Design Defects & Restructuring

Week 3: 16 Sep 22

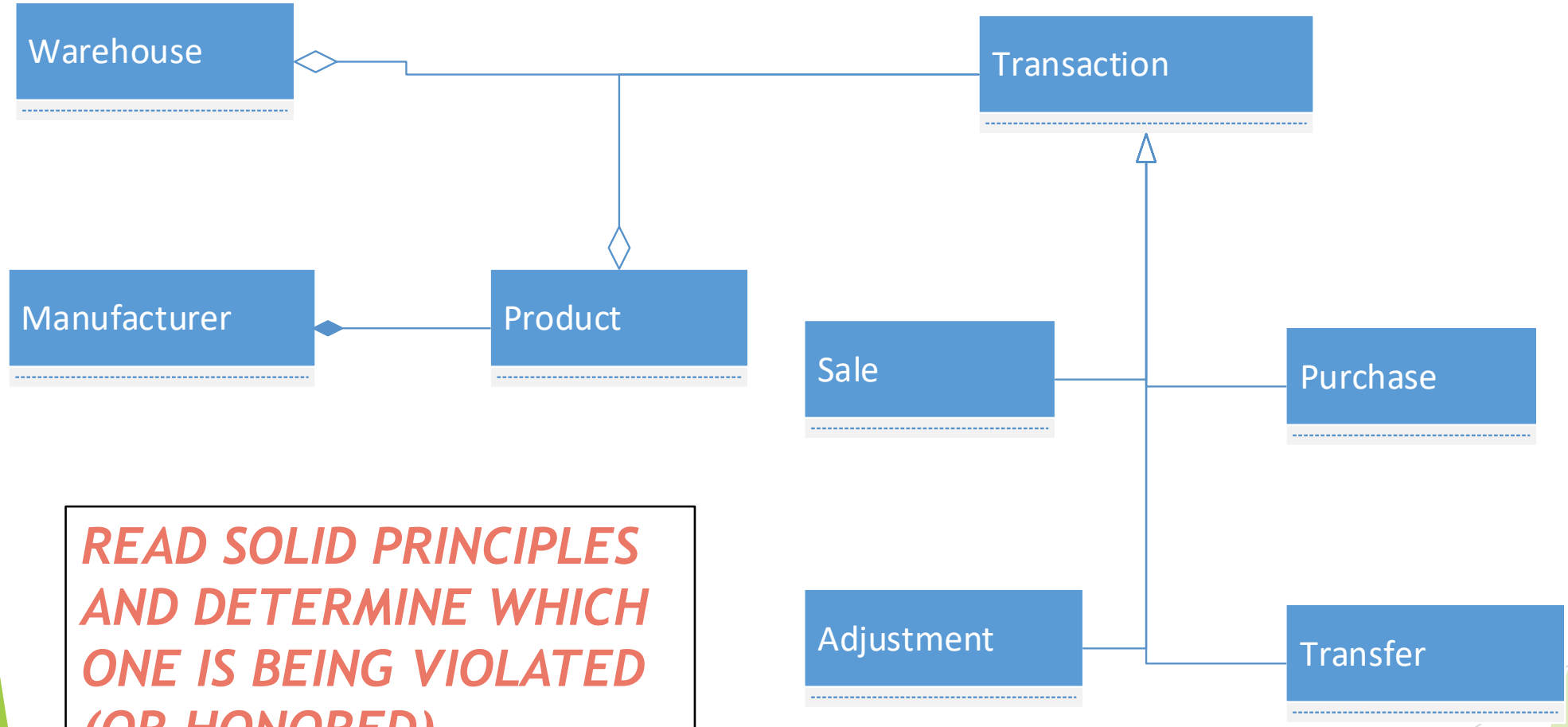
Rahim Hasnani

Agenda

- Home works
- Any questions on last week
- Last week's question
- SOLID Principles
- First Chapter of Head First Design Patterns
- GOF Design Patterns - Introduction

Exercise

- *GoldSoft has recently won a project for development of an inventory system. The inventory system is to be developed for a big distributor with over 1,000 products. The distributor has warehouses where products are stored. The distributor currently carries products of five principals (manufacturers) and this is likely to increase with time. [The main business of a distributor is to order and carry products from principals and distribute them to wholesalers]. The major transactions in the inventory system are 1) Receipt of products from principal 2) Distribution of products to wholesaler 3) Transfer of stock from one warehouse to another 4) Stock adjustment as a result of stock count. Each of these transactions involves a number of products in different quantities (i.e., a receipt usually involves tens or even hundreds of different products in different quantities). Products have different unit of measures (some products are distributed as individual units like keyboard, some are distributed in meters like network wire, some are distributed in Kgs like Sugar, etc).*

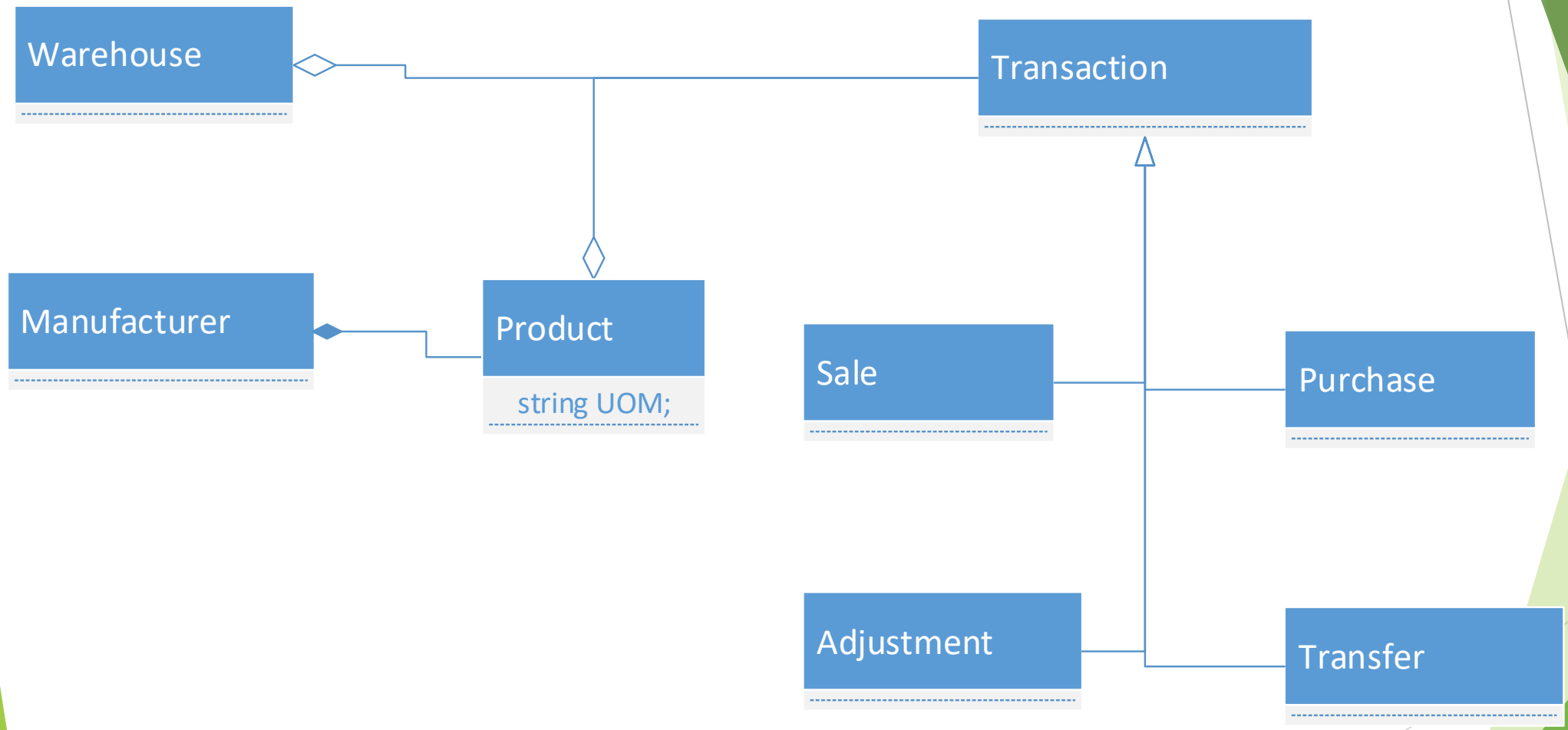


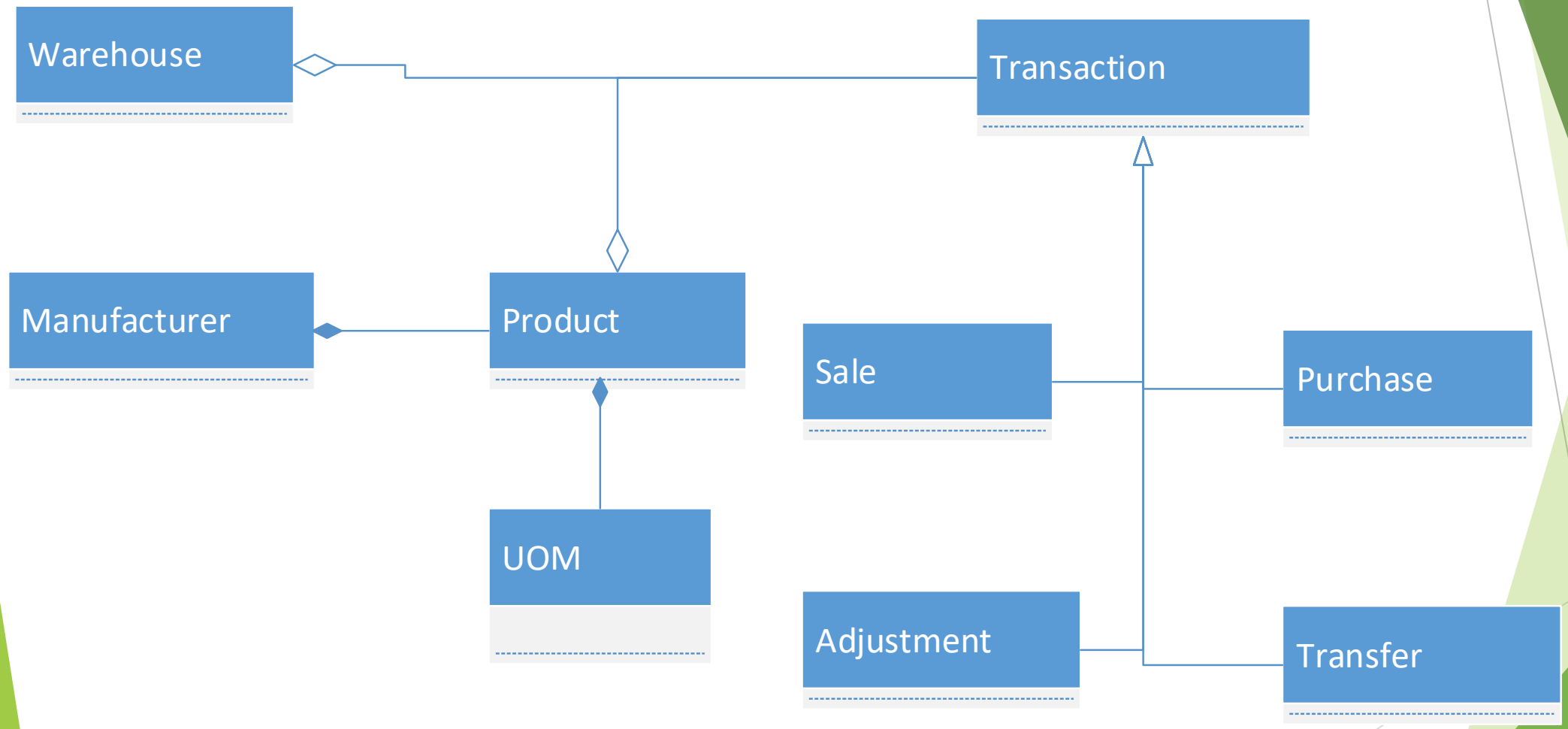
READ SOLID PRINCIPLES AND DETERMINE WHICH ONE IS BEING VIOLATED (OR HONORED)

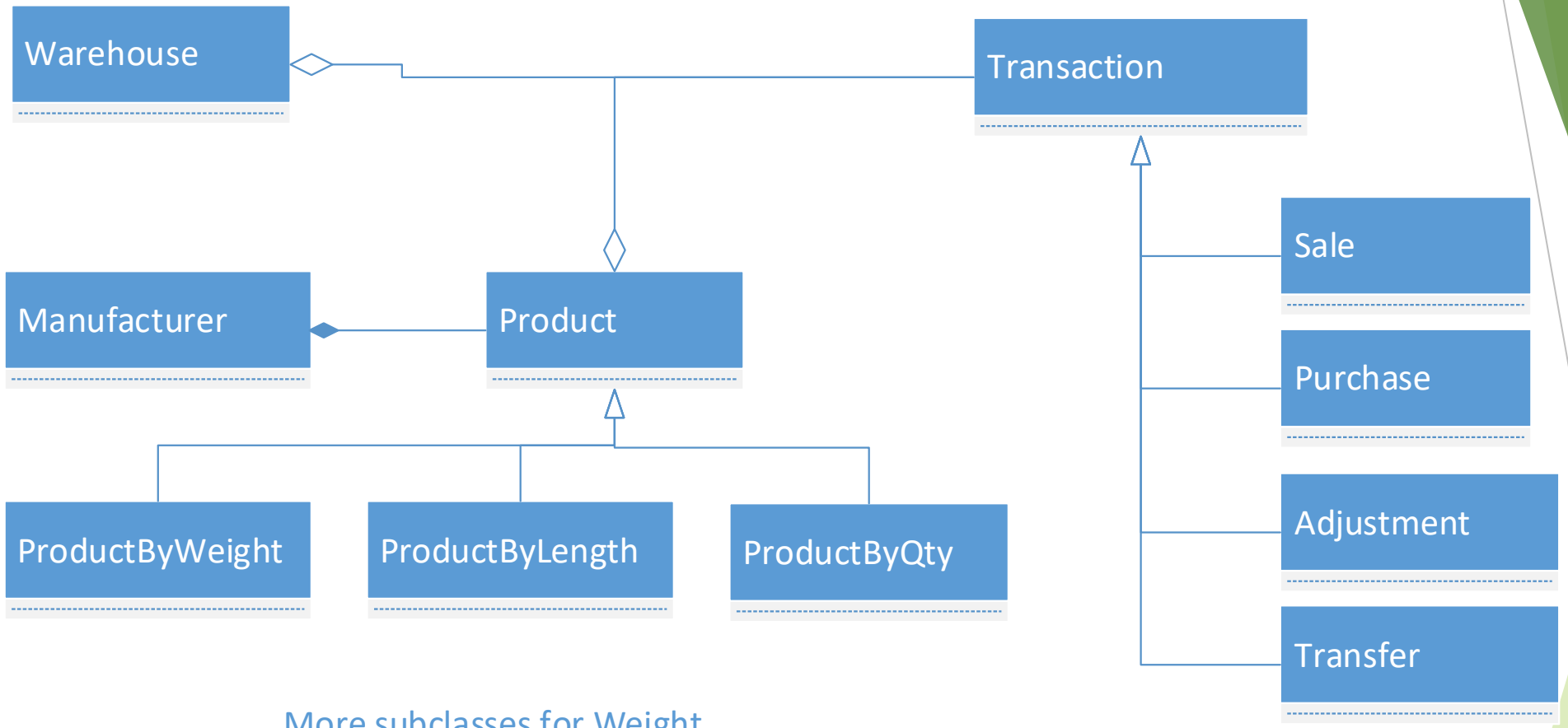
Where is UOM??

Alternatives:

1. Do nothing; UOM is just an attribute
2. Make it a composition: Product has UOM
3. Make it a hierarchy


Warehouse
Transaction
Manufacturer
Product
string UOM;
Sale
Purchase
Adjustment
Transfer


Warehouse
Transaction
Manufacturer
Product
Sale
Purchase
UOM
Adjustment
Transfer



More subclasses for Weight (Kg, Pound, Stones, Tonnes), Length (Meter, Feet, Yard, Inches) and Qty (Singles, 10s, Dozens, Gross)

Where is the Stock

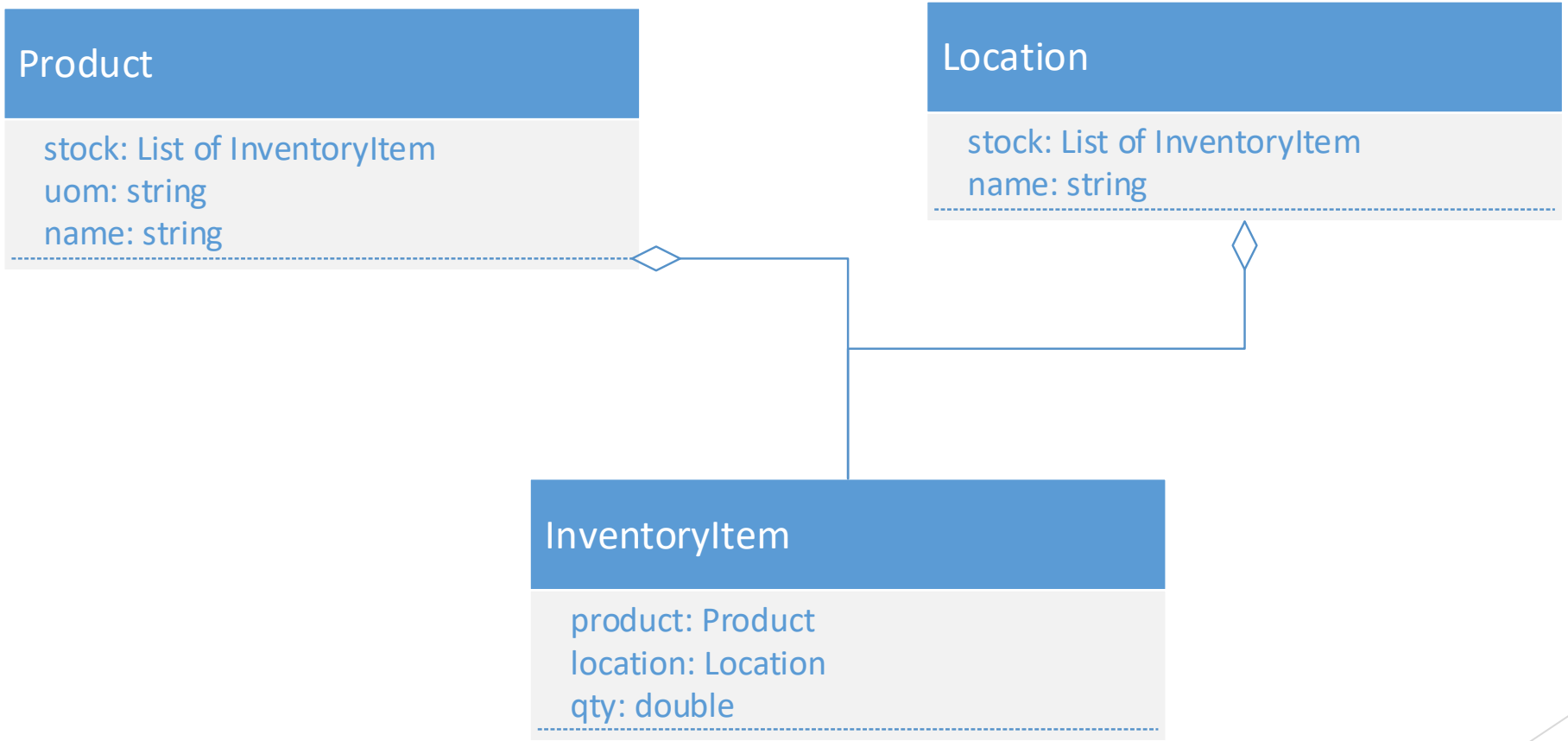
Its an inventory system!

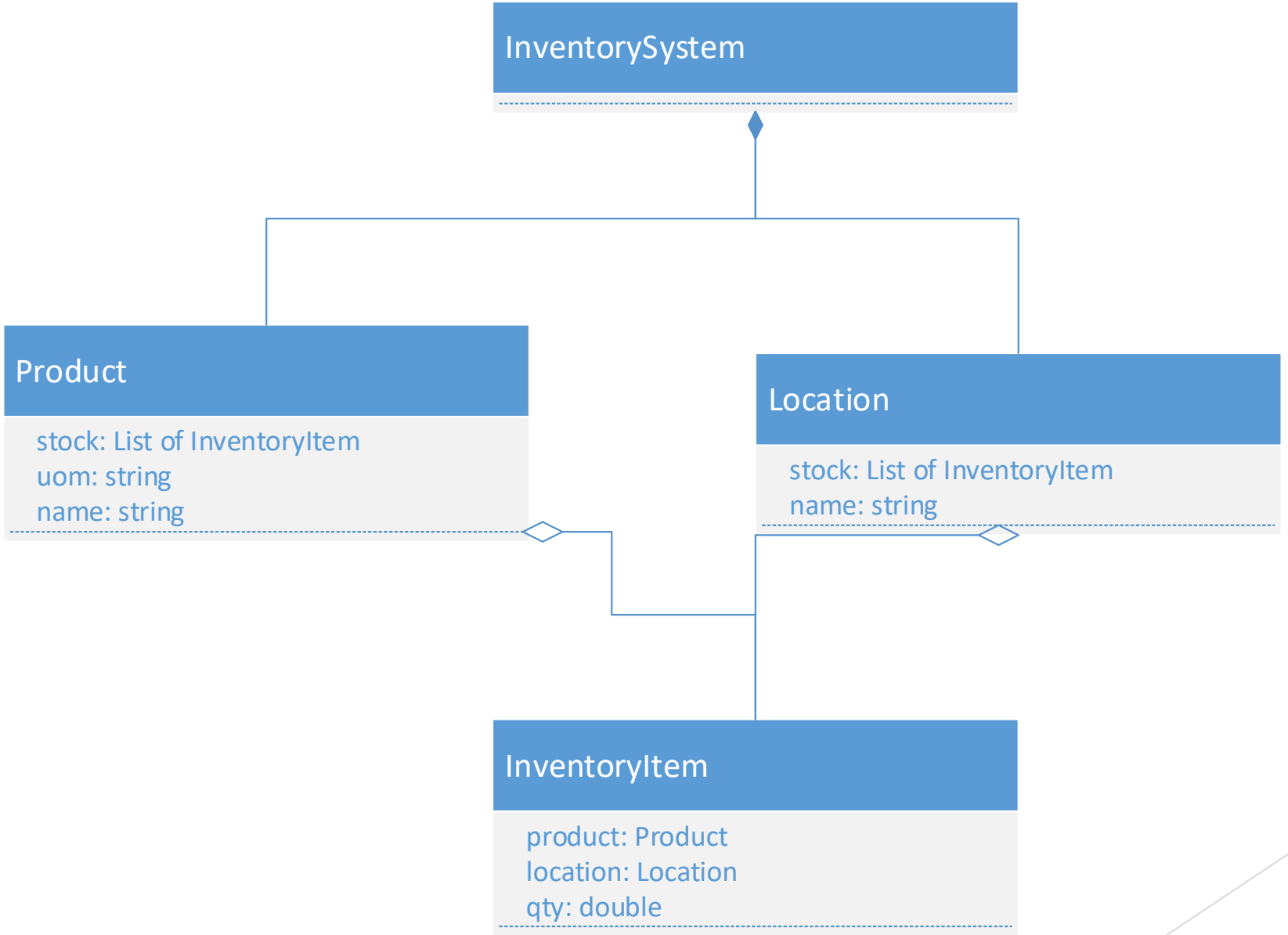
Whole purpose to keep ***track*** of stock, which is:

For a particular product, where it is and in what quantity

For a particular location, what is currently held

So, where is the stock information??


Product
stock: List of InventoryItem
uom: string
name: string
Location
stock: List of InventoryItem
name: string
InventoryItem
product: Product
location: Location
qty: double


InventorySystem
Product
stock: List of InventoryItem
uom: string
name: string
Location
stock: List of InventoryItem
name: string
InventoryItem
product: Product
location: Location
qty: double

What’s wrong with last two designs

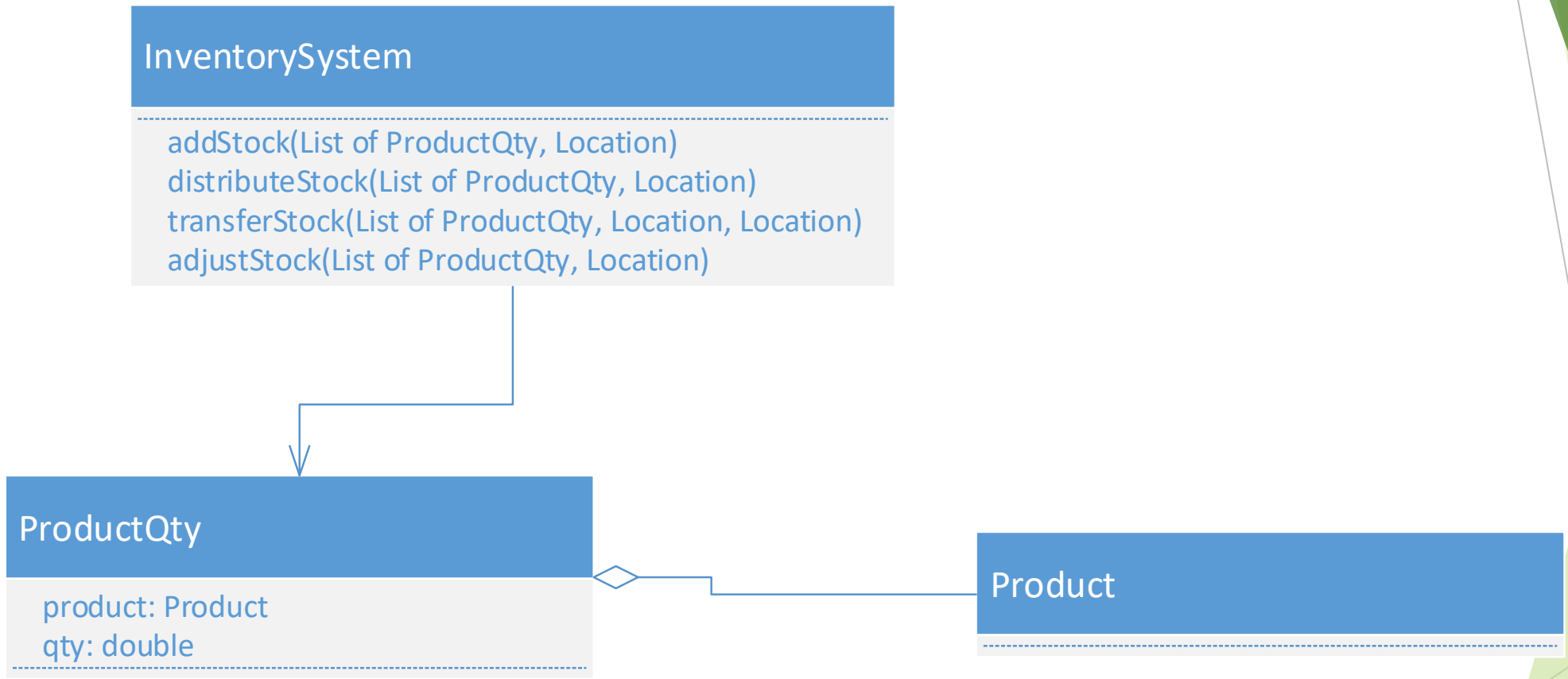
1. Nothing wrong
2. Why are we trying to replicate a database schema in our class diagram?
3. Don’t know; We certainly do not want our class design to look like a database but we still need to carry data, no?

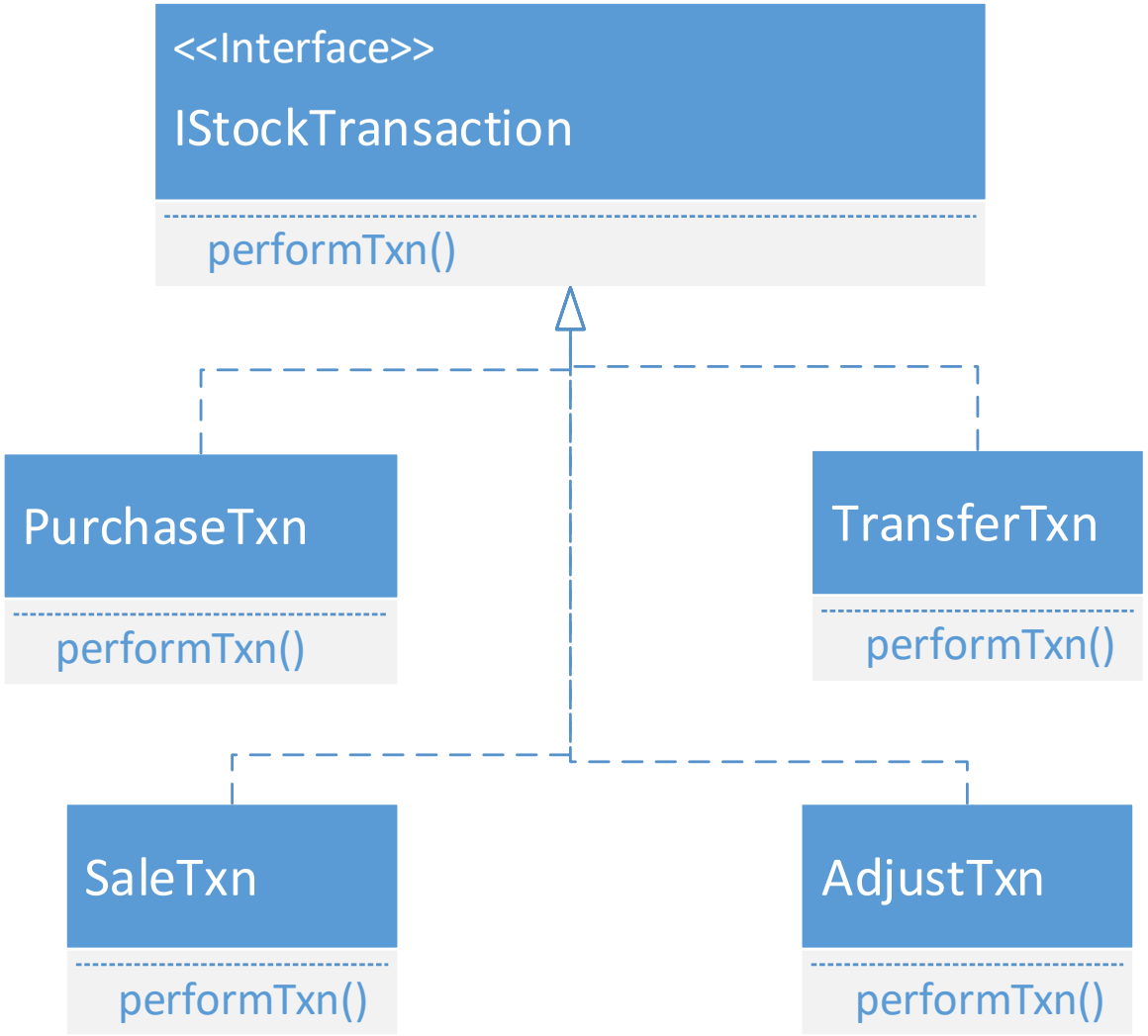
Let’s discuss Transactions

1. Four transactions are described (for now):
    1. Purchase: increases quantities of one or more products in a particular warehouse/location; Comes from ONE manufacturer
    2. Sale: decreases quantities of one or more products in a particular warehouse/location; products could be from multiple manufacturers
    3. Transfer: moves (increases at to end and decreases from from end) quantities of one of more products from one warehouse/location to another warehouse/location; products could be from One or more manufacturer
    4. Adjustment: increases or decreases quantities of one or more products in a single warehouse/location; products could be from multiple manufacturers
2. Is this something that should be abstracted?
3. Can we add more transactions later?
4. Can a non-transfer transaction involves more than one location?

Alternatives:

- Keep transactions as behaviors
- Each Transaction ***is a*** transaction
- Interfaces??


InventorySystem
addStock(List of ProductQty, Location)
distributeStock(List of ProductQty, Location)
transferStock(List of ProductQty, Location, Location)
adjustStock(List of ProductQty, Location)
ProductQty
product: Product
qty: double
Product



WOULD THIS WORK??

GOF Patterns

- Walk-through of First chapter of head first design patterns book