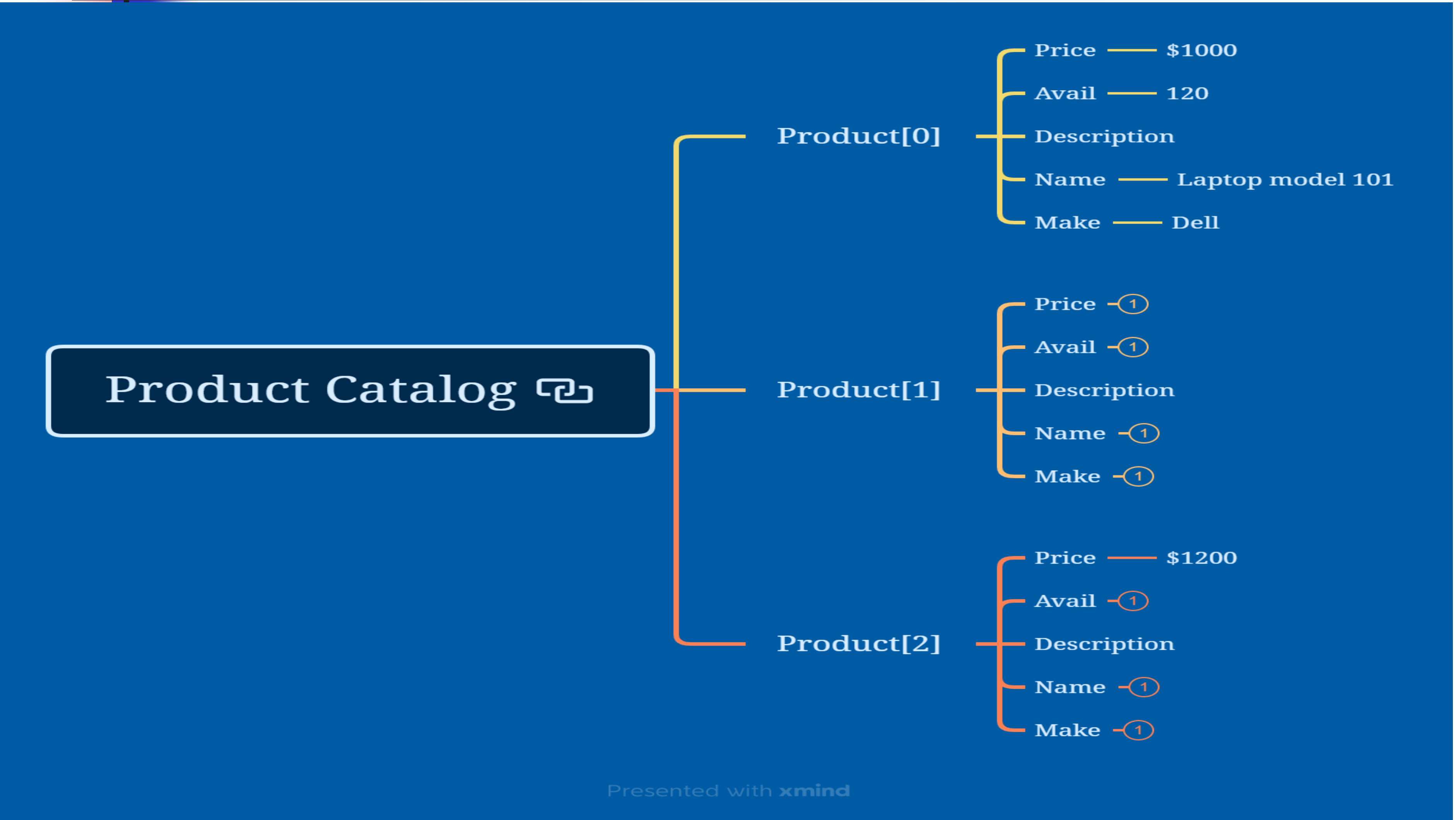
Presented with xmind

Methods for moving data in/out of component



Methods give us ways to push data into the component as well as mechanism for extracting data out of the component

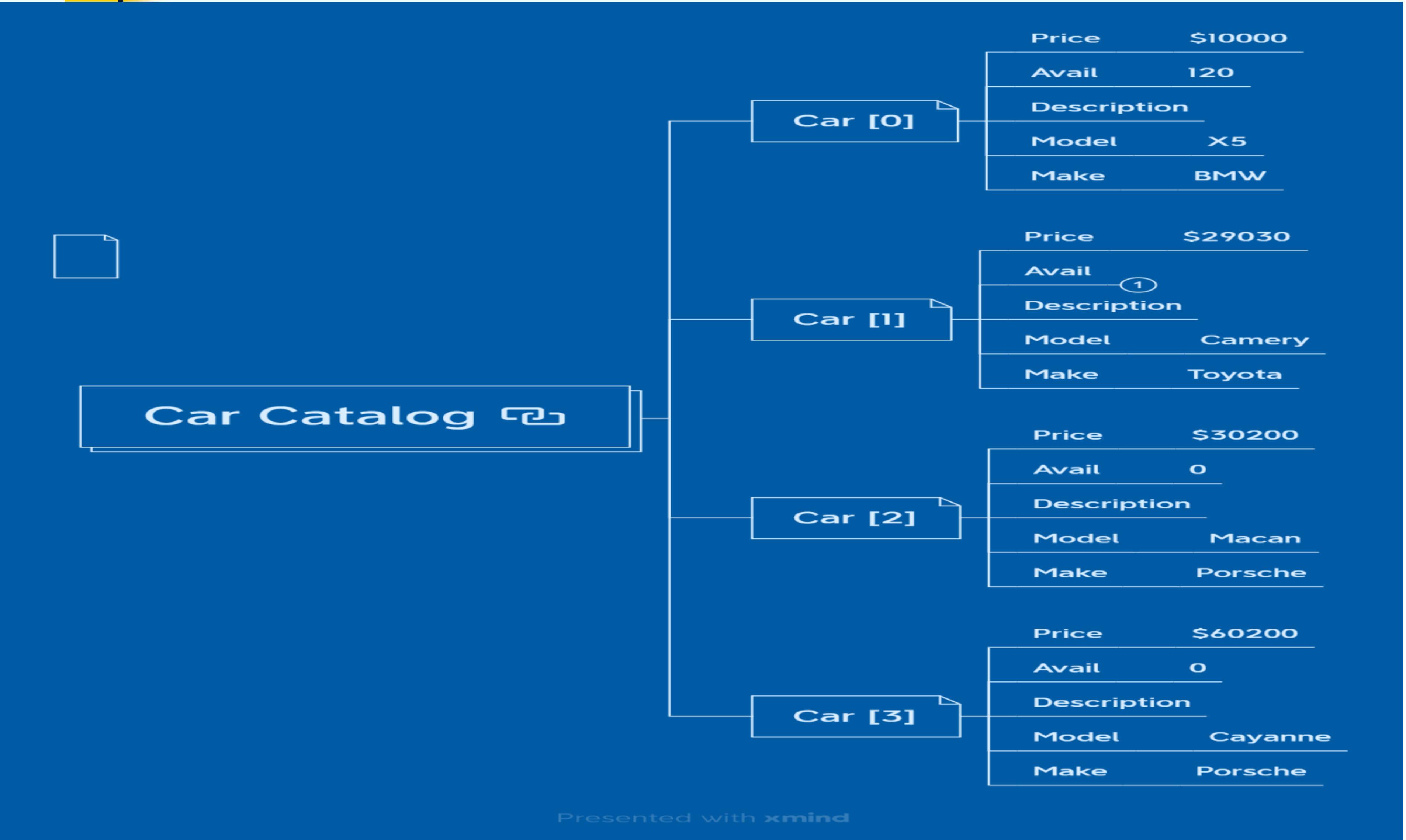
Product Catalog with 3 indexed product instances



- Product Catalog
  - Product[0]
    - Price — $1000
    - Avail — 120
    - Description
    - Name — Laptop model 101
    - Make — Dell
  - Product[1]
    - Price ①
    - Avail ①
    - Description
    - Name ①
    - Make ①
  - Product[2]
    - Price — $1200
    - Avail ①
    - Description
    - Name ①
    - Make ①

Presented with xmind

Car Catalog containing instances of car components



Car Catalog

Car [0]

Price	$10000
Avail	120
Description	
Model	X5
Make	BMW

Car [1]

Price	$29030
Avail	1
Description	
Model	Camery
Make	Toyota

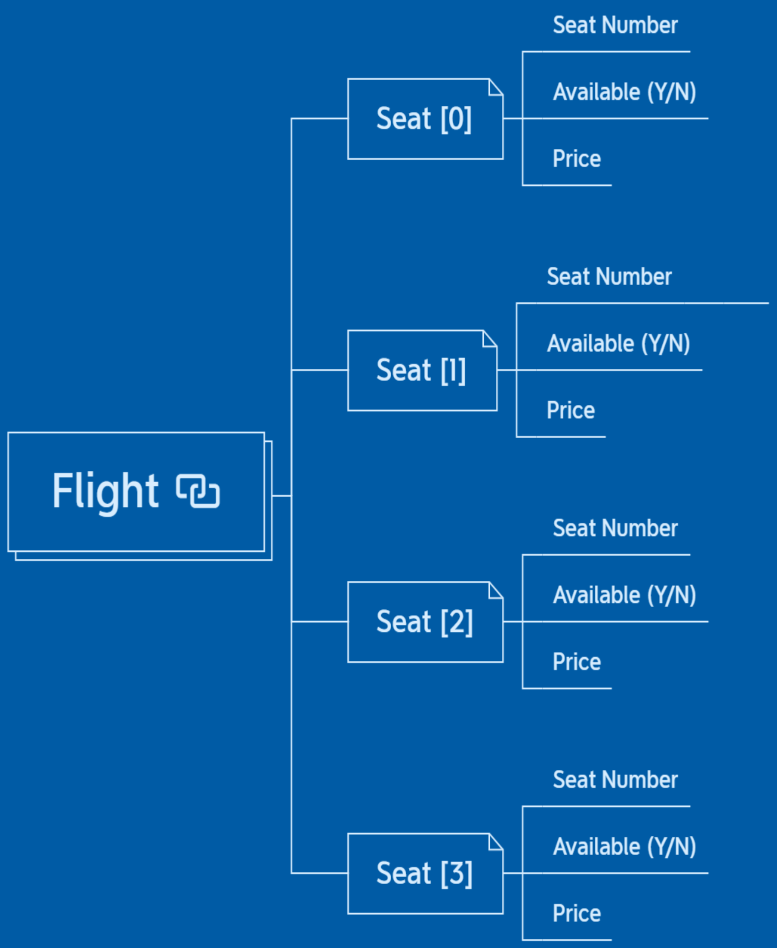
Car [2]

Price	$30200
Avail	0
Description	
Model	Macan
Make	Porsche

Car [3]

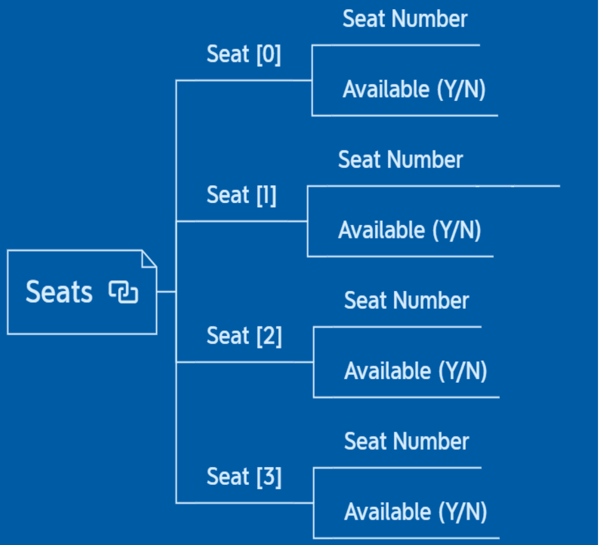
Price	$60200
Avail	0
Description	
Model	Cayanne
Make	Porsche

Presented with xmind



Flight

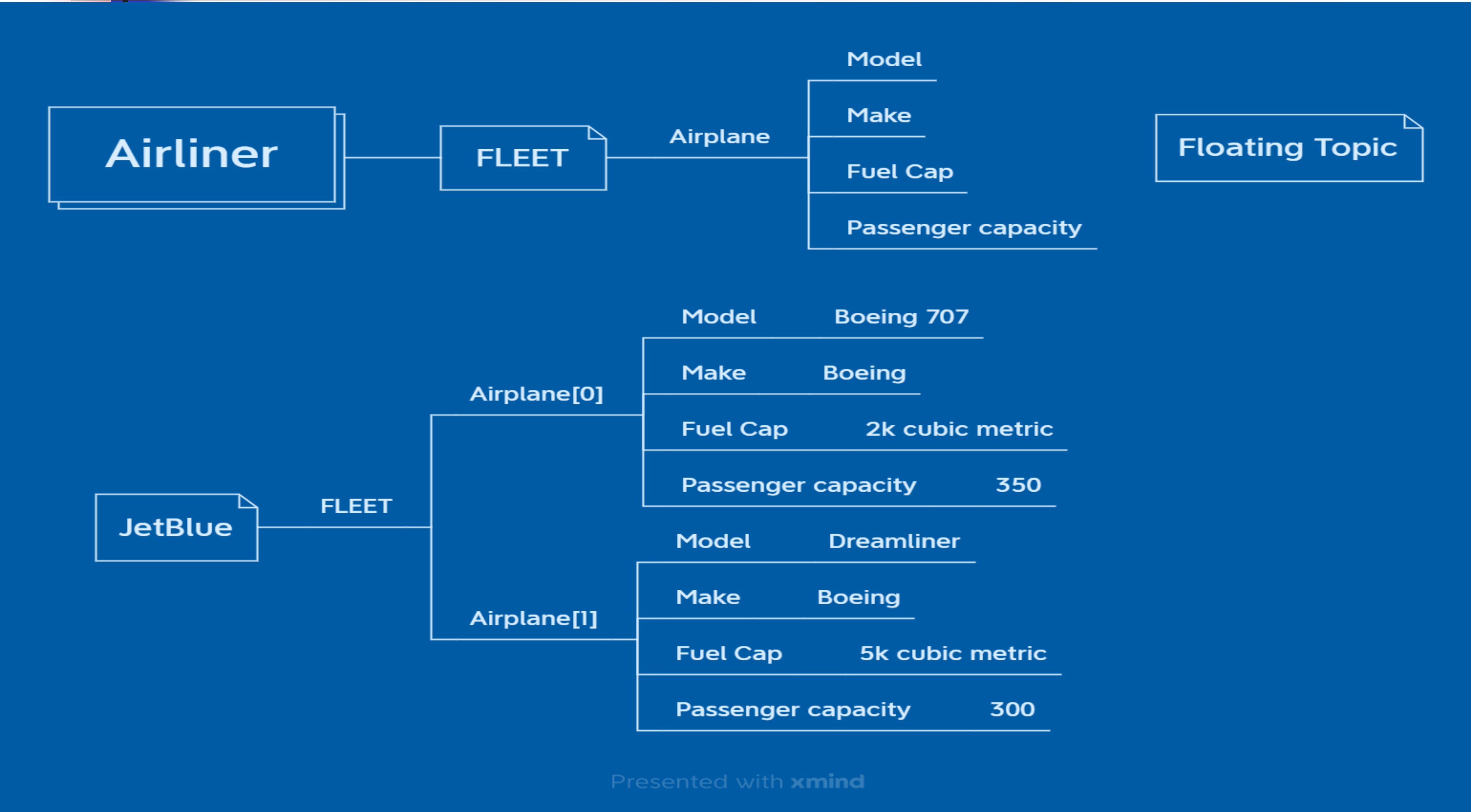
- Seat [0]
  - Seat Number
  - Available (Y/N)
  - Price
- Seat [1]
  - Seat Number
  - Available (Y/N)
  - Price
- Seat [2]
  - Seat Number
  - Available (Y/N)
  - Price
- Seat [3]
  - Seat Number
  - Available (Y/N)
  - Price



Seats

- Seat [0]
  - Seat Number
  - Available (Y/N)
- Seat [1]
  - Seat Number
  - Available (Y/N)
- Seat [2]
  - Seat Number
  - Available (Y/N)
- Seat [3]
  - Seat Number
  - Available (Y/N)

Presented with xmind



Airliner

FLEET

Airplane

Model

Make

Fuel Cap

Passenger capacity

Floating Topic

JetBlue

FLEET

Airplane[0]

Model Boeing 707

Make Boeing

Fuel Cap 2k cubic metric

Passenger capacity 350

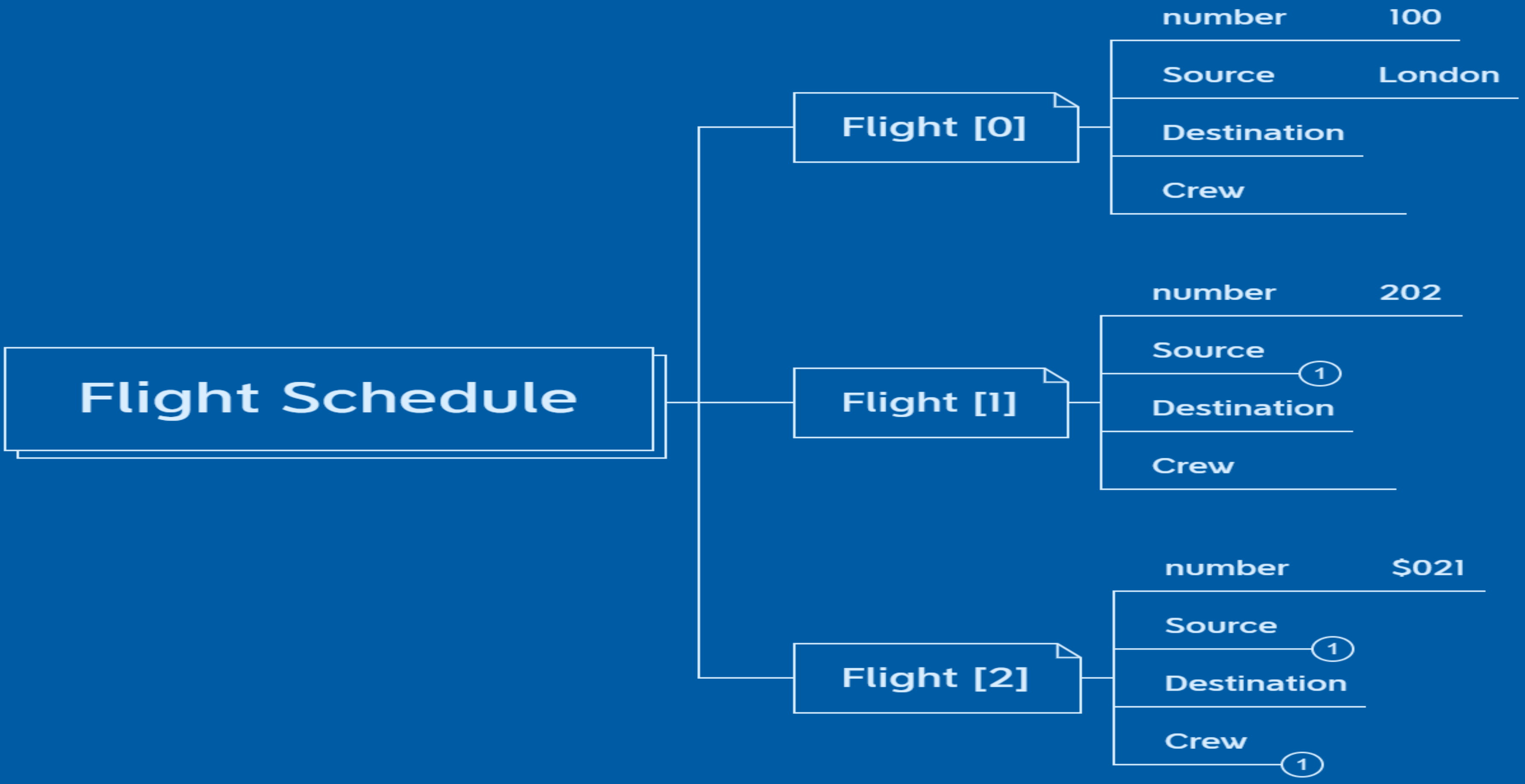
Airplane[1]

Model Dreamliner

Make Boeing

Fuel Cap 5k cubic metric

Passenger capacity 300



Flight Schedule

Flight [0]

number 100

Source London

Destination

Crew

Flight [1]

number 202

Source 1

Destination

Crew

Flight [2]

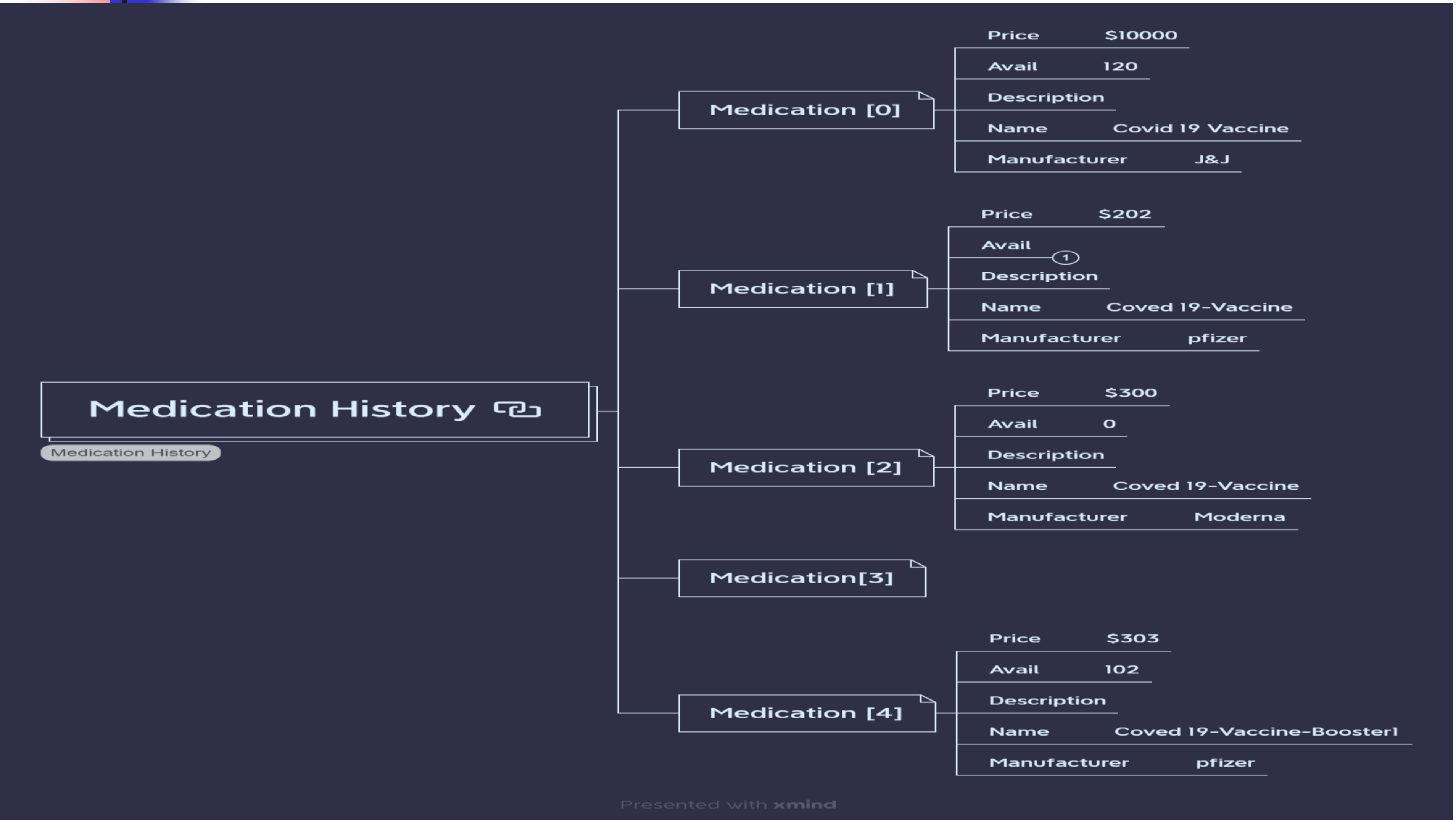
number $021

Source 1

Destination

Crew 1

Presented with xmind



Medication History

Medication History

Medication [0]

Price $10000

Avail 120

Description

Name Covid 19 Vaccine

Manufacturer J&J

Medication [1]

Price $202

Avail 1

Description

Name Coved 19-Vaccine

Manufacturer pfizer

Medication [2]

Price $300

Avail 0

Description

Name Coved 19-Vaccine

Manufacturer Moderna

Medication[3]

Medication [4]

Price $303

Avail 102

Description

Name Coved 19-Vaccine-Booster1

Manufacturer pfizer

Presented with xmind